Introduction à l'environnement UNIX

R&I 1

Antoine Rollet

Bordeaux INP - Enseirb-Matmeca

Emacs

Edition de texte

Documentation

Lien vers le cours :

https://rollet.zzz.bordeaux-inp.fr/

Un peu de lecture :

`https://rollet.zzz.bordeaux-inp.fr/IntroUnix/documentation/refcard-emacs.pdf`

Introduction

`emacs` :

- est un logiciel permettant de saisir et de modifier du texte
- n'est pas un traitement de texte
- permet de manipuler des fichiers contenant du texte au sens large
- possède les fonctions classiques : déplacement dans les fichiers, recherche, positionnement, copier/couper/coller, ...
- s'interface avec le système : parcours de répertoires, lecture de courrier, compilation, etc
- possède de nombreuse bibliothèques et modes spécialisés
- utilise un mode d'édition adapté au type de fichier ouvert (automatique ou manuel)
- environnement extensible et configurable ; programmation en `lisp`
- est un logiciel libre
- peut s'utiliser sans fenêtre (mode `no-window`), sans souris

Alternatives (?) : `nano, pico, vi, vim, VSCode, gedit, ...`

Principales façons de lancer emacs en ligne de commande

- `emacs &` → lance emacs seul en tâche de fond
- `emacs file &` → emacs ouvre le fichier (si le fichier n'existe pas il le crée)
- `emacs dir/ &` → emacs ouvre le répertoire et montre son contenu.

Lexique (1)

- **buffer** (tampon) : zone mémoire dans laquelle emacs copie le fichier
- **fenêtre** : écran dans lequel emacs affiche le fichier. Emacs est capable de diviser l'écran en plusieurs fenêtres
- chaque fenêtre est munie d'une **ligne de mode** : position du curseur, nom du buffer, buffer modifié, ...
- en bas de l'écran, se trouve le **mini-buffer** utilisé pour effectuer des opérations d'entrée/sortie, affiche des messages, ...
- le cycle d'édition classique : ouverture d'un fichier dans un buffer, édition (modification) du fichier, sauvegarde (il faut sauvegarder le plus souvent possible)

Dans la suite :

- `C` : Contrôle, touche 'CTRL'
- `M` : Méta, touche 'ESC' ou ALT
- `C-x` : Appuyer sur 'CTRL' et 'x' simultanément
- `M-x` : Appuyer sur 'ALT' et 'x' simultanément, ou 'ESC' puis 'x'

Lexique (2)

- **clé** : suite de caractères interprétée comme un tout. Certains caractères sont des **préfixes** et ne constituent pas des clés : `C-c, C-x, C-h, ESC`, i.e., ils attendent une suite dans la séquence de caractères.
- **commande** : toute action effectuée dans emacs est le résultat d'une commande (même l'insertion de caractère). De nombreuses commandes sont liées à des clés.

  Ex. :
  - `forward-char C-f` ;
  - `backward-char C-b` ;
  - `save-buffers-kill-emacs C-x C-c`
- l'exécution d'une commande se fait via :
  - la clé associée (ex. : `C-f`),
  - le `mini-buffer` en tapant le nom de la commande (ex. `M-x forward-char`)
- chaque commande est définie par une fonction en `lisp`

Remarque

- La complétion est souvent très utile dans le `mini-buffer`
- ⚠ Éviter d'ouvrir deux instances d'`emacs` sur un même fichier !

Exercice : lancer la commande `snake` (et quitter avec le raccourci `C-x k`)

Quelques commandes/clés de base (1)

Commandes sur les fichiers/buffers

- ouvrir un fichier : `C-x C-f`
- enregistrer un fichier : `C-x C-s`
- enregistrer sous : `C-x C-w`

Recherche/remplacement de texte

- recherche incrémentale en avant `C-s` et en arrière `C-r`
- remplacer un texte interactif `M-x query-replace`, `M-%`
- remplacer toute les occurrences `M-x replace-string`
- complétion d'un mot existant dans les buffers ayant le même préfixe `M-/`

Exercice : Chargement d'un fichier

Tous les exercices suivants sont à réaliser sans la souris

- exécuter la commande `find-file` à l'aide de sa clé. Emacs affiche dans le mini-buffer le message : `Find file : ~/`
- essayer `esc esc esc`. Commenter
- relancer `find-file`, mais cette fois sans utiliser le raccourci
- essayer `tab tab`. Commenter.
- Récupérer le fichier `https://rollet.zzz.bordeaux-inp.fr/IntroUnix/ressources/gnu.txt` (en dehors d'emacs), l'enregistrer dans votre répertoire `~/IntroUnix`.
- à l'aide de la complétion, allez chercher le fichier `gnu.txt` et l'ouvrir dans emacs. Quitter emacs, et ouvrir le même fichier avec la commande `emacs ~/IntroUnix/gnu.txt`
- écrire un peu de texte à la suite, et enregistrer le contenu.
- écrire encore un peu de texte, et enregistrer le contenu du buffer à l'adresse `~/IntroUnix/gnu2.txt`
- rechercher toutes les occurrences du mot `GNU`, remplacer toutes les occurrences du mot `public` par le mot `private`.
- à la fin du fichier, taper `Fou` puis `M-/`. Que s'est-il passé ?
- fermer le buffer. Quelle est la commande à exécuter ? Quelle est sa clé ?

Quelques commandes/clés de base (2)

Correction orthographique

- `M-x ispell-change-dictionary` pour modifier le dictionnaire
- `M-x flyspell-mode` analyse à la volée (`M-Tab` pour corriger automatiquement)
- `M-x ispell-buffer` analyse le buffer (correction auto possible via un menu idoine)

Mise en forme

- `M-x auto-fill-mode` ajouter automatiquement un saut de ligne en fin de ligne

Exercice

- Ouvrir à nouveau le fichier `gnu.txt`.
- Le rendre un peu plus "lisible" grâce à la commande `auto-fill-mode`
- Repérer, et si possible corriger tous les fautes d'orthographe, sauvegarder.
- Passer en mode *Analyse à la volée ("flyspell")*
- Rajouter un court paragraphe en anglais avec quelques fautes.

Quelques commandes/clés de base (3) : édition

Quelques commandes d'édition

- sélectionner tout le buffer : `C-x h`
- copier/couper/coller/couper une sélection :
  - `M-w` : copier (vers le kill ring)
  - `C-w` : couper (et placer la sélection dans le kill ring)
  - `C-y` : coller (depuis le kill ring)
  - `C-k` : couper (et placer la sélection dans le kill ring) la (fin de la) ligne
- coller depuis le kill ring : `C-y`, puis `M-y` autant de fois que désiré pour parcourir l'anneau de ce qui a été coupé
- annuler l'action précédente : `C-_`
- `C-u n char` répète n fois char (n=4 par défaut, char peut être une clé)

Exercice (toujours sans la souris)

- Reprendre le fichier `gnu.txt`. Copier le 1er paragraphe à la fin.
- Couper/coller le 2è pargraphe à la fin.
- Coller à nouveau le 1er paragraphe à la fin (sans retourner au début).
- Annuler la dernière action.

Quelques commandes/clés de base (4) : manipulation de fenêtres

Commande de manipulation des fenêtres

- effacer cette fenêtre : `C-x 0`
- effacer toutes les autres fenêtres : `C-x 1`
- split fenêtre horizontal : `C-x 2`
- split fenêtre vertical : `C-x 3`
- déplacer le curseur vers une autre fenêtre : `C-x o`
- changer de buffer : `C-x b`

Exercice (toujours sans la souris)

- Reprendre le fichier `gnu.txt`. Ouvrir une autre fenêtre (split vertical) contenant le fichier `/etc/passwd`. Copier la première ligne du ficher `/etc/passwd` à la fin du fichier `gnu.txt`.
- Revenir à une seule fenêtre contenant le fichier `gnu.txt`. Ouvrir une 2è fenêtre (split horizontal) sur ce même fichier. Se placer à la fin. Ajouter une ligne et sauvegarder. Cette modification a t'elle été prise en compte dans l'autre fenêtre ?

Lexique (3) : liaisons

liaison : on peut modifier dynamiquement une liaison entre une clé et une commande (valables pour tous les buffers) via `global-set-key`.
→ cette modification est temporaire.

- `global-unset-key` pour annuler,
- `C-h k: describe-key` pour vérifier l'existence,
- `C-h b: describe-bindings` pour lister l'ensemble des clés.
- `describe-function` pour obtenir de l'aide sur une commande (notamment sa clé)

Exercice

- Faire en sorte que la commande `C-l` lance automatiquement la commande `goto-line`. Essayer.
- D'après la documentation, à quoi sert la commande `compile`. A partir de quelle version a t'elle été introduite ?
- Associer à la touche `f4` la commande `compile`. Essayer. S'assurer que la clé existe bien (`describe-key`).
- Annuler la dernière association
- Quelle commande permet de passer un paragraphe en majuscule ? Vérifier. Si vous utilisez la clé associée, `emacs` vous demander confirmation. Pourquoi ?

Lexique (4) : macros

macro : suite de commandes clavier définie par l'utilisateur.

- `C-x (` / `C-x )` : début/fin de définition.
  ⚠ attention, il s'agit de la touche "parenthèse" du clavier ici !
- `C-x e` : exécute la dernière macro définie.
- `M-x name-last-kbd-macro` : permet de nommer la macro (définie en dernier).
- `M-x` *unemacro* pour exécuter la macro nommée *unemacro*.

Macros : exercices

- Définir une macro permettant de placer les guillemets anglais (") autour du mot sur lequel est positionné le curseur (utiliser les touches `M-b` et `M-f` pour se déplacer en début et fin de mot)
- Nommer la macro en `insert-quote`. La tester sur le fichier `gnu.txt` avec `M-x insert-quote`.
- Vérifier que la clé `M-"` est libre et la lier globalement à la macro `insert-quote`. Tester.
- Ouvrir un nouveau fichier (`C-x C-f` puis saisir l'adresse d'un fichier inexistant)
- Saisir deux lignes de texte en minuscules, se placer au début du fichier
- Ecrire une macro qui transforme complètement en majuscules le 1er mot de la ligne, et ajoute une majuscule au 3è, puis replace le curseur en début de ligne. Pour cela, on pourra utiliser les commandes `forward-word`, `upcase-word` et `capitalize-word`
- Appliquer cette macro sur quelques lignes du fichier `gnu.txt`

Personnalisation `.emacs`

- si un fichier de configuration `~/.emacs` se trouve à la racine du `home` alors à chaque lancement, le fichier de configuration est chargé et permet de configurer emacs.
- ce fichier contient une liste de commandes écrites en `lisp`.
- ⚠ toujours en faire une sauvegarde avant de le modifier

Quelques exemples :

```
;; a comment
(set-background-color "gray")
(global-set-key (kbd "C-l") 'goto-line) ;; goto line
(global-set-key (kbd "C-c c") 'comment-region) ;; comment region
(global-set-key (kbd "C-c u") 'uncomment-region) ;; uncomment
(global-set-key (kbd "C-c i") 'indent-region) ;; indent
(show-paren-mode t) ;; show parenthesis mode
(setq inhibit-startup-message t) ;; no startup screen
```

Exercice

Pensez à sauvegarder le fichier `.emacs` avant toute modification. Fixer la couleur de fond que vous souhaitez, associer à la clé `C-l` le lancement de la commande `goto-line`. Vérifier que les changements ont bien été pris en compte. (Vous pourrez configurer le reste plus tard).

Gestionnaire de paquetages

emacs est équipé d'un gestionnaire de paquetages très complet

- Pour installer un nouveau paquetage[1] :
  - `M-x package-refresh-contents` ← mettre à jour la base de données des paquetages
  - `M-x package-install packageName` ← installer la paquetage
  - (`M-x package-delete packageName` ← désinstaller un paquetage)

  Exercices :

  - installer le paquetage `2048-game`, le lancer, quitter (`C-x k`)
  - installer le paquetage `doom-themes`, installer le thème de votre choix (commande `load-theme`)
  - installer le paquetage `which-key`, l'activer avec la commande `which-key-mode` et l'essayer (`c-x` puis attendre un peu)

⚠ les paquetages ne seront (a priori) pas disponibles en examen sur machine

1. la marche à suivre précise est en commentaires dans votre fichier `.emacs`

Note : gestionnaire de paquetage

Contenu du fichier `.emacs` pour installer des paquets :

```
;; Configuration des plugins de Melpa
(require 'package)
(add-to-list 'package-archives '("melpa" .
"https://melpa.org/packages/"))
(package-initialize)
;; Les mises à jour de la base de paquetages se font avec la commande
;; M-x package-refresh-contents
;; ;; Si les mises à jours des paquetages ne fonctionnent pas à cause
;; d'un 'Failed to verify signature archive-contents.sig', il suffit de
:
;;
;; 1) positionner la variable suivante a nil
;; M-x set-variable package-check-signature nil
;; 2) rafraichir la base de paquetages
;; M-x package-refresh-contents
;; 3) installer le paquetage de la cle GPG necessaire
;; M-x package-install gnu-elpa-keyring-update
```

Exemple de thème : `load-theme RET doom-nord RET`

Navigation dans `which-key` : `c-h n/p`, refaire le `c-h` à chaque fois (pas très pratique)

Emacs

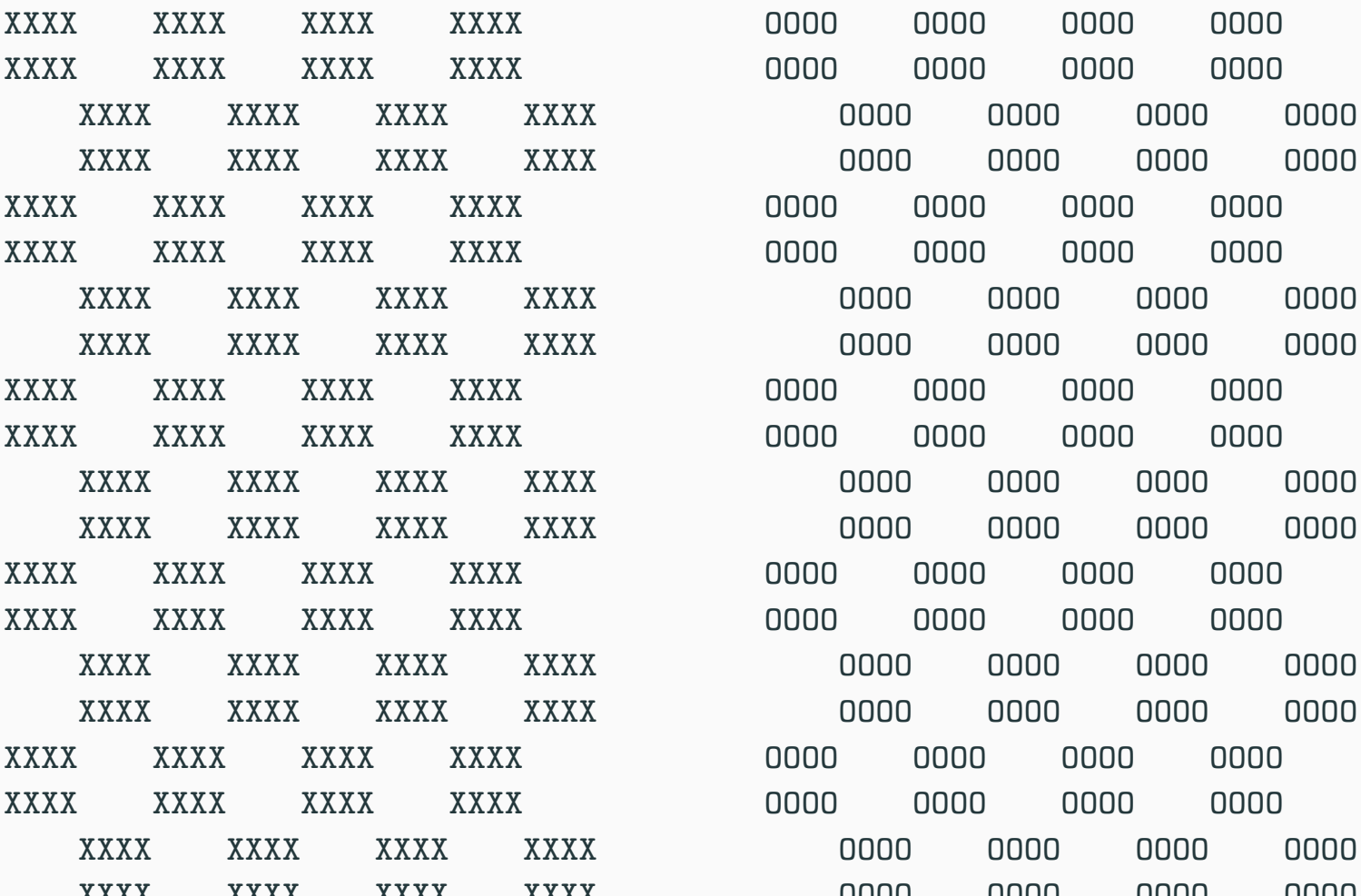
Exercices

Tous les exercices sont à faire **SANS** la souris

Exercice : Damier

- éditer un nouveau fichier
- en un minimum de manipulation réaliser le damier de gauche puis celui de droite (en partant de celui de gauche)

  Question optionnelle : est-ce qu'on pourrait passer du damier de gauche à celui de droite en ligne de commande ?



Exemple : programmation en C

- Récupérer le fichier `https://rollet.zzz.bordeaux-inp.fr/IntroUnix/ressources/tictactoeBugged.c` et l'ouvrir avec `emacs`
- Utiliser la touche `TAB` sur les 6 premières lignes. Que remarquez-vous ?
- Indenter complètement le code (aide : la commande `C-x h` permet de sélectionner tout le buffer)
- Compiler avec la commande `compile`. Il y a des erreurs. Expliquer dans quelle mesure les couleurs et l'indentation permettent d'aider à trouver ces erreurs
- Corriger le code et le compiler. Lancer le programme dans emacs, à l'aide d'un `shell`.
- Ajouter un compteur pour limiter le nombre de parties.

Pour réviser un peu

Quelques exercices pour vous entraîner / réviser :
`https://rollet.zzz.bordeaux-inp.fr/IntroUnix/ressources/exos_emacs-sujet.pdf`