Linux/UNIX Cheat Sheet

Navigation

`pwd`	Prints current directory.
`cd` */some/path/*	Change current directory to */some/path/*.
`cd`	Change current directory to home directory.
`cd ~`	Change current directory to home directory.
`cd ..`	Move up one folder.
`cd -`	Revert the current directory to last one.

List Files

`ls`	List files in current directory.
`-a`	List all files (including hidden ones).
`-l`	Long listing of files.
`-F`	Append symbol to signify special files. `/` for directories `@` for symbolic links
`--color`	Enable colorful output (Linux).
`-G`	Enable colorful output (Mac OS X).

Creating Files

`mkdir` *folder1*	Create folder *folder1*.
`touch` *file1*	Create *file1* if it doesn't exist.

Viewing Files

`file` */some/path*	Identify the type of */some/path*.
`cat` *file1 file2 ...*	Display one or more files.
`less` */some/path*	Versatile file view. Press `h` for help.

Manipulating Files

`cp` *file1 file2*	Make a copy of *file1* named *file2*.
`cp` *file1 folder1*	Make a copy of *file1* in *folder1*.
`cp -a` *folder1 folder2*	Make a copy of *folder1* named *folder2*.
`mv` *p1 p2*	If *p2* is not a folder, renames *p1* to it.
`mv` *path1 path2*	If *path2* is a folder, moves *path1* to it.
`rm` *file1*	Delete *file1*. There is **no** Recycle Bin!
`rmdir` *folder1*	Delete *folder1* if empty.
`rm -rf` *folder1*	Delete *folder1* and its contents.

All of these commands accept a `-v` option to explicitly say the file(s) they are affecting. `cp`, `mv`, and `rm` also have an `-i` option to confirm any dangerous operation.

Editing Files

`nano` *file1*	Run a small, friendly editor In shortcut listing, `^X` means Control-X and `M^X` means Alt-X (on most keyboards)
`emacs` *file1*	Use emacs to edit *file1*. Follow the tutorial. Use Control-X, Control-S to save, and Control-X, Control-C to quit
`vim` *file1*	Run vim to edit *file1*. Follow the tutorial. Use `:q!` to quit without saving, and `ZZ` to save and quit.

Other text editors include Sublime, Atom, Gedit, and Kate, and XEmacs.

Locating Files

`locate` *name*	List files with *name* in their name. This uses daily scans of the system.
`find` *path*	Displays files in *path*
`-name` *name1*	Filter by name *name1*
`-iname` *name1*	Filter by name *name1*, ignoring case
`-maxdepth` *num*	Only search *num* levels of folders

There are many more options to `find`, as well as ways to combine them. See `man find` for more possibilities.

Grep and Regular Expressions

`grep` *regex file1*	Displays lines from *file1* matching *regex*.
`-v`	Only show lines that don't match the regex.
`-n`	Display line numbers of matching lines.
`-i`	Ignore case (i.e., upper or lowercase).

`grep` can also be used without a file to use the standard input.

Regular Expressions can be made up of normal characters, but several symbols have special meaning: `*.^[]()$\|?{}+`.

`*` signifies that the preceding letter may be repeated zero or more times. For example, `abc*d` would match `abccccd`, `abcd`, or `abd`.

`.` signifies any character. For example, `shar.` would match `shark` or `sharp`.

`[ ]` signifies any of a set of characters can be used. For example, `t[aeh]e` would match `tee`, but not `toe`.

Regular Expressions are a deep and powerful subject, defining a restricted class of languages. O'Reily's Mastering Regular Expressions book is an excellent book on the subject.

Advanced Commands

`awk '{print$`*n*`}'` *file1*	Prints the *n*-th column from *file1*.
`sed 's/`*text1*`/`*text2*`/'` *file1*	Replaces *text1* with *text2* in *file1*.

Both of these commands can be used without a file to use the standard input.

`awk` and `sed` are full programming languages, mainly for generating reports and processing text respectively.

Tar Files

`tar` combines multiple files together into a `.tar` archive. Normally this is compressed into a `.tar.gz` or `.tar.bz2` archive. `tar`'s options include:

x Extract files from an archive

c Create an archive

v List files being extracted or added

z Enable Gzip compression (used for `.tar.gz` or `.tgz` files)

j Enable Bzip2 compression (used for `.tar.bz2` or `.tbz2` files)

f Specifies the name of the archive created/extracted

For example:

`tar zxvf` *file1.tar.gz*	Extract a Gzip-compressed tar file
`tar jcvf` *file1.tar.bz2 files*	Create a tar file with *files* in it

Remote Access

`ssh` *username@server*	Remotely login via SSH protocol.
`ssh -Y` *username@server*	As above, but show GUI applications

Getting Help

`man` *command*	Open the manual (man) page for *command*.
`info` *command*	Open the info documentation for *command*.

Almost every command has a man page. Info has more complex documentation, with multiple cross referenced pages.