Preface

UNIX systems usually provide several shells from which you can choose, either vendor-supplied or available from other sources. This handbook focuses on two of them. The first is the C shell (*csh*), a popular command interpreter that has its origins in Berkeley UNIX. *csh* is particularly suited for interactive use and offers a number of features to make it easy to issue commands, such as the ability to recall and modify previous commands, a facility for creating command shortcuts, shorthand notation for pathnames to home directories, and job control.

The second shell is *tcsh*, an enhanced version of *csh*. *tcsh* is almost entirely upward compatible with *csh*, so whatever you know about the C shell you can apply immediately to *tcsh*. But *tcsh* goes beyond *csh*, adding capabilities like a general purpose command line editor, spelling correction, and programmable command, file, and user name completion.

Shells other than *csh* and *tcsh* may be available on your system. The two most significant ones are the Bourne shell (*sh*) and the Korn shell (*ksh*). The Bourne shell was the earliest of the currently popular shells and is the most widely available. The Korn shell was developed at AT&T and is most prevalent on System V-based UNIX systems. Both shells are amply documented elsewhere, so we won't deal with them here.

Purpose of This Handbook

This handbook is designed to help you use *csh* or *tcsh* more effectively. By demonstrating what these shells can do for you and illustrating techniques for using their features, it will show you how to get your work done more quickly and easily. You can increase your effectiveness on a number of fronts, ranging from learning how to repeat the previous command without typing the whole thing in again, to learning how to use filename patterns, to using the shell startup files to make your working environment more pleasant and productive.

This handbook has two particular emphases:

- Interactive use of the shell is stressed. Although a shell can be used non-interactively by telling it to execute commands contained in a file, the majority of the time you spend with your shell is probably at the command line. This handbook reflects that pattern of use by concentrating on how to improve your interaction with the shell rather than on writing shell scripts. Another reason for emphasizing interactive use over scripting is that *csh* and *tcsh* are not good shells for writing scripts (Appendix C, *Other Sources of Information*, references a document that describes why). *sh* or *perl* are better for writing scripts, so there is little reason to dicuss doing so with *csh* or *tcsh*.
- Special features of *tcsh* are discussed extensively. Most shell books mention *tcsh* not at all or only in passing as an obscure relative of *csh*. This handbook discusses the many capabilities *csh* and *tcsh* have in common, but it also describes *tcsh*-specific features throughout rather than ignoring them. In fact, I recommend that you use *tcsh* as your preferred shell—it's just a lot better than *csh*.

Intended Audience

This handbook can be used by anyone at any level of proficiency with *csh* or *tcsh*. It should be helpful in any of the following circumstances:

- If you don't know much about the shell, a simple introduction is provided to help you avoid the frustration that can be a part of the early learning stages. If you're new not only to the shell, but also to UNIX, that's fine, too. This handbook isn't designed as a general UNIX tutorial, but it contains many examples showing how to use UNIX commands.

- If you're already familiar with the fundamentals of the shell, this handbook will help you get more out of *csh* and *tcsh* by demonstrating how to use their interactive capabilities with greater effectiveness. It will show you how to compose commands that do more by building on the skills you already have, and how to save time by issuing those commands more quickly and with less typing.
- If you've used only *csh* and are wondering about *tcsh*, you'll find out about the latter's special features and discover how *tcsh* builds on *csh*'s capabilities to provide a more productive working environment.

Scope of This Handbook

This handbook is divided into three parts. Part I is designed to get you off the ground by providing a basic introduction to the shell and a broad survey that shows how to use the shell's more advanced features to accomplish many different kinds of tasks.

Part II provides a more topical approach to the shell's capabilities. The chapters in Part II focus on particular aspects of the shell. You'll find these chapters useful when you want more information about features of the shell demonstrated in Part I.

The appendices provide information about obtaining and installing *tcsh*, a quick reference to those shell capabilities discussed in this handbook, and pointers to other documentation relating to *csh* and *tcsh*.

Part I

Chapter 1, *Introduction*, discusses the purpose of the shell and describes how to change your login shell to one of those discussed in this handbook.

Chapter 2, *A Shell Primer*, provides a summary of introductory essentials. If you already know something about running commands, you can skim this chapter as a review or skip it.

Chapter 3, *Using the Shell Effectively*, is an overview of many of the facilities that the shell provides to help you work more efficiently—easy, practical ways to get more out of the command line. Often users are not aware of these features. This chapter demonstrates a number of them and gives you a glimpse of the kinds of techniques that are covered in detail in the rest of the book.

Part II

Chapter 4, *The Shell Startup Files*, describes *~/.cshrc* and *~/.login*, the files that the shell uses to initialize your working environment. It also provides guidelines for modifying the startup files to suit your preferences.

Chapter 5, *Setting Up Your Terminal*, discusses how to find out what your terminal's special characters are (e.g., for backspacing) and how to change them if you don't like the default settings.

Chapter 6, *Using Your Command History*, discusses how to type less by using your command history.

Chapter 7, *The tcsh Command Line Editor*, describes the facilities offered by the editor that *tcsh* provides for interactively editing commands.

Chapter 8, *Using Aliases To Create Command Shortcuts*, discusses how to type less by using the shell's alias mechanism to create command shortcuts.

Chapter 9, *Using File Naming Shortcuts*, describes ways in which the shell can supply filename arguments for you so you don't have to type them all out yourself.

Chapter 10, *Using Filename and Programmed Completion*, describes how you can use the shell to complete filenames after you've typed just the leading prefix. It also discusses how *tcsh* can complete other types of words besides filenames and how to set up programmed completions.

Chapter 11, *Quoting and Special Characters*, describes the shell's quoting rules. When you know these rules, you can type any characters in a command line, even special ones, without having them misinterpreted.

Chapter 12, *Using Commands To Generate Arguments*, discusses command substitution, a facility that lets you construct pieces of a command line from the output of other commands.

Chapter 13, *Navigating the File System*, describes how to move around in the file system efficiently.

Chapter 14, *Keeping Track of Where You Are*, describes how to stay informed about your current location in the file system.

Chapter 15, *Job Control*, shows how to suspend, resume, and kill jobs, switch between jobs, and move jobs between the foreground and background.

Appendices

Appendix A, *Obtaining and Installing tcsh*, describes how to make *tcsh* available on your system if you don't have the current version.

Appendix B, *csh & tcsh Quick Reference*, is a summary of those features and capabilities of *csh* and *tcsh* described in this handbook.

Appendix C, *Other Sources of Information*, contains pointers to other documentation. The most important of these is the *tcsh* manual page. It's available in several forms, one of which can be navigated via hypertext links with a World Wide Web browser.

How To Read This Handbook

First you should read Chapter 1, *Introduction*. If necessary, change your login shell to *csh* or *tcsh* using the instructions there.

If you're a beginner unfamiliar with the shell or just want to review the basics of executing simple commands, you should then proceed to Chapter 2, *A Shell Primer*. If you're also having trouble figuring out how to do things like getting your terminal to backspace correctly, you may find it helpful to read Chapter 5, *Setting Up Your Terminal*.

If you already know something about the shell, you can skip past the *Primer* to Chapter 3, *Using the Shell Effectively*. This chapter is designed to ‘‘jump start’’ you into better use of the shell by presenting lots of shortcuts that save you time and effort. You'll quickly become a more fluent command line user by applying the information you find here.

For a more systematic and topical approach to the shell's capabilities, look at the chapters in Part II. These explain in greater detail the features of the shell more briefly demonstrated in Chapter 3.

Finally, use the index. The shell's features interrelate, so you'll often find that a particular capability can be applied in many different contexts, and thus is likely to be discussed in several places.

Conventions Used in This Handbook

Certain special characters on your keyboard are represented in the text using all caps, e.g., `TAB` for the "tab" key. Other special characters are `RETURN` (sometimes labeled `ENTER`), `ESC` (sometimes labeled `ESCAPE`), `DEL` (sometimes labeled `DELETE` or `RUBOUT`), and `BACKSPACE` (which is the same as `CTRL-H`). Control characters, which you enter by holding down the `CTRL` (or `CTL`, or `CONTROL`) key and typing another key, are represented as `CTRL-`*`X`*, where *`X`* is some letter. (You might need the `SHIFT` key for non-alphabetic characters. For instance, if `@` is on the same key as `2`, you might have to hold down `SHIFT` to type `CTRL-@`.)

This book uses the following typographical conventions:

Italic is used for UNIX commands, filenames, and variables.

Bold is used occasionally for emphasis.

`Constant Width` is used in examples to show the contents of files or the output from commands.

`Constant Bold` is used in examples to show interaction between you and the shell; any text you type in is shown in **`Constant Bold`**. For example:

```
% chsh                       Run the change-shell command
Old shell: /bin/csh          chsh displays your current shell
New shell: /bin/tcsh         Type in the shell you want to use instead
```

`Constant Italic` is used in examples to represent dummy parameters that should be replaced with an actual value.

`Reverse Video` is used to show the position of the cursor on a command line. For example, the cursor is positioned on the asterisk in the command below:

```
% sed -e ’s/ˆ *//’ datafile
```

`Box` is used in examples to show where you need to type a special character. In the example below, you type a `CTRL-V` and a `TAB` between the quotes:

```
% grep "CTRL-VTAB" myfile
```

Comments and Corrections

If you have comments on this book or suggestions for improvement, or if you find errors, please contact me by sending electronic mail to *dubois@primate.wisc.edu*.

Acknowledgments