First Steps

Open a terminal window. The window should be empty apart from an input prompt, which will look something like this (your mileage may vary):

```
vibe@debian:~$
```

This prompt shows my user name (vibe), host name (debian) and the directory I'm currently in. The tilde sign (~) is a code denoting the current user's home directory, /home/vibe in my case. When you launch a terminal window, you should start out in your home directory. Verify this by typing in the following and pressing Enter:

```
pwd
```

You have just executed your first shell command. *pwd* means "present working directory" and, as the name implies, shows you the directory you're currently in. Now have a look at the contents of your home directory by entering:

```
ls
```

The →*ls* command lists the contents of the current directory. You will see the files and directories present in your home folder. Use the command →*cd* ("change directory") to move to another directory. Let's move to /etc:

```
cd /etc
```

Execute *pwd* and *ls* again to verify that you have moved to another directory with different contents. You'll see that /etc contains many files. Using the shell's wildcard mechanism, you can make the output of *ls* more selective. Try the following:

```
ls *.conf
```

This will list all files that end in .conf. The wildcard * represents any number of characters. The shell offers more wildcards, but * is the most important one.

You don't have to retype commands that you executed previously. Use the up arrow key on your keyboard to move up in the history of commands entered. Press the key multiple times until you find the command that you want to re-execute and press Enter. Use the down arrow key to move down in the command history.

While we're at it, pressing the left and right arrow keys will move the cursor along the current command line, giving you the opportunity to correct any typos you have made. However, note that using the arrow keys is a rather inefficient way of editing the command line (although it's sufficient for beginners). Appendix B [see the printed book] presents some keyboard shortcuts that can speed up your editing dramatically.

Use the *cat* command to display the contents of the file /etc/group:

```
cat group
```

Because you're in the directory /etc, you don't have to specify the file's full name /etc/group when invoking cat or another command on it. If you want to view the same file from another directory (e.g. your home folder), you have to provide its full name, like this:

```
cat /etc/group
```

I'm going to use full names in the following examples, just to make sure that they work regardless of your present working directory. You can abbreviate some of the examples by changing to the directory that contains the file(s) acted on.

From the shell's abstract perspective, a command line consists of a number of so-called arguments. The first argument on the command line that we just executed was *cat*, the second argument was *group*. Arguments are separated by whitespace. If you fail to insert whitespace between arguments, or if you insert whitespace in the wrong places, you'll get an error message. Another common stumbling block for beginners is that the shell and the file system are case-sensitive. Invoking *cat Group* or *Cat group* will give you an error.