Introduction to the Linux Command Line

Launching a “terminal emulator”

There are many to choose from in addition to those that come with your Linux distribution.

See for example http://www.slant.co/topics/794/~linux-terminal-emulators

Typing at the command line

Everything is a command!

```
command
```

Things to try:

`ls`	list files in current directory
`lspci`	list PCI devices
`ifconfig`	show network configuration

Editing commands

Left/Right cursor and Home/End keys are supported by most terminals for editing commands.

You don't have to move to the end of the line after editing before pressing the return key.

Copy and paste are usually available but NOT using the regular GUI controls keys because these control keys had a meaning in terminals long before their GUI use. eg Ctrl-C means *“interrupt/stop the currently running program”*

What commands can I use?

Shell “builtin” commands and external commands

```
type command
type -a command
```

The shell searches for external commands in folders specified in the search path. To find the exact location of a command program, use

```
which command
```

To show the current search path *(which is configured by script settings when you log in)*

```
echo $PATH
```

To explicitly run a command using a file in the current directory

```
./command
```

Format of commands

`command`	`ls`
`command -option(s)`	`ls -al`
`command --long-option(s)`	`ls --size`
`command argument(s)`	`ls *.pdf`
`command -option(s) argument(s)`	`ls -al *.pdf`

Help with builtin commands

```
help
help builtin-command
```

eg `help echo`

Help with external commands

```
man command
```

Many external programs will also show brief help if you use one of the *"help"* options

```
command --help
```

or

```
command -h
```

Command history

Accessible with most shells and terminals using Up/Down cursor keys

Search backwards through command with Ctrl-R and start typing previous command

Look through your command history (if using the Bash shell) in the file **`.bash_history`** in your home folder

Paging output from commands

We can "pipe" the output of any command that may take multiple screens through a "pager" program such as **`more`**.

The pipe symbol is the vertical bar **|** *(to the left of the Z key, shifted, on a UK keyboard)*

```
help | more
```

Working with Folders (directories)

print working directory	**`pwd`**
change directory	**`cd`**
create directory	**`mkdir`**
remove directory	**`rmdir`**

Working with files

List files in current folder	ls
Detailed file list	ls -l
Copy a file	**`cp`** ***`sourceFilename destinationFilename`***
	or **`cp`** ***`sourceFilename destinationFoldername`***
Move a file	**`mv`** ***`sourceFilename newFoldername`***
Rename a file	**`mv`** ***`sourceFilename newFilename`***
Create an empty file	**`touch`** ***`filename`***
	`touch` is a program used to modify file timestamps but it can also be used like this to create an empty file.

Shortcut symbols for folders

home directory	**`~`**
parent directory	**`..`**
current directory	**`.`**
root directory	**`/`**

Redirecting screen output to a file

```
command > filename
```