

# CHAPTER 26

---

## The vi Editor

### The vi Editor

In this chapter, we'll cover the following topics:

- Regular expressions
- **vi**

Many UNIX users have a Graphical User Interface (GUI) through which they access their UNIX system. The Common Desktop Environment (CDE) is the most commonly used GUI on UNIX systems. It is based on the X Windows System and Motif, which together provide an advanced windowing environment. Chapter 14 in this book is devoted to CDE. Most UNIX GUIs provide a graphical editor. Despite the fact that these graphical editors are a standard part of most GUIs, the visual editor, **vi**, remains the most popular UNIX editor. With many fine graphics-based editors as a standard part of most UNIX GUIs and a plethora of editors available as part of personal computer windowing environments, why am I covering **vi**? The answer is two-fold. First, not everyone using a UNIX system has access to a graphics display and may therefore need to know and use **vi**. Because **vi** comes with most UNIX-based systems and is a powerful editor, many new

UNIX users end up using and liking it. Second, **vi** has traditionally been thought of as *the* UNIX editor. Few UNIX users have not used **vi**. This fact does not mean that it is everyone's primary editor; however, virtually all UNIX users have had some experience with **vi**.

Also, a line editor called **ed** comes with many UNIX systems. It is now seldom used because **vi** is a screen editor. Also available is an enhanced version of **ed** called **ex**. **vi** is much more widely used than either of the line editors, so I'll cover only **vi** in this chapter.

I'll cover the *basics* of using **vi** in this chapter. You can experiment with what is covered here, and if you really like it, you can investigate some of the more advanced features of **vi**. A quick reference card summarizing all the **vi** commands covered in this chapter is included with this book.

The following table is a list of tables in this chapter that summarize some of the more commonly used **vi** commands by function:

Table Number	vi Function
Expr	Regular Expressions
Introduction	Modes and Notations in <b>vi</b>
1	Starting a <b>vi</b> Session
2	Cursor Control Commands in <b>vi</b>
3	Adding Text in <b>vi</b>
4	Deleting Text in <b>vi</b>
5	Changing Text in <b>vi</b>
6	Search and Replace in <b>vi</b>
7	Copying in <b>vi</b>
8	Undo in <b>vi</b>
9	Saving Text and Exiting <b>vi</b>
10	Options in <b>vi</b>
11	Status in <b>vi</b>
12	Positioning and Marking in <b>vi</b>
13	Joining Lines in <b>vi</b>

Table Number	vi Function
14	Cursor Placement and Adjusting Screen in <b>vi</b>
15	Shell Escape Commands in <b>vi</b>
16	Macros and Abbreviations in <b>vi</b>
17	Indenting Text in <b>vi</b>
18	Shell Filters in <b>vi</b>
19	Pattern Matching in <b>vi</b>

## Regular Expression Words-of-Caution

Regular expressions describe patterns for which you are searching. The regular expression usually defines the pattern for which you are searching using wildcards. Since a regular expression defines a pattern you are searching for, the terms *regular expression* and *pattern matching* are often used interchangeably.

Let's get down to a couple of words-of-caution immediately:

- **Regular expressions are different from file-matching patterns used by the shell.** Regular expressions are used by both the shell and many programs, including those covered in this chapter. The file matching done by the shell and programs such as **find** are different from the regular expressions covered in this chapter.
- **Use single quotes around regular expressions.** The meta-characters used in this chapter must be quoted in order to be passed to the shell as an argument. You will, therefore, see most regular expressions in this chapter quoted.

## Expressions Are Strings and Wildcards



When using the programs in this book, such as **grep** and **vi**, you provide a regular expression that the program evaluates. The command will search for the pattern you supply. The pattern could be as simple as a string or it could include wildcards. The wildcards used by many programs are called meta-characters.



Table 26-Expr shows a list of meta-characters and the program(s) to which they apply. Only the programs covered in this book (**awk**, **grep**, **sed**, and **vi**) are shown in Table 26-Expr. These meta-characters may be used with other programs, such as **ed** and **egrep**, as well, which are not covered in the book. Table 26-Expr describes the meta-characters and their use.



**TABLE 26-Expr** Meta-Characters and Programs to Which They Apply

Meta-Character	awk	grep	sed	vi	Use
.	Yes	Yes	Yes	Yes	Match any single character.
*	Yes	Yes	Yes	Yes	Match any number of the single character that precedes *.
[...]	Yes	Yes	Yes	Yes	Match any <i>one</i> of the characters in the set [...].
\$	Yes	Yes	Yes	Yes	Matches the end of the line.
^	Yes	Yes	Yes	Yes	Matches the beginning of the line.
\	Yes	Yes	Yes	Yes	Escape the special character that follows \.
\{n,m\}	Yes	Yes	No	No	Match a range of occurrences of a single character between <i>n</i> and <i>m</i> .

Meta-Character	awk	grep	sed	vi	Use
+	Yes	No	No	No	Match one or more occurrences of the preceding regular expression.
?	Yes	No	No	No	Match zero or one occurrence of the preceding regular expression.
	Yes	No	No	No	The preceding <u>or</u> following regular expression can be matched.
()	Yes	No	No	No	Groups regular expressions in a typical parenthesis fashion.
{\}	No	No	No	Yes	Match a word's beginning or end.

You may want to refer to this table when regular expressions are used for one of the commands in the table.

## Modes and Notations

We're first going to cover some of the fundamentals of the operation of **vi** called modes, and then go over some of the notations used in the tables in this chapter.

A feature of **vi** that often confuses new users is that it has modes. When you are in *command mode*, everything you type is interpreted as a command. In *command mode*, you can specify such actions as the location to which you want the cursor to move. When you are in *input mode*, everything you type is information to be added to the file. *Command mode* is the default when you start **vi**. You can move into *command mode* from *input mode* at any time by pressing the *escape* key. You move into *insert mode* from *command mode* by typing one of the *input mode* commands covered shortly.

**vi** commands don't really have a standard form. For this reason, I cover common notations. Table 26-Introduction summarizes modes and commands in **vi**:

**TABLE 26- Introduction** Modes and Notations in **vi**

Mode or Notation	Description
Command Mode	You are issuing commands such as moving the cursor or deleting text, rather than inserting or changing text when in command mode. You can switch to insert mode by issuing an insert mode command such as <b>i</b> for insert or <b>a</b> for add text.
Insert Mode	You are in insert mode when changing or inserting more than one character of text. You can switch to command mode by pressing the <i>escape</i> key.
: (colon commands)	Commands that start with a <b>:</b> are completed by pressing the <i>return</i> key.
<i>control</i> (^) commands	When a command uses the <i>control</i> (^) key, you press and hold down the <i>control</i> key and then press the next key that is part of the command. For instance, <b>^g</b> means press and hold <i>control</i> and then <b>g</b> to get the status on the file you are editing.
<i>file</i> for the name of a file	Many commands require you to specify the name of a file. For instance, in the command <b>vi file</b> , you would substitute the name of the file you wish to edit for <i>file</i> .
<i>char</i> for the name of a character	Many commands require you to specify a single character. For instance, in the command <b>fchar</b> , you would substitute the character you wish to search for in place of <i>char</i> .