Techniques et outils d'ingénierie logicielle

Cours 1 — bases

John Gliksberg — Bull

Objet du cours

- Maîtriser un *shell* et ses automatisations
- Avoir un environnement de développement productif
- Savoir compiler un programme
- Appréhender le cycle de vie d'un programme
- Gestion de code source
- Déboguer un programme
- Profiler un programme

Contents

Règles du jeu

Matière **pratique**
TP **notés**

Matière **pratique**
PC **à chaque séance**

Environnement type **UNIX**
(Linux, BSD, Max, WSL)

Accès facile **à tout moment** au terminal

Participez

mais lisez les manuels

Pas de magie, explications **spécifiques**

Testez ce que vous dites

Testez ce que vous pensez

Montrez que vous avez compris

« Pas le temps de corriger ma config » **interdit**

La peur n’évite pas le danger

La peur est la petite mort

Les outils de l'ingénieur·e informaticien·ne

Le terminal, l'éditeur de texte, le VCS
le compilateur, le débogueur, le profileur

L’invite de commande, les commandes, les raccourcis

Le système d’exploitation, l’environnement de bureau

Le PC, le clavier

Les mains, la dextérité

La tête, le sens critique, la curiosité, la parole
la lecture, la mémoire, la capacité d'abstraction

Les manuels
(et StackOverflow, Wikipédia, LLM)

Le système d'exploitation

- Un noyau, des drivers, des systèmes de fichier
- un environnement graphique
  - KDE, Gnome, Sway, XFCE
- des programmes inclus
- un gestionnaire de paquets
- un ecosystème
- des choix, par des gens

- C’est votre maison, prenez-en soin
- Modifiez le, testez en plusieurs
- Soyez prêt·e à perdre tous les fichiers

Tous les fichiers sont dans un grand arbre de dossiers

qui commence à la racine « / »

Il y a des chemins absolus `/home/`*`john`*`/.config/nvim/init.lua`

et des chemins relatifs `Téléchargements/meme.gif`

`../../img/../src`

`./.zshrc`

Il peut y avoir ~ n'importe quel caractère

Les *dotfiles* sont « cachés »

L'invite de commande

Il vous invite (*prompt*) à écrire des commandes

```
john@superbecane:~$
```

Les commandes sont une list de chaînes de caractères

Les commandes sont structurées selon des conventions

```
john@superbecane:~$ find src -name '*.h'
```

Les commandes sont sensibles à la casse

```
john@superbecane:~$ Find src -name ‹ *.h ›
-bash: Find: command not found
john@superbecane:~$ cd téléchargements
-bash: cd: téléchargements: No such file or directory
```

Les commandes peuvent fournir des interfaces utilisateur :

- en « ligne de commande » (CLI)
- « textuelles » (TUI)
- « graphiques » (GUI)
- plusieurs à la fois ?
- autres interfaces utilisateurs ?
- autres interfaces ?

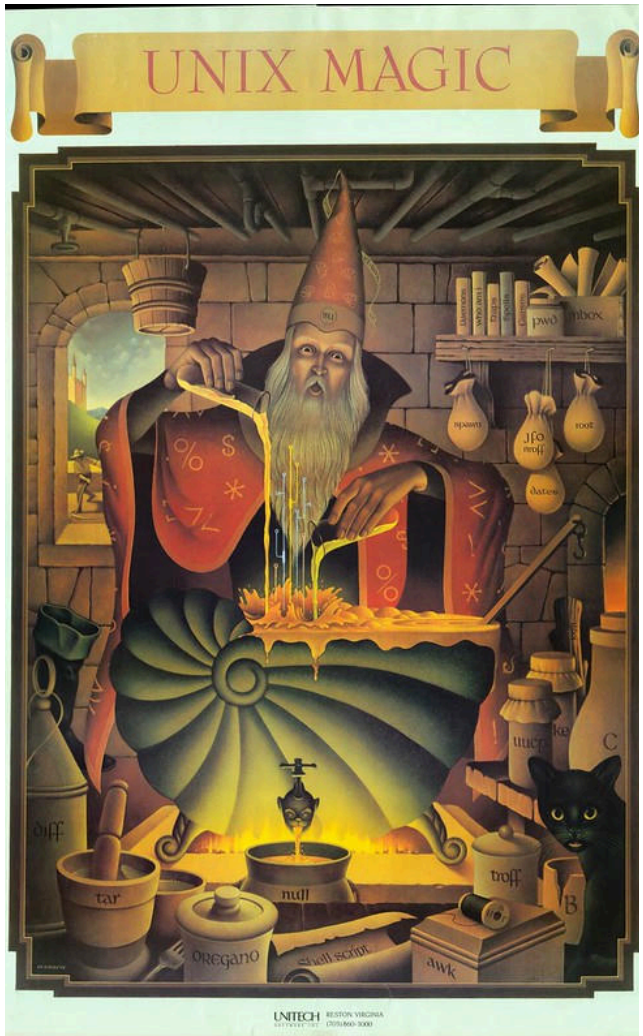
Le grand bazaar

ls mv cp rm ln mkdir chmod chown touch wc sort cat tac tee pwd test du df dd [time date cal rename which whereis file more less sed grep awk tr head tail cut find tree mc ed vi em nano python perl lua go cargo m4 lex git jq gpg xargs dot tput su sudo sudoedit visudo w whoami ssh scp ftp tmux screen nc strace ping rsync ip ps top htop btop vmstat iostat free lsof nice dmesg lsusb lsblk fdisk mount mkfs cc as ar ld nm ldd gcc clang make cmake automake gdb valgrind perf lp tar zip curl wget zcat convert ffmpeg man info apt dpkg dnf yum rpm pacman emerge npm patch diff kill mkfifo mktemp fsck swapon swapoff watch cron reboot shutdown sqlite systemctl usermod timeout uptime chroot lscpu lstopo rg fd fzf bat nu neofetch

---

help cd echo printf test [[[((time if case while for break continue return exit kill sleep set export unset alias function source pushd pwd read shift trap & > >> < << $ $$ $? $0 $(${ !! !$ { (&& || fg bg disown jobs clear reset ulimit


UNIX VIEWS
UNITECH RESTON, VIRGINIA


UNIX MAGIC
tar
null
troff
oregano
awk
UNITECH RESTON, VIRGINIA


UNIX FEUDS
UNITECH

Commandes de base

pwd cd mkdir ls cat touch du df grep

À L’AIDE

Le manuel

```
$ man [<section>] <page>
```

```
man ls
man pwd
man man
man man-pages
man firefox
man 1 ls
man 7 signal
```

See also **ls**(**1**), **signal**(**7**).

Le manuel s'ouvre avec **less**(**1**) (TUI) par défaut

- Flèches, PageDown/PageUp, `d`/`u`, `g`/`G`
- `q`
- `h`
- `/ -d`

Les pages de manuel sont structurées

```
STANDARDS
       POSIX.1, except as noted.

NOTES
       For a discussion of async-signal-safe functions, see signal-safety(7).

       The /proc/pid/task/tid/status file contains various fields that show
       the signals that a thread is blocking (SigBlk), catching (SigCgt), or
       ignoring (SigIgn).  (The set of signals that are caught or ignored will
       be the same across all threads in a process.)  Other fields show the
       set of pending signals that are directed to the thread (SigPnd) as well
       as the set of pending signals that are directed to the process as a
       whole (ShdPnd).  The corresponding fields in /proc/pid/status show the
       information for the main thread.  See proc(5) for further details.

BUGS
       There are six signals that can be delivered as a consequence of a
       hardware exception: SIGBUS, SIGEMT, SIGFPE, SIGILL, SIGSEGV, and
       SIGTRAP.  Which of these signals is delivered, for any given hardware
       exception, is not documented and does not always make sense.

       For example, an invalid memory access that causes delivery of SIGSEGV
       on one CPU architecture may cause delivery of SIGBUS on another
       architecture, or vice versa.

       For another example, using the x86 int instruction with a forbidden
       argument (any number other than 3 or 128) causes delivery of SIGSEGV,
       even though SIGILL would make more sense, because of how the CPU
       reports the forbidden operation to the kernel.

SEE ALSO
       kill(1), clone(2), getrlimit(2), kill(2), pidfd_send_signal(2),
       restart_syscall(2), rt_sigqueueinfo(2), setitimer(2), setrlimit(2),
       sgetmask(2), sigaction(2), sigaltstack(2), signal(2), signalfd(2),
       sigpending(2), sigprocmask(2), sigreturn(2), sigsuspend(2),
       sigwaitinfo(2), abort(3), bsd_signal(3), killpg(3), longjmp(3),
       pthread_sigqueue(3), raise(3), sigqueue(3), sigset(3), sigsetops(3),
       sigvec(3), sigwait(3), strsignal(3), swapcontext(3), sysv_signal(3),
       core(5), proc(5), nptl(7), pthreads(7), sigevent(3type)

Linux man-pages 6.15             2025-05-17                        signal(7)
```

- manpages-fr
- manpages-fr-dev
- manpages-fr-extra

```
cat --help
```

Au secours

- CTRL + C → SIGINT
- CTRL + S → TTY XOFF
- CTRL + Q → TTY XON

Automatisations dans l'invite de commande

```
test -f README.md
echo $?
```

```
if ! test -f README.md
then
  echo "Missing read me file" >&2
fi
```

```
read_me () {
  if ! test -f README.md
  then
    echo "Missing read me file" >&2
    return 1
  fi
  less README.md
}
```

```
n=1
while valgrind --error-exitcode=123 test_injection_stress
do
  ((++n))
done
echo $n
```

```
n=1 ; while valgrind --error-exitcode=123 test_injection_stress ; do
((++n)) ; done ; echo $n
```