How to Conquor the World with Less Keystrokes

Wrap Your Mind Around Vim

John Filippone

March 27, 2017

0.1 ideas that need to be fit in somewhere

1. This book does not go into detail about how to use vim, just what it does. But because vim is so customizable, the how does not matter very much once you know which features you want to use, if it turns out that the way to do those things in vim takes a lot of keystokes and are therefore not effective, you can remap keys or write scripts to get the same functionality in a totally customized way that works for you

0.2 Purpose

The object of this book is to help the Vim user efficiently learn the tool by enumerating the vast capabilites of Vim 8.0 concisly and simply.

0.3 This Book's Unique Approach

This is not a 'HOW to use Vim' book. It is more of a 'this is WHAT Vim can do; now that you know what it does and what the features are technically called, you can easily learn how to use only the features of Vim that you actually care about on your own' book. The approach of this book is based on the premise that learning HOW to use a particular feature in Vim is easy; the hard part is figureing out the WHAT feature you want to learn. Once you know what the thing you want to do is and what it is called, all it takes some internet searching and some practice. Most people get stuck in Vim because they do not understand the capabilities of the tool or the technical terminology associated with it. Either the user does not know that Vim is capable of doing the thing they want or the user knows what they want and suspects that Vim can do it but has no idea what the feature is called. This book provides a straight forward, consice, and easy-to-understand way of enumerating WHAT Vim does so that you, the user, can get on with the easy part of learning HOW to use only the features you want.

0.4 Important Historical Perspective

When Vim was created, it was made by Bram for Bram, meaning: the engineer who developed Vim initially, developed it for his personal use. He

developed the features that he personally wanted to use. As years went on and Bram continued developing, the community asked for other features that Bram did not necessarily care about. He implemented them anyway as long as he felt enough people would enjoy having them. After decades of development, Vim has become a vast collection of features. There are too many for any one person to really need or want or use them all. The vast set of features is meant to accomodate a vast set of users where each user only needs and/or wants a small subset of all features. It is for this reason that this book focues on efficient learning of Vim rather than complete learning.

0.5 How Most People Learn Vim

Most people are thrust into Vi and/or Vim and are forced to frantically learn the basic editing commands. Chances are that if this is the case we are talking about an engineer or scientist; let's call him Steve. Steve realizes after learning hjkl cursor navigation that the impact of editor efficiency on his workflow is significant so Steve sets out to learn some more useful commands on the internet or perhaps a book. Unfortunately, most books for Vim are recipe books. They provide tutorial on specific features or describe great ways to solve common problems with Vim. These are great for picking up tricks but you will brush through many of these before learning the key set of Vim features that work best for you and the task you are applying them to. Steve gets tired of that pretty quickly. The internet can be very helpful if you know what you are looking for as long as you don't fall into a wormhole of the extremely mystifying power-user language. The weird dialog about buffers and Ex-mode etc. is easy enough for you to sift through but the real trouble for Steve is the former. When you start using Vim you have no idea what you want to learn because you have no idea what Vim can do! Even though Steve has a vuage idea of what functionality he wants, he does not know the proper Vim terminology for the thing he wants to do. That makes searching for a how-to very hard. Naturally, this is where Steve stops learning and decides he is relatively content with getting by on his fair amount of editing efficiency. Every once in a while Steve sees his coworker doing something slick in the editor and says, "hey...how did you do that?" Steve might also learn something new after getting fed up with doing some repetative task and frantically searching the internet for a solution but Steve has the same key issue. Steve is smart and can learn to do just about anything, but when

it comes to Vim he has no idea what it is that he needs to learn.

0.6 How You Ought To Learn Vim

- Understand what Vim has to offer.
- Pick a subset of key features you need to learn in order to have an efficient workflow for your specific task.
- Learn each of these features through the Vim User Manual, books, and/or the internet.
- Practice using them enough to commit them to muscle memory for a truly efficient workflow.

This is an efficient method of learning Vim or any tool. Only learn the features you actually care to use. Know what the tool can do so that later on down the road when you need some other functionality, you know what to look for.

0.7 How to Use This Book

You can read it straight through or you can skip to the parts you really care about. It will be valuable to at least skim all of the material so that you come a way with an understanding of the full scope of Vim capabilites. If you are mystified by the terminology in this book, see the Vim Terminology section in the back. That section will also give you a general scope and sense of how Vim is structured.

0.8 Vim Terminology

Buffers, word, WORD, motions, modes (normal, command, ex, insert, visual), commands, yank v copy, put v paste, registers, text objects, windows, tabs, scripts, plugins, vimrc, dotfiles, NERDTree, Pathogen, Vundle, mapping, abbreviations, macros, autocommands, options, text object text object is like the w in cw command, one edit, movement command, pattern in context of searching, bottom-line command (includes searches and anything on bottom line that is not formally a command)

0.9 Inner workings

General form of (page 21 O'Reilly Learning the Vi and Vim editors) (command)(number)(text object) or (number)(command)(text object) for change, delete, yank, and put what Vim puts in the registers (sometimes called buffers?) switching between modes

0.10 Vim Capabilities

Writing text basic insert/append auto completion start inserting at nifty places end of line start of line smart indenting insert x amount of something at once i.e. 3iabcESC writes abcabcabc Cursor Navigation hjkl search find moving vertically through wrapped line searching by word, sentence, paragraph, and section (can set different macros for identifying paragraphs, and sections) beggining/end of line jump/goto to line by number (G or goto command) to column number with num prefix and — goto top, middle, or bottom line on screen (H, M, L) num prefix to go x lines away from top or bottom scrolling automatic scroll as you type passed bottom scroll by screen or half screen scroll by line with cursor in place scroll relative to cursor (zENTER, z., z-) scroll relative to any line (same as z with numeric prefix) Editing text substitutions deletion command and all variations yank and put replace change and all of its variations: cw, cb, c2b, c0, C, etc type over existing text toggle upper/lower case copy/paste within a single file transpose characters (xp) join lines Searching search forward/backward with wrapping fuzzy find automatically search word under the cursor Environment windows (splits) tabs file browsing buffers? redraw screen Crl-l GUI Color line numbering normal numbers relative numbers Undo/redo multi level undo/redo undo by line repeat an edit Customization Vim script Remapping keys command combinations like ddp File management creating files opening/closing files opening files read only