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 example for an absolute directory path /home/foo/ example for a relative file path ./a.out

File system hierarchy

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

man hier (or man file-hierarchy on recent Linux Expansions 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, quake

Opening a terminal

Unity/GNOME Ctrl + Alt + \rightarrow "terminal" \rightarrow [\downarrow] Mac OS $[Win] + [R] \rightarrow "bash" \rightarrow []$ Bash on Windows

Shell

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

man intro to get an introduction into basic shell usage

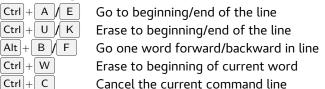
Examples for shell programs: sh, bash, zsh, fish, ksh

Prompt

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

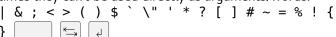
[foo@bar /var/www]\$ example prompt in bash ⇒ user foo is operating in the working directory /var/www at the computer with the host name bar

Line editing



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

| • | |
|---------------------|--|
| ~ | home directory of the current user |
| * | matches any character sequence |
| ? | matches a single character |
| [] | matches a character enclosed by the braces |
| \${ <i>var</i> } | value of the environment variable \emph{var} |
| \$(<i>cmd</i>) | output of <i>cmd</i> |
| \$((<i>expr</i>)) | result of the mathematical expression expr |
| | |

Shell utilities

| apropos text | searches the manual pages for text |
|---------------------------|--|
| cat file | prints the contents of <i>file</i> |
| cd dir | changes the working directory to di |
| <pre>chmod prm file</pre> | changes permissions of file to pri |
| cp src dst | copies the file/directory src to dst |
| 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 dir |
| man cmd | displays the manual for <i>cmd</i> |
| mkdir dir | creates the directory dir |
| mv src dst | moves/renames <i>src</i> to <i>dst</i> |
| pwd | prints the current working directory |
| rm file | removes the file <i>file</i> |
| sort | sorts lines of text from input |
| touch file | creates the empty file file |
| | |

Input output redirection

| input output realisection | | |
|----------------------------------|---|--|
| cmd1 cmd2 | runs cmd1 and cmd2 and redirects the | |
| | output of <i>cmd1</i> to the input of <i>cmd2</i> | |
| <pre>cmd > file</pre> | runs <i>cmd</i> and redirects output to <i>file</i> , | |
| | content of <i>file</i> is overwritten | |
| <pre>cmd >> file</pre> | like >> but appends output to file | |
| <pre>cmd < file</pre> | runs cmd and redirects file to its input | |
| <pre>cmd <<< text</pre> | runs <i>cmd</i> with input <i>text</i> | |

Job control

Ctrl + D

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

| 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 job 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 pid | |
| 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) | |

aborts the active job (most of the times)

sends an EOF character