

Directories and files

Concepts

- Files are organised in a directory tree/hierarchy
- Everything is a file (e.g. keyboard, printers, ...)
- Each process has access to the files *stdin* (input), *stdout* (buffered output), *stderr* (unbuffered output)
- Each process operates in a *working directory*
- Each user has a *home directory*

Paths

Path = Identifier for the location of file/directory

- Paths consists of a parent directory list + file/directory
- Files and directories are separated by a '/'
- Directory paths may contain a trailing '/'

Absolute path = Full location (first character = '/')

Relative path = Relative location (first character \neq '/')

.	Path to the directory itself
..	Path to the parent directory
/usr/bin/ls	Example for an absolute file path
/home/foo/	Example for an absolute directory path
./a.out	Example for a relative file path

File system hierarchy

/	Root directory
/bin	Essential command executables
/dev	Device files
/etc	System-wide configuration files
/opt	Manually added software
/sbin	Essential administrative executables
/tmp	Temporary files
/usr	System resources for users
/usr/bin	Command executables
/usr/local	Site-local data
/usr/sbin	Administrative executables
/var	Variable files

man hier (or **man** file-hierarchy on recent Linux distributions) to get a more detailed overview

Terminal (emulator)

Text terminal = Computer interface for text entry/display

Terminal emulator = Application that emulates a *text terminal* in a graphical environment

Examples for terminal emulators: xterm, urxvt, guake

Opening a terminal

Unity/GNOME	Ctrl + Alt + T
Mac OS	Cmd + [] → "terminal" → ↵
Bash on Windows	Win + R → "bash" → ↵

Shell

Unix shell = User interface that accepts commands to operate a computer

man intro to get an introduction into basic shell usage

Examples for shell programs: sh, bash, zsh, fish, ksh

Prompt

Prompt = Text sequence that precedes each line that prompts the user to enter a command

[foo@bar /var/www]\$ Example prompt in bash
⇒ User foo is operating in the *working directory* /var/www at the computer with the *host name* bar

Line editing

Ctrl + A / E	Go to beginning/end of the line
Ctrl + U / K	Erase to beginning/end of the line
Alt + B / F	Go one word forward/backward in line
Ctrl + W	Erase to beginning of current word
Ctrl + C	Cancel the current command line

Metacharacters

The following characters have special meaning and sometimes they can't be used directly as arguments/words:

| & ; < > () \$ ` \" ' * ? [] # ~ = % ! { } [] ← ↵

Their special meaning can be disabled:

\	Preserves the literal value of the following character
' '	Preserves the literal values of enquoted characters
" "	Like ' ' but characters ` \$ \ retain their meaning

Expansions

~	Home directory of the current user
*	Matches any character sequence
?	Matches a single character
[...]	Matches any one of the enclosed characters
\${var}	Value of the environment variable <i>var</i>
\$(cmd)	Output of <i>cmd</i>
\$((expr))	Result of the mathematical expression <i>expr</i>

Shell utilities

apropos text	Searches the manual pages for <i>text</i>
cat file	Prints the contents of <i>file</i>
cd dir	Changes the <i>working directory</i> to <i>dir</i>
chmod prm file	Changes permissions of <i>file</i> to <i>prm</i>
cp src dst	Copies the file/directory <i>src</i> to <i>dst</i>
echo text	Prints <i>text</i>
file file	Determines the file type of <i>file</i>
find dir expr	Finds files in <i>dir</i> that match <i>expr</i>
grep expr file	Searches for pattern <i>expr</i> in <i>file</i>
ls dir	List the entries in the directory <i>dir</i>
man cmd	Displays the manual for <i>cmd</i>
mkdir dir	Creates the directory <i>dir</i>
mv src dst	Moves/renames <i>src</i> to <i>dst</i>
pwd	Prints the current <i>working directory</i>
rm file	Removes the file <i>file</i>
sort	Sorts lines of text from input
touch file	Creates the empty file <i>file</i>

Input output redirection

cmd1 cmd2	Runs <i>cmd1</i> and <i>cmd2</i> and redirects the output of <i>cmd1</i> to the input of <i>cmd2</i>
cmd > file	Runs <i>cmd</i> and redirects output to <i>file</i> , content of <i>file</i> is overwritten
cmd >> file	Like >> but appends output to <i>file</i>
cmd < file	Runs <i>cmd</i> and redirects <i>file</i> to its input
cmd <<< text	Runs <i>cmd</i> with input <i>text</i>

Job control

Job = Shell command and its associated process(es)

- Each job has a job id and corresponding process ids
- Jobs can run in the foreground or in the background
- The execution of a job can be temporarily suspended

cmd &	Starts <i>cmd</i> as background job (id is printed)
fg %job	Puts the job <i>job</i> in foreground
bg %job	Continues suspended job <i>job</i> in background
jobs	Prints job numbers of jobs in current terminal
kill pid	Terminates a process with the process id <i>pid</i>
ps	Prints PIDs of processes in current terminal
Ctrl + S / Q	Suspends/resumes active job
Ctrl + Z	Puts active job to background and suspends it
Ctrl + C	Aborts the active job (most of the times)
Ctrl + D	Ends an EOF character