**Using the Text Editors**

Linux distributions include a number of applications known as *text editors* that you can use to create text files or edit system configuration files. Text editors are similar to word processing programs, but generally have fewer features, work only with text files, and might or might not support spell checking or formatting. Text editors range in features and ease of use and are found on nearly every Linux distribution. The number of editors installed on your system depends on what software packages you've installed on the system.

Some of the more popular console-based text editors include:

* *emacs—*The comprehensive GNU emacs editing environment, which is much more than an editor; see the section "Working with emacs" later in this chapter.
* *joe—*Joe's Own Editor, a text editor, which can be used to emulate other editors.
* *nano—*A simple text editor similar to the pico text editor included with the pine email program.
* *vim—*An improved, compatible version of the vi text editor (which we call vi in the rest of this chapter because it has a symbolic link named vi and a symbolically linked manual page).

Note that not all text editors are *screen oriented*, meaning designed for use from a terminal. Some of the text editors are designed to run from a graphical desktop and which provide a graphical interface with menu bars, buttons, scrollbars, and so on, are:

* *gedit—*A GUI text editor for GNOME
* *kate—*A simple KDE text editor
* *kedit—*Another simple KDE text editor

A good reason to learn how to use a text-based editor, such as vi or nano, is that system maintenance and recovery operations almost never take place during X Window sessions, negating the use of a GUI editor. Many larger, more complex and capable editors do not work when Linux is booted to its single-user or maintenance mode. If anything does go wrong with your system and you can't log into the X Window system and its graphical user interface, knowledge and experience of using both the command line and text editors will turn out to be very important. Make a point of opening some of the editors and playing around with them; you never know—you might just thank me someday!

Another reason to learn how to use a text-based editor under the Linux console mode is so that you can edit text files through remote shell sessions, because many servers will not host graphical desktops.

**Working with vi**

The one editor found on nearly every UNIX and Linux system is the vi editor, originally written by Bill Joy. This simple-to-use but incredibly capable editor features a somewhat cryptic command set, but you can put it to use with only a few commands. Although many experienced UNIX and Linux users use viextensively during computing sessions, many users who do only quick and simple editing might not need all its power and may prefer an easier-to-use text editor such as pico or GNU nano. Diehard GNU fans and programmers definitely use emacs.

However, learning how to use vi is a good idea. You might need to edit files on a Linux system with a minimal install, or a remote server without a more extensive offering of installed text editors. Chances are nearly 100 percent that vi will be available.

You can start an editing session by using the vi command like this:

**matthew@seymour:~$** vi file.txt

The vi command works by using an insert (or editing) mode, and a viewing (or command) mode.

When you first start editing, you are in the viewing mode. You can use your arrow or other navigation keys (as shown later) to scroll through the text. To start editing, press the i key to insert text or the a key to append text. When you're finished, use the Esc key to toggle out of the insert or append modes and into the viewing (or command) mode. To enter a command, type a colon (:), followed by the command, such as w to write the file, and press Enter.

Although vi supports many complex editing operations and numerous commands, you can accomplish work by using a few basic commands. These basic vi commands are

* **Cursor movement**—h, j, k, l (left, down, up, and right)
* **Delete character**—x
* **Delete line**—dd
* **Mode toggle**—Esc, Insert (or i)
* **Quit**—:q
* **Quit without saving**—:q!
* **Run a shell command**—:sh (use 'exit' to return)
* **Save file**—:w
* **Text search**—/

**NOTE**

Use the vimtutor command to quickly learn how to use vi's keyboard commands. The tutorial takes less than 30 minutes, and it teaches new users how to start or stop the editor, navigate files, insert and delete text, and perform search, replace, and insert operations.

**Working with emacs**

Richard M. Stallman's GNU emacs editor, like vi, is included with Ubuntu and nearly every other Linux distribution. Unlike other UNIX and Linux text editors, emacs is much more than a simple text editor—it is an editing environment and can be used to compile and build programs and act as an electronic diary, appointment book, and calendar; use it to compose and send electronic mail, read Usenet news, and even play games. The reason for this capability is that emacs contains a built-in language interpreter that uses the Elisp (emacs LISP) programming language. emacs is not installed in Ubuntu by default. To use emacs, the package you need to install is called emacs. See Chapter 32, "Managing Software."

You can start an emacs editing session like this:

**matthew@seymour:~$** emacs file.txt

**TIP**

If you start emacs when using X11, the editor launches in its own floating window. To force emacs to display inside a terminal window instead of its own window (which can be useful if the window is a login at a remote computer), use the -nw command-line option like this: emacs -nw file.txt.

The emacs editor uses an extensive set of keystroke and named commands, but you can work with it by using a basic command subset. Many of these basic commands require you to hold down the Ctrl key, or to first press a *meta* key (generally mapped to the Alt key). The basic commands are listed in Table 4.2.

**Table 4.2. Emacs Editing Commands**

| **Action** | **Command** |
| --- | --- |
| Abort | Ctrl+G |
| Cursor left | Ctrl+B |
| Cursor down | Ctrl+N |
| Cursor right | Ctrl+F |
| Cursor up | Ctrl+P |
| Delete character | Ctrl+D |
| Delete line | Ctrl+K |
| Go to start of line | Ctrl+A |
| Go to end of line | Ctrl+E |
| Help | Ctrl+H |
| Quit | Ctrl+X, Ctrl+C |
| Save As | Ctrl+X, Ctrl+W |
| Save file | Ctrl+X, Ctrl+S |
| Search backward | Ctrl+R |
| Search forward | Ctrl+S |
| Start tutorial | Ctrl+H, T |
| Undo | Ctrl+X, U |