2.3.1 Text Editors

Click one of the buttons to take you to that part of the video.

Text Editors 0:00-0:59

Knowing how to use a text editor in Linux is really important because most of the configuration changes that you're going to be making to the system will be made using a text file. Therefore, in order to configure a Linux system, you have to, first of all, be able to open a text file. You need to know how to make the appropriate changes, save those changes, and then exit out of the text editor. At this point, you're probably thinking, "Okay. Using a text editor, how hard could that possibly be?" In fact, you're probably already familiar with using some type of text editor, maybe Notepad on Windows.

Here's the problem. The text editor that's used in Linux—"it's named vi—"it's very different from the text editor that's used on other operating systems. It's very different, and frankly, it's a lot more difficult to use than most of your other common text editors. I see many new Linux students try to use the vi editor, and just sit there and scratch their heads, going, "I don't get this."

The vi Editor 1:00-1:24

In this lesson, we're going to talk about the role and function of the vi editor, as well as how to use it to edit basic text files. The vi editor is a text editing program that we can use to edit text files from the command prompt on a Linux system. You don't need a graphical environment to be running.

And the vi editor's been around a very long time. In fact, it's probably the most widely used text editor used by Linux system administrators.

Open an Existing File with vi 1:25-2:24

To start vi, you run either the vi or the vim command, depending on your distribution. Let's first talk about how you open up an existing file with vi.

An example is shown here. You run the 'vi' command, and then you simply tell it what file you want to edit. In this example, we're assuming that the zombie.c file resides in the current directory, so I don't need to provide any path information. If that file did reside in a directory other than the current directory, then, of course, I would need to insert the full path to that file right here. In addition to opening an existing file in vi, it's also possible to create a new text file using vi.

Create a New Text File Using vi 2:25-3:13

The syntax for doing so is exactly the same as that which we use to open an existing file. We enter the vi command, and then we specify the name of the file that we want to create. What vi is going to do is look and see if that file exists in the path that you specified. And if it doesn't exist, it's going to create it for you.

It's important to note that if you do this, if you create a new file with vi, that file doesn't actually get created on disk until you save it. Until you do that, all the lines of text that you enter in the vi interface are only saved in the memory buffer. Therefore, if you don't save the file, you're going to lose it. So far, so good, right? Not too tough to open or create a file with vi.

Four vi Modes 3:14-3:43

However, once that file is either opened or created for editing, that's when things start to get a little confusing when you're using vi. This is because vi actually uses four different operating modes, and you have to be familiar with these if you want to use vi successfully. The first mode is called Command Mode, the second one is called Command Line Mode, which is not the same as Command Mode. You also have Insert Mode, and you also have Replace Mode. Let's take a look at what each one of these modes does.

Command Mode 3:44-5:16

By default, vi opens or creates a file in Command Mode. Sometimes this is also called Normal Mode. In the screen you see here, we have the vi editor opening a file, and it has entered into Command Mode. The confusing part for most students is the fact that while you're in Command Mode, or in Command Line Mode for that matter, you can't actually edit the file. This bugs people because you can see a cursor right there, and I can move that cursor around with my arrow keys. But if I try to start typing text, nothing happens. Before you can edit the text of the file itself, you have to go into Insert Mode. Within Insert Mode, you can actually edit the text. In order to get to Insert Mode from Command Mode, there are several different keys that you can press, and these are listed here.

Insert Mode 5:17-6:33

You can press i, you can press the Insert key on your keyboard, you can press s, you can press o, or you can press a. Any of these keys will switch you from Command Mode into Insert Mode. Once you're in Insert Mode, you can edit the text of the file. Here's a key tip: Notice down here it says INSERT at the bottom of the screen. That's how you can tell whether or not you're in Insert Mode or in Command Mode. If you're in Command Mode, you'll see the name of the file, but you will not see the insert text. If you want to, you can actually press Insert again while you're in insert mode, and that will switch you to a different mode called Replace Mode. Replace Mode also allows you to edit the text of the file directly, but it does things differently.

Replace Mode 6:34-7:14

Replace mode is analogous to overtype mode in that if I start typing new characters--for example, the cursor is right here in this example--if I start typing new characters, the new characters I write will overwrite whatever existing characters already exist in the file. Frankly, I don't find that very useful. I don't use Replace Mode very often. I almost always do my editing in Insert Mode. You can tell you're in Replace Mode by looking at the bottom of the screen. Notice, down here, it says REPLACE. If you end up in Replace Mode, and you don't want to be in Replace Mode--you want to get back to Insert Mode--all you have to do is press the Insert key again. The Insert key toggles you back and forth between Insert Mode and Replace Mode.

Enter Text 7:15-7:30

Once you've opened a file in vi and entered into insert or replace mode, you can then edit the text as you would any other text editor. You just start typing. You can use the arrow keys to move around, you can use the Backspace key and the Delete key, whatever it is you need to do in order to manipulate the text in the file.

Command Line Mode 7:31-8:41

Once you're done editing the text, you need to switch back to Command Line Mode in order to save your changes. You do this by pressing the Escape key. At this point, we've edited the text in the file, and we probably want to save it. Remember, any changes we make to this file while in insert or Replace Mode are not actually committed to disk until we save. It's just like using a word processor.

The vi editor, as we talked about earlier, does not provide any kind of menu that you can click on in order to accomplish file tasks, like opening, and saving, and so on. In order to accomplish this, we have to enter commands. And because of its name, you would think that you would use Command Mode in order to execute these commands. You don't. Instead, you have to enter these commands in Command Line Mode, not Command Mode. We've been in Insert Mode. We've hit Esc to get back to Command Mode. Now, in Command Mode, we need to switch to Command Line Mode. In order to enter Command Line Mode in vi, you enter a colon (:) while you're in Command Mode. Just a simple colon. When you do, you'll see a colon appear at the bottom of the screen. This colon is where you can enter commands, because now we are in Command Line Mode.

Working in Command Mode 8:42-10:40

There are many commands that you can enter at this prompt in order to accomplish file-related tasks. For example, you can use the 'w' command to save the file to disk and leave the file open so you can continue editing, like hitting File > Save in a typical word processor. You can also enter —˜w filename'. This will save the file that you're working on to a different filename. It's basically the same as doing File > Save as in a standard word processor. A command that I use all the time is the exit command. When I'm done editing the file, if I type —˜exit', it will save the current file that I'm working on to disk, and then exit out of the vi editor. The 'wq' command does exactly the same thing. It also saves the file to disk (thus the w) and then exits out of the vi editor. That's the q for quit.

If you've opened a file in vi, you don't make any changes to it, and you want to just get out, you want to close the file--exit the vi editor without saving any changes--you use the q command in Command Line Mode. Be aware that this command will only work if you haven't

made changes to the file. If you've made changes to the file, you'll get a nasty little warning down here when you type the q command, saying, "Hey, changes have been made, I can't quit." If you have made changes to the file and you do not want to save them--you want to exit out without committing the changes to disk--this command is your best friend, 'q!'. This says to the vi editor, "Yeah, I made changes. I know. I don't want them saved. Quit anyway." You can also use the w command with an exclamation point (!) to overwrite the current file.

Another useful one is 'e!', which forgets all the changes you've made since the last write. The 'e!' command is basically the same as doing a revert to saved command in other word processing applications. It's "Let's chuck everything that I've done and reload the last saved version of the file."

Command Mode 10:41-11:09

With that in mind, let's now look at working in Command Mode. You can switch to Command Mode from any other vi mode by pressing the Esc key. For example, if you're in Insert Mode or you're in Replace Mode, you can get back to Command Mode by pressing Esc. Remember, when you're in Command Mode, you can't actually type text into the file, but there are other things you can do to the text in the file.

Cut Commands 11:10-12:35

For example, you can use the dw command to delete the word that comes immediately after the cursor, including the space following the word. This is like a cut because the text is not completely deleted. It's saved in a memory buffer so you can paste it later. Likewise, there's the de command, which will delete the word that comes immediately after the cursor, but does not include the following space. And, again, that text is saved in a memory buffer. These two commands are like different versions of the cut command. The text is removed, placed in a memory buffer, and you can paste it later. The only real difference is whether you include the space after the word or if you don't include the space after the word.

Another delete command you can use is right here. It's —˜d$'. This will delete from the current point of the cursor to the end of the line. This isn't really as much a delete command as it is a cut command because, just as with the other d commands, the text that gets cut is saved into a memory buffer. Our 'd$' cuts from the cursor to the end of the line, the 'dd' command deletes the entire line.

This is another version of the cut operation, because that text is again saved in the memory buffer and you can paste it later. If you have cut text with one of these d commands, you can then use the p command to paste that text that was put in the memory buffer at the current cursor location.

Paste, Undo, and Delete Commands 12:25-13:06

I would arrow over to where I wanted the text to be in the file, and then I would press p while in command mode, and the test would be pasted.

The u command undoes whatever the last action was. If you want to just delete text without putting it in the memory buffer, you can use the D command. This will delete from the current cursor position to the end of the line, and nothing is saved in the memory buffer. This is an actual delete command, not a cut command like the other d commands that we looked at.

Copy and Paste Commands 13:07-13:41

Just as you can cut or delete text in vi, you can also copy text in vi. For example, if you need to copy the entire line in which the cursor currently resides, you enter the 'yy' command, and the entire line is copied into your memory buffer, and you can paste it with 'p'.

You can also use the a key, the lower case a, in order to paste text after the current cursor location. It works in the same way as p. Or you can use A, which will append the text in the memory buffer after the current line, instead of after the current cursor position.

Cursor Movement, File Status, and Text Search Commands 13:42-14:33

The C option changes to the end of the line. The Ctrl + g command displays a status line at the bottom of the interface which will tell you the name of the file, the status of the file, the total number of lines in the file, and the current cursor location. If you need to find text within the file, which is really useful if you're dealing with a very long configuration file, you can enter a forward slash (/) followed by the search term that you want to look for.

The vi editor will jump to the next instance of the term specified in the file. If there are multiple hits found, you can press the End key to jump to the next instance of that search term. Forward slash searches forward in the file. If you want to search backwards in the file, use a question mark (?) instead of a forward slash, and it will do exactly the same thing as the forward slash option. It will just go the other direction. It goes from the current cursor position backwards in the file.

Summary 14:32-14:44

That's it for this lesson. In this lesson we introduced you to the vi editor. We talked about how to open files in vi. We talked about how to create files in vi. We talked about the various vi modes. We talked about how to work in Insert Mode, how to work in Command Line Mode, and how to work in Command Mode.
