Using The Terminal

- adaptated from http://help.ubuntu.com/community/UsingTheTerminal

Why use the command-line?

"Under Linux there are GUIs (graphical user interfaces), where you can point and click and drag, and hopefully get work done without first reading lots of documentation. The traditional Unix environment is a CLI (command line interface), where you type commands to tell the computer what to do. That is faster and more powerful, but requires finding out what the commands are."
-- from ***man intro(1)***

There are many varieties of Linux, but almost all of them use similar commands that can be entered from a command-line interface terminal.

There are also many graphical user interfaces (GUIs), but each of them works differently and there is little standardization between them. Experienced users who work with many different Linux distributions therefore find it easier to learn commands that can be used in all varieties of Ubuntu and, indeed, in other Linux distributions as well.

For the novice, commands-line interface commands can appear daunting:

```
sudo gobbledegook blah_blah -w -t -h --long-switch
aWkward/ComBinationOf/mixedCase/underscores_strokes/and.dots
```

However, it is important to note that even experienced users often cut and paste commands (from a guide or manual) into the command-line terminal; they do not memorize them.

It is important, of course, to know how to use the command-line terminal - and anyone who can manage typing, backspacing, and cutting and pasting can manage the command-line terminal (it is not more difficult than that).

This page will outline a few crafty shortcuts which can make using a command-line interface easier.

Starting a Terminal

Applications menu -> **Accessories** -> **Terminal**.

Keyboard Shortcut: Ctrl + Alt + T

File & Directory Commands

- The tilde (~) symbol stands for your home directory. If you are *user*, then the tilde (~) stands for /home/*user*
- **pwd**: The **pwd** command will allow you to know in which directory you're located (**pwd** stands for "print working directory"). Example: **"pwd"** in the Desktop directory will show "~/Desktop". Note that some Terminals also displays this information in the title bar of its window.
- **ls**: The **ls** command will show you ('list') the files in your current directory. Used with certain options, you can see sizes of files, when files were made, and permissions of files. Example: **"ls ~"** will show you the files that are in your home directory.

- **cd**: The **cd** command will allow you to change directories. When you open a terminal you will be in your home directory. To move around the file system you will use **cd**. Examples:
    - To navigate into the root directory, use **"cd /"**
    - To navigate to your home directory, use **"cd"** or **"cd ~"**
    - To navigate up one directory level, use **"cd .."**
    - To navigate to the previous directory (or back), use **"cd -"**
    - To navigate through multiple levels of directory at once, specify the full directory path that you want to go to. For example, use, **"cd /var/www"** to go directly to the /www subdirectory of /var/. As another example, **"cd ~/Desktop"** will move you to the Desktop subdirectory inside your home directory.
- **cp**: The **cp** command will make a copy of a file for you. Example: **"cp file foo"** will make an exact copy of "file" and name it "foo", but the file "file" will still be there. If you are copying a directory, you must use **"cp -r directory foo"** (copy recursively).
- **mv**: The **mv** command will move a file to a different location or will rename a file. Examples are as follows: **"mv file foo"** will rename the file "file" to "foo". **"mv foo ~/Desktop"** will move the file "foo" to your Desktop directory but will not rename it. You must specify a new file name to rename a file.
    - To save on typing, you can substitute '~' in place of the home directory.
- **rm**: Use this command to remove or delete a file in your directory.
- **rmdir**: The **rmdir** command will delete an *empty* directory. To delete a directory and all of its contents recursively, use **rm -r** instead.
- **mkdir**: The **mkdir** command will allow you to create directories. Example: **"mkdir music"** will create a directory called "music".
- **man**: The **man** command is used to show you the manual of other commands. Try **"man man"** to get the man page for **man** itself. See the "**Man** & Getting Help" section down the page for more information.

Running a File Within a Directory

So you've decided to run a file using the command-line (a program you just compiled, for example)? Well... there's a command for that too!

- ***./****filename*

After navigating to the file's parent directory, this command will enable any Ubuntu user to parse files compiled via g++ or any other programming language.

System Information Commands

- **free**: The **free** command displays the amount of free and used memory in the system. **"free -m"** will give the information using megabytes, which is probably most useful for current computers.
- **top**: The **top** command displays information on your Linux system, running processes and system resources, including CPU, RAM & swap usage and total number of tasks being run. To exit **top**, press **"q"**.
- **uname -a**: The **uname** command with the **-a** option prints all system information, including machine name, kernel name & version, and a few other details.

Options

The default behavior for a command may usually be modified by adding a ***--option*** to the command. The **ls** command for example has an **-s** option so that **"ls -s"** will include file sizes in the listing. There is also a **-h** option to get those sizes in a "human readable" format.

Options can be grouped in clusters so **"ls -sh"** is exactly the same command as **"ls -s -h"**. Most options have a long version, prefixed with two dashes instead of one, so even **"ls --size --human-readable"** is the same command.

"Man" and getting help

man *command*, **info *command*** and ***command* --help** are the most important tools at the command line.

Nearly every command and application in Linux will have a man (manual) file, so finding them is as simple as typing **"man "command""** to bring up a longer manual entry for the specified command. For example, **"man mv"** will bring up the **mv** (Move) manual.

Move up and down the man file with the arrow keys, and quit back to the command prompt with **"q"**.

"man man" will bring up the manual entry for the **man** command, which is a good place to start!

"man intro" is especially useful - it displays the "Introduction to user commands" which is a well-written, fairly brief introduction to the Linux command line.

There are also **info** pages, which are generally more in-depth than **man** pages. Try **"info info"** for the introduction to info pages.

Some software developers prefer **info** to **man**, so if you find a very widely used command or app that doesn't have a **man** page, it's worth checking for an **info** page.

Virtually all commands understand the **-h** (or **--help**) option which will produce a short usage description of the command and it's options, then exit back to the command prompt. Try **"man -h"** or **"man --help"** to see this in action.

*Caveat: It's possible (but rare) that a program doesn't understand the -h option to mean help. For this reason, check for a **man** or **info** page first, and try the long option --help before -h.*

Pasting in commands

Often, you will be referred to instructions that require commands to be pasted into the terminal. You might be wondering why the text you've copied from a web page using **ctrl+C** won't paste in with **ctrl+V**. Surely you don't have to type in all those nasty commands and filenames? Relax. **ctrl+shift+V** pastes into a terminal; you can also do Middle Button Click on your mouse (both buttons simultaneously on a two-button mouse) or Right Click and select *Paste* from the menu.

Save on typing

Up Arrow or **ctrl+p**	Scrolls through the commands you've entered previously.
Down	Takes you back to a more recent command.

Arrow or **ctrl+n**	
Enter	When you have the command you want.
tab	A very useful feature. It auto-completes any commands or filenames, if there's only one option, or else gives you a list of options.
ctrl+r	Searches for commands you've already typed. When you have entered a very long, complex command and need to repeat it, using this key combination and then typing a portion of the command will search through your command history. When you find it, simply press **Enter**.
history	The **history** command shows a very long list of commands that you have typed. Each command is displayed next to a number. You can type **!x** to execute a previously typed command from the list (replace the X with a number). If you **history** output is too long, then use **history | less** for a scrollable list.

Change the text

The mouse won't work. Use the Left/Right arrow keys to move around the line.

When the cursor is where you want it in the line, typing *inserts* text - ie it doesn't overtype what's already there.

ctrl+a or **Home**	Moves the cursor to the *start* of a line.
ctrl+e or **End**	Moves the cursor to the *end* of a line.
esc+b	Moves to the **b**eginning of the previous or current word.
ctrl+k	Deletes from the current cursor position to the end of the line.
ctrl+u	Deletes from the start of the line to the current cursor position.
ctrl+w	Deletes the word before the cursor.