Mini-Référence : Environnement UNIX - Fichiers - Éditeurs

Introduction

Ce document a pour but de vous présenter le système UNIX ainsi que l'environnement de travail que vous utiliserez tout au long de cette année. Il n'a pas la prétention d'être exhaustif, mais plutôt de décrire les manipulations de base du système.

Contenu

Utiliser le gestionnaire de fichiers *(file manager)*

- Pour ouvrir le gestionnaire de fichiers, cliquez sur le bouton «Home Folder» dans le menu «Places» dans la barre en haut.
- Pour créer de nouveaux fichiers, sélectionnez «`File`», ou cliquez avec le bouton droit sur le fond de la fenêtre du gestionnaire de fichiers, puis «`Create Document...`» et «Empty File». Ceci crée un nouveau fichier dans le répertoire actuellement visualisé dans le gestionnaire de fichiers.

> **Note :** pour créer de nouveaux fichiers on utilise plus couramment un éditeur, un autre outil **décrit plus bas**.

- Pour créer de nouveaux répertoires, procédez de façon similaire mais choisissez «`Create Folder`».
- Pour effacer des fichiers ou des répertoires, sélectionnez-les avec le bouton gauche de la souris, puis appuyez sur la touche «Del» (et non pas «Backspace» !» ou bien cliquez la sélection avec le clic *droit* de la souris puis sélectionnez «Move to Trash» dans le menu qui apparaît.
- Pour vous déplacer dans la structure des répertoires, «*double-cliquez*» sur les dossiers qui sont présents dans la fenêtre (si vous double-cliquez sur un fichier et non sur un répertoire, le fichier sera visualisé directement dans le gestionnaire de fichier ou alors l'application concernant le fichier s'ouvrira automatiquement : par exemple `acroread` pour un fichier PDF (**voir plus bas**).

Langage de commande UNIX

Guide du système

L'interface de commande (Shell `tcsh`)

L'interface de commande, ou Shell, permet d'entrer des commandes au clavier et de recevoir des messages en retour. Elle permet également de définir des variables d'environnement, qui contrôlent le comportement de votre

environnement. Seul le Shell **tcsh** sera présenté, car c'est l'un des plus pratiques et c'est également celui qui se trouve sur vos machines, mais il en existent d'autres comme par exemple *bash* ou *zsh*.

Variables d'environnement

Les variables d'environnement sont un moyen de personnaliser votre environnement de travail, ainsi que certains programmes. Par exemple, le Shell `tcsh` utilise la variable

```
$prompt
```

qui définit l'allure du message d'invite (par exemple «`vous@Ubuntu ~>`»).

Les variables d'environnement commencent par un dollar (**$**) et, par convention, sont écrites en majuscules. Les instructions suivantes permettent de les manipuler :

`setenv`	Affiche toutes les variables ainsi que leurs valeurs.
`setenv TEST montestamoi`	Créé ou met à jour la variable `$TEST` avec la valeur «`montestamoi`»
`echo $TEST`	Affiche la valeur de la variable `$TEST`.

Fichiers et répertoires

Sous UNIX, les fichiers sont organisés d'après une architecture en arborescencenbsp;: les fichiers sont disposés dans des répertoires, eux-mêmes pouvant être subdivisés en sous-répertoires. Chaque étudiant(e) possède un répertoire principal, ou «répertoire maison» (*home directory*, noté `~` ou `$HOME`). Seul le propriétaire pourra lire ou écrire dans ce répertoire. Ce dernier sera également accessible depuis n'importe quelle station des salles CO/20, CO/21 et CO/23.

Principales commandes

Nous allons décrire ici la syntaxe de quelques commandes UNIX importantes : **`pwd`**, **`cd`**, **`ls`**, **`mkdir`**, **`rm`**, **`rmdir`**, **`cp`**, **`mv`**, **`cat`**, **`chmod`**, **`man`**, **`quota`**.

`pwd`	affichage du répertoire courant
Le chemin *absolu* (i.e. de la racine du système de fichiers à la position courante dans l'arborescence) est indiqué.	

`cd`	changement de répertoire
Permet de se déplacer dans l'arborescence des répertoires.	
`cd dir` aller dans le répertoire `dir`. `dir` peut être un chemin absolu ou un chemin relatif. `cd ..` revenir au répertoire précédent dans l'arborescence. `cd /` revenir à la racine de l'arborescence. `cd` (tout seul) revenir au répertoire «maison» (*home directory*) de l'utilisateur.	

`ls`	liste le contenu d'un répertoire
Permet de voir le contenu d'un répertoire de diverses manières. Beaucoup d'options sont disponibles, dont les suivantes :	
`ls -l` liste les fichiers sous forme longue, c'est-à-dire avec la taille, les droits de lecture, écriture, exécution, etc... `ls -a` liste tous les fichiers, y compris les fichiers cachés (commençant par un point). `ls -la` liste tous les fichiers, sous forme longue.	

`mkdir`	création de répertoire

Permet de créer un répertoire.

`mkdir dir` crée le répertoire `dir`. `dir` peut être un chemin absolu ou un chemin relatif ; mais il faut avoir le droit de créer le répertoire en question (droit d'écriture dans le répertoire père) !

`rm`	destruction de fichier(s)

Permet de détruire un fichier. Les fichiers ainsi supprimés sont alors **irrémédiablement perdus**.
Pour pallier ce défaut, vous pouvez préférer la command `trash` qui, au lieu de détruire le fichier le met dans la poubelle (ceci dit, depuis que sur nos machines seul `myfiles` est archivé, votre poubelle sera perdue lors de votre déconnexion `:-(`).

La commande `rmdir` permet de détruire un répertoire préalablement vidé.

`cp` et `mv`	copie et déplacement de fichiers

Ces deux commandes s'utilisent de façon similaire. `cp` permet de copier des fichiers d'un répertoire à l'autre, tandis que `mv` efface le fichier source après avoir effectué la copie (il déplace donc le fichier).

`cp fichier1 fichier2`	copie `fichier1` vers `fichier2`.
`cp fichier1 dir`	copie le même fichier dans le répertoire `dir`.
`mv fichier1 dir`	déplace `fichier1` dans le répertoire `dir`.
`mv fichier1 fichier2`	renomme `fichier1` en `fichier2`.

`cat`	concaténation (= mise à la suite, agrégation) de fichiers

Concatène (= met bout à bout), sur la sortie standard, les fichiers passés en arguments (ou l'entrée standard si aucun argument n'est fourni).

`cat fichier1 fichier2 > fichier3` crée le `fichier3` comme concaténation (mise bout à bout) des fichiers `fichier1` et `fichier2`.

`cat > fichier1` crée le `fichier1` et copie dedans tout ce que vous tapez au clavier. Tapez `Ctrl-D` (i.e. la touche «Control» et la touche «D» en même temps) pour quitter la commande.

`chmod`	droits d'accès aux fichiers

Lorsqu'on liste un fichier `monfichier` avec la commande `ls -l monfichier`, on obtient une ligne de la forme suivante :

```
-rw-r--r--   1 prog     liaguest      135 Oct 27 13:56 monfichier
```

- Le premier caractère indique le type du fichier (un "`-`" pour un fichier *classique*, et un "`d`" pour un répertoire).
- Les neuf caractères suivants indiquent les droits d'accès dans l'ordre user, group, other (voir ci-dessous).
- Le chiffre qui suit indique la profondeur de l'arborescence démarrant à l'entrée `monfichier` (1 pour un fichier, au moins 2 pour un répertoire).
- Le code qui suit est le nom de l'*utilisateur* propriétaire du fichier
- Le second code est le nom du *groupe* propriétaire du fichier
- Ensuite viennent : la taille du fichier, les dates et heures de sa dernière modification, et son nom.

Les droits d'accès sont codés par neuf caractères en trois blocs de trois caractères. Le premier bloc correspond aux droits du propriétaire, le suivant aux droits du groupe, et le dernier aux droits des "autres" (utilisateurs qui ne font pas partie du groupe spécifié).

Dans chacun des blocs la présence de la lettre :

- **r** indique un droit de lecture
- **w** indique un droit d'écriture (modification)
- **x** indique un droit d'exécution

Dans l'exemple ci-dessus, `monfichier` peut être lu par l'utilisateur "prog", les utilisateurs appartenant au groupe "liaguest", et les autres. Seul l'utilisateur "prog" peut le modifier. Personne ne peut l'exécuter.

`chmod` permet de changer ces droits (par exemple, pour empêcher les autres utilisateurs de pouvoir lire un fichier).

> **Note :** cette commande `chmod` ne fonctionne pas sous `myfiles` (en tout cas en 2015) ; si vous voulez la tester, il faut aller ailleurs que sous `myfiles`, par exemple à la racine de votre répertoire personnel en faisant tout simplement : `cd`.

La syntaxe générale de `chmod` est :

```
chmod [ugo][+-][rwx] nom_de_fichier
```

Le `+/-` ajoute/enlève les droits de lecture/écriture/exécution (`r/w/x`) pour le propriétaire/groupe/monde (`u/g/o`).

Exemples :
Après la commande `chmod go-r monfichier`, l'entrée `monfichier` listée plus haut apparaîtra ainsi :

```
-rw-------   1 prog     liaguest     135 Oct 27 13:57 monfichier
```

Maintenant, seul l'utilisateur "prog" peut alors lire (et modifier) le contenu du fichier `monfichier`.

Si l'on exécute ensuite `chmod u-w monfichier`, l'entrée devient :

```
-r--------   1 prog     liaguest     135 Oct 27 13:58 monfichier
```

Le fichier est protégé de toute modification, par n'importe qui. Il ne peut pas non plus être effacé.

Enfin, avec `chmod ugo+rwx monfichier`, l'entrée devient :

```
-rwxrwxrwx   1 prog     liaguest     135 Oct 27 13:59 monfichier
```

Tout le monde peut alors manipuler le fichier comme il l'entend, y compris essayer de le faire exécuter.

`man`	manuel en ligne

Permet d'obtenir des informations sur des commandes (pages de « *man*uel» ou «manpage» dans le jargon). Ce manuel peut aussi être accessible par le biais de l'environnement de fenêtres via le bouton «Help» dans le menu principal (celui du chapeau rouge).

`man man` donne la manpage de la commande `man` (!)

`quota -v`	place disponible

La commande "quota -v" vous permet de connaître l'espace disponible sur votre compte, ainsi que l'espace occupé par vos fichiers. **Attention :** s'il vous arrive de dépasser l'espace autorisé pour vos fichiers, vous ne pourrez plus rien écrire sur votre compte; il se peut même que vous ne puissiez plus vous connecter.

Editeurs de textes

Le développement de programmes peut se faire au moyen d'un simple éditeur (dont deux variantes sont décrites ci-dessous), c'est-à-dire un programme servant à créer/éditer de simples fichiers textes.

C'est cette option que nous choisissons dans ce cours. Il est cependant assez courant en programmation plus avancée d'avoir recours à des outils plus sophistiqués, les « Environnement de développement intégrés » (IDE en anglais pour « Integrated Developement Environment ») offrant de nombreuses commodités pour la gestion de projets. Nous n'avons pas retenue cette dernière solution pour des raisons pédagogique (montrer « ce qu'il y a vraiment derrière »).

L'éditeur Geany

Geany est un éditeur de texte récent, léger et qui excelle comme éditeur de code source (programmation) du fait qu'il intègre les fonctionnalités de base d'un IDE (Integrated Development Environment).

Le but de ce mode d'emploi est de vous familiariser avec la puissance de cet éditeur et de faciliter ainsi votre travail. Un conseil important : utiliser autant que nécessaire les commandes d'aide. La version de Geany qui sera décrite est la 0.14 (celle que vous avez). On trouvera un manuel complet de la dernière version sur le site **http://www.geany.org/**.

Frames & *Buffers*

Geany permet d'ouvrir plusieurs fichiers en même temps à l'aide d'onglets (buffers). Il est d'ailleurs recommandé de procéder ainsi (plusieurs fichiers dans le **même** Geany) plutôt que d'ouvrir un nouveau Geany pour chaque fichier.

Quelques raccourcis claviers importants

Voici quelques raccourcis clavier utiles à connaître. Il faut noter que ces combinaisons de touches sont intuitives. La convention suivante est utilisée :

«**Ctrl**» (parfois aussi noté **C-**) signifie qu'il faut laisser enfoncée la touche « **Control**» pendant que l'on presse la suivante. Par exemple, `Ctrl-c` indique qu'il faut presser la touche **Control** et la laisser enfoncée en pressant **c**. La même convention a cours pour la touche *Meta* notée «**M-**», représentée sur le clavier par un losange (on peut aussi utiliser la touche *Escape* - «**Esc**»).

La liste exhaustive des raccourcis clavier peut être obtenue à cette adresse : **http://www.geany.org/manual/current/#keybindings** our directement depuis Geany via le menu «Help» / «Keyboard Shortcuts».

`Ctrl-o`	Ouvrir un fichier
`Ctrl-s`	Sauvegarder le fichier courant
`Ctrl-q`	Quitter Geany
`Ctrl-PgDn`	Passer au buffer suivant
`Ctrl-PgUp`	Passer au buffer précédent
`Ctrl-w`	Fermer le buffer courant
`Ctrl-f`	Rechercher une chaîne de caractères
`Ctrl-l`	Rechercher un numéro de ligne
`Ctrl-h`	Remplacer les occurrences d'une chaîne de caractères par une autre

Copier - Coller

Les opération d'édition de type "Couper - copier - coller" peuvent être effectuées à l'aide des commandes :

`Ctrl-x`	Couper la sélection courante
`Ctrl-c`	Copier la sélection courante
`Ctrl-v`	Coller la dernière sélection

Vous pouvez aussi utiliser les commandes du menu "*Edit*", ainsi que les touches **Cut** (couper), **Copy** (copier) et **Past** (coller) sur la gauche du clavier.

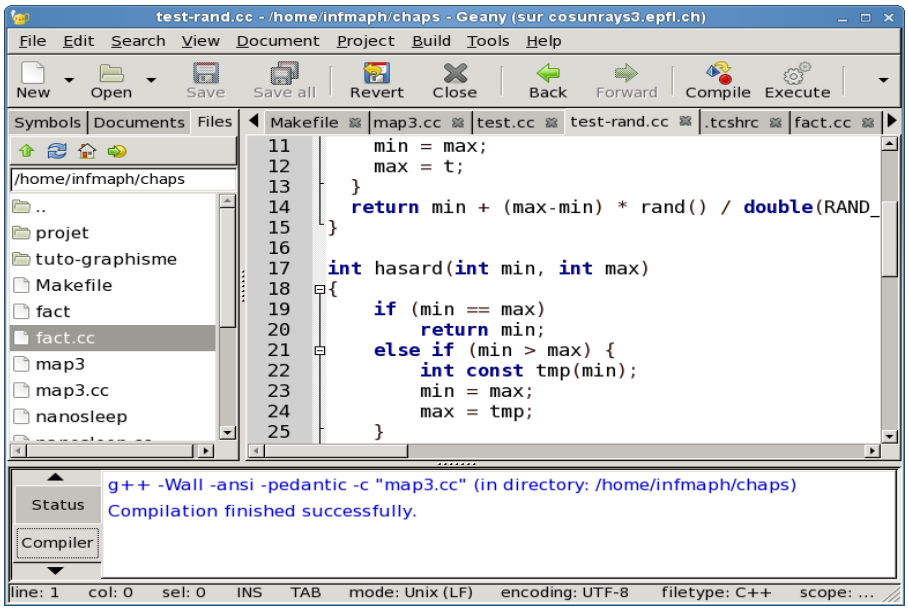
Pour le copier-coller cependant, la solution la plus simple est d'utiliser uniquement la souris : sélectionnez la partie du texte qui vous intéresse en pressant le **bouton gauche** et en déplaçant la souris; relâchez le bouton gauche lorsque le texte sélectionné vous convient, puis placez le pointeur de la souris à l'endroit où vous voulez coller le texte, et cliquez avec le **bouton du milieu**. Cette technique a l'avantage de fonctionner avec quasiment tous les programmes pour Unix (vous pouvez l'utiliser par exemple pour copier du texte depuis mozilla (navigateur Internet) et le coller dans Geany).

Configuration de Geany

La configuration de l'éditeur *Geany* est définie de manière globale. Aucune configuration n'est nécessaire de votre part. Cependant, si vous désirez ajouter vos paramètres personnels, il vous suffit d'éditer le fichier suivant: `~/.geany/geany.conf`.

Fonctionnalités de Geany :

- indentation C++ automatique ;
- commande de compilation (simple) définie automatiquement ;
- colorisation contrastée (cf. image ci-contre) ;
- affichage des n° de ligne ;
- positionnement sur les erreurs de compilation par un *click* ;
- etc.



Pour que la mise en évidence syntaxique, l'auto-indentation et surtout l'adaptation automatique de la commande de compilation puissent se réaliser, il est obligatoire que vous spécifiez un nom de fichier **avec une extension correcte**, c'est-à-dire `.cc` ou `.cpp` pour un programme C++.

Dernière mise à jour le 15 septembre 2021
Last modified: Wed Sep 15, 2021