Unix Shell

1 Paths

Files are organized in a tree structure. To refer to a file, you can use either absolute paths, or relative paths. An *absolute path* starts with /. If a path is *relative*, it is taken from the current directory. In a path, .. refers to the parent directory, ˜ refers to your personal directory (also called $HOME).

2 Some Important Shell Commands

Here are some important shell commands; [*option*] denotes an optional parameter:

- man *cmd*: display the **man**ual page associated to *cmd*. The manual also documents OCaml libraries: 'man *Module*' will show the documentation for *Module*.
- cd [*dir*]: **c**hange the current **d**irectory to *dir*. Without parameter, go to the home directory.
- chmod *mode file*: **ch**ange file **mod**e bits, i.e. change file access permissions. Typical modes: '700' is user exec, read and write, no rights for anyone else; '644' is user read and write, reading rights for everyone else.
- ls [*file*]: **l**i**s**t the contents of *file*. Without parameter, display the contents of the current directory.
- cp [−r] *src dest*: **c**o**p**y *src* into *dest*. With '−r', copy a directory recursively.
- mv *src dest*: **mov**e a file or directory from *src* to *dest*. This can be used to rename a file/directory.
- rm [−r] *file*: **r**e**m**ove *file*. With '−r', remove a directory. Use with caution.
- find *dir* −name *pat*: find files whose name matches *pat* in *dir*. Many other options exist for this command; another similar command is 'locate'.
- mkdir *dir*: **m**a**k**e a **dir**ectory *dir*.
- ps [a][x]: list the current **p**rocesse**s** run by the user in a terminal. With 'a', also shows the processes started by other users. With 'x', also shows the processes started outside a terminal. Check out 'jobs' for background processes.
- pwd: **p**rint the current **w**orking **d**irectory.
- ssh [−X] dptinfoxx.dptinfo.ens−cachan.fr: opens a **s**ecure **sh**ell on *dptinfoxx* in the computer lab. With '−X', display graphical windows on your local machine.

- *cmd* [*params*]&: launches *cmd* with arguments *params* in the background, giving the prompt back (to execute other commands).

3 Signals

If a program is stuck, you can send it a signal to terminate it. 'Ctrl+C' will usually terminate it (if it is still able to notice the signal!). 'Ctrl+Z' pauses it and give you back the prompt shell with a job number; fg [%*job*] will resume it, bg [%*job*] will resume it in the background—no arguments meaning the last paused process, and kill −KILL %*job* will terminate the program.

4 Development Tools

Here are a few tools you will probably need sooner or later:

A text editor Recommended: emacs, vi, gedit. In emacs, check the tutorial (C−h t) and tuareg−mode for OCaml editing.

gcc The GNU C compiler. Compiles C files into object code ('−c' option), assembly code ('−S' option), or executable code (default). Use the '−o *file*' option to select the output file name. Check out the '−Wall −ansi −g −pg −m32 −DNDEBUG' options. The gcc command is actually a front-end to other tools, notably cpp, as, and ld.

make A Makefile is a file that describes how some other files can be build using shell commands. The 'make *file*' command runs these shell commands according to the Makefile to build *file*.

The Makefile itself is written in an obscure language, where a typical entry looks like

```
%.o: %.c
	$(CC) $(CFLAGS) -c $< -o $@
```

meaning that *file* .o depends on *file* .c (thus % acts as a sort of wildcard), and can be built by executing the command '$(CC) $(CFLAGS) −c *file*.c −o *file*.o' where '$(CC)' and '$(CFLAGS)' are environment variables selecting the compiler (e.g. gcc) and its compilation options (e.g. −Wall −ansi −g −m32). An OCaml-specific variant is omake.

gprof A profiler. Tells you how long a C program spends in each function.

valgrind A profiling and debugging tool (Linux only). Quite handy for memory leaks.

ddd A graphical debugger.

svn A version-control system. Allows to collaborate and manage conflicting updates to files. Other similar tools: hg, darcs, . . .

doxygen Generate documentation from annotations in source files. An OCaml-specific equivalent is ocamldoc.