The journal of Apple technology.

Volume Number: 18 (2002)
Issue Number: 06
Column Tag: Section 7

Some Basic Commands

by Rich Morin

Learn these first; the others can wait!

Before we look at any specific commands, we need to look at some fundamental differences between BSD terminal sessions and the ordinary Mac OS X (OSX) experience. Each interface has its advantages and disadvantages; by understanding these, you'll be able to use both interfaces more effectively!

- Applications vs. Commands OSX applications tend to be highly interactive, providing customized environments for accomplishing particular tasks. BSD commands, in contrast, are generally run in "batch mode". The user issues the command, giving all of the needed information, waits for the result, and goes on.
- Documents vs. Files OSX documents are generally encoded in an application-specific data format. Consequently, it is only by extra effort that other applications can make use of them. BSD files are expected to be useful to a wide range of commands. As a result, they are often formatted as ASCII text files, using white space (blanks, spaces, and newlines) to delineate internal data elements.
- Subjects vs. Verbs In OSX, the user double-clicks on the icon for a document (subject) and the appropriate application (verb) starts up. Alternatively, s/he drags the document icon (subject) over to the application icon (verb). In BSD, the user types in the name of a command (verb), followed by assorted file names (subjects) and flags (adverbs).
- Mice vs. Keyboards OSX employs a graphical user interface (GUI), using the mouse (trackball, ...) for nearly every action. Keyboard "shortcuts" are provided in some cases, but they are never required. In BSD, the situation is exactly reversed. The keyboard is used for everything; mouse "shortcuts" are used only occasionally.
- Windows vs. Sessions OSX applications often put up multiple windows; actions taken in one window are expected to have ramifications in all of the other windows. Each BSD terminal session, in contrast, is expected to be independent of all of the other sessions.
- Aqua vs. The Shell Aqua (including the Dock and the Finder) allows the user to navigate through the file system, manipulate documents and folders, and start up commands. The BSD "shell" (assisted by some common commands) provides equivalent capabilities, plus others (e.g., scripting, session logging). Of course, the user interfaces vary substantially!

Navigation Commands

As noted above, BSD command lines start with a verb, followed by some number of subjects and adverbs. Last month, we used the "man" command to view manual pages; let's use it again, trying out some variations.

```
[localhost:~] rdm% man 2 sync
```

```
[localhost:~] rdm% man 8 sync
[localhost:~] rdm% man sync
```

The sync(2) manual page describes a system call; sync(8), in contrast, is a system maintenance command. By specifying the section number, we can tell man which part of the manual to search. In most cases, there is no conflict, so the section number can be left off.

None of man's "arguments" are actually file names. Instead, they are hints that allow man to search assorted directories. The actual list of directories that man examines is specified by MANPATH, a BSD "environment variable". Environment variables and control files perform many of the functions of OSX preferences:

```
[localhost:~] rdm% echo $MANPATH
/Users/rdm/man:/usr/local/share/man:/usr/share/man
```

This tells us that man looks first under /Users/rdm/man (the user's personal man pages), /usr/local/share/man (this machine's "local" man pages), and /usr/share/man (OSX man pages). Let's wander over to the latter directory and take a look:

```
[localhost:~] rdm% cd /usr/share/man
[localhost:/usr/share/man] rdm% ls -F
man1/       man3/       man5/       man7/       whatis.db
man2/       man4/       man6/       man8/
```

The cd(1) command sets /usr/share/man as the "current directory" for this terminal session. This can save a lot of typing! For convenience, the shell puts the name of the current directory in the command line's "prompt string".

The ls(1) command provides a directory listing, appending slashes to directory names (as directed by the -F option). So far, the Finder seems a lot more convenient, but hang on a bit! Let's find out which directories have "sync" man pages:

```
[localhost:/usr/share/man] rdm% ls */sync*
man2/sync.2  man8/sync.8
```

The asterisk "wild cards" tell the shell to look through every subdirectory, looking for files whose names begin with "sync". Hmm; looks like the shell has some advantages when lots of items are involved. How many pages might that be, anyway?

```
[localhost:/usr/share/man] rdm% ls */* | wc -l
    2853
```

This command "pipeline" ran two commands, directing the "standard output" of one into the "standard input" of the other. The first command listed every file in every subdirectory; the second counted the number of lines in the first command's output. The result (2853) was something which neither base command was "designed" to produce.

This pipeline is actually a minuscule instance of a "shell script". BSD users frequently write scripts to automate repetitive tasks. Next month, we'll look at some more commands and some fancier ways of writing shell scripts.

Rich Morin has been using computers since 1970, Unix since 1986, and Mac-based Unix since 1986 (when he helped Apple create A/UX 1.0). When he isn't writing this column, Rich runs Prime Time Freeware (www.ptf.com), a publisher of books and CD-ROMs for the Free and Open Source software community. Feel free to write to Rich at rdm@ptf.com.