Linux Command Cheat Sheet

For those people will little prior experience to the Linux command line "bash" shell, this quick reference provides a guide to the most useful and essential commands. The following table describes these commands. For more information, run `man command_name` or in some cases `command_name -h` for more information about `command_name`.

Name	Example	Description
`ls`	`ls /bin`	"List" the names of the files in a directory or an individual file.
`ls -l`	`ls -l /bin`	Show a "long listing" of the contents of a directory or file; that is, show more information about each directory and file.
`cd`	`cd new_dir`	"Change directory" to a new working directory, `new_dir`. You can use `~` as a shorthand for your "home" directory.
`pwd`	`pwd`	"Print working directory" to show where you are!
`cat`	`cat /tmp/file.log`	"Concatenate" one or more files to the screen. That is, dump the contents to the screen.
`less`	`less /tmp/file.log`	Show the contents a page at a time. Use space to show the next page and `q` to quit.
`rm`	`rm /tmp/file.log`	"Remove" the file. **Warning:** there is no "undo"!
`rm -rf`	`rm -rf directory`	"Remove" `directory` and its contents, recursively. **Warning:** there is no "undo"!

`mkdir`	`mkdir ~/test`	"Make directory". In this example, a directory named `test` inside your home directory.
`man`	`man ls`	Show the "manual" page for the specified command.
`grep`	`grep ERROR /tmp/file.log`	Show all lines in the file(s) that match the second argument. In this example, show all lines in the log file that have the string `ERROR`.
`cp`	`cp file1 file2`	Copy `file1`, creating a new `file2`. **Warning**: Deletes `file2` first if it already exists.
`cp -r`	`cp -r dir1 dir2`	Recursively copy `directory1` to `directory2`. It `directory2` already exists, adds any new files to, deleting any previous copies.
`find`	`find home -name 'toss*'`	Find all files and directories under `home` recursively whose names begin with `toss`.

You also have several options for using files in commands:

Example	Description
`grep ERROR /tmp/file.log > /tmp/error.log`	Find all lines with `ERROR` and write the output to `/tmp/error.log`
`mycommand < /tmp/file.log > /tmp/output`	Run `mycommand` and when it reads the *standard input* (what you would normall type yourself), use the contents of the file `/tmp/file.log` instead. Write the resulting output to `/tmp/output`.
`mycommand < /tmp/file.log >& /tmp/output`	Like the previous command, but also write any *error* output from `mycommand` to the same output file. Otherwise, the error output will be written to the screen.
`ls -l /bin | grep ca`	List the files in `bin`, then "pipe" the output into the `grep` command and show only the lines that contain the string `ca`.
`ls -l /bin |& grep ca`	Like the previous command, but also send any *error* output from `ls -l /bin` to the `grep ca` command.