

TP de familiarisation avec Unix

Terminal – Éditeur (x)emacs – SGF

Vincent Berry - vberry@lirmm.fr

Indications

Les temps indiqués à la fin des titres de paragraphes ne sont qu'indicatifs et approximatifs : votre expérience passée vous amènera à dépasser ou ne pas utiliser complètement le temps indiqué : ce n'est pas grave. Ceux qui ont plus d'expérience que les autres, sont invités à explorer à fond chaque aspect évoqué dans le TP, au besoin en effectuant des expérimentations complémentaires.

Rappel : les questions annotées par un symbole **Ⓢ** peuvent être laissées de côté pour être regardées à la fin de la séance si le temps le permet, sinon elles devront être examinées en dehors des séances de TP.

Table des matières

1	Apprivoisons l'éditeur de texte (x)emacs (15 mn)	1
2	Le terminal de commandes (25 mn)	3
2.1	Les parties de l'arborescence du SGF situées en dehors de votre espace personnel (10mn)	3
2.2	Expérimentation dans votre espace de travail (15mn)	4
3	Intégration des notions vues sur le SGF – assemblage d'un projet (50 mn)	5

1 Apprivoisons l'éditeur de texte (x)emacs (15 mn)

Xemacs et emacs sont deux éditeurs réalisés depuis le même code initial. L'application xemacs est une version d'emacs exploitant plus les possibilités de l'interface graphique, en proposant par exemple des icônes pour réaliser des opérations (couper, copier, coller, etc) qui se font uniquement par raccourcis claviers dans emacs – ces raccourcis sont toujours disponibles dans xemacs. Une conséquence est qu'xemacs fera parfois apparaître des fenêtres de dialogue (par ex pour le choix d'un fichier), ce qu'emacs se refuse à faire. D'autres petites différences existent entre emacs et xemacs dans la composition des menus.

Les TPs Unix peuvent être faits indifféremment avec emacs ou xemacs, aussi verrez-vous souvent "(x)emacs" indiqué, signifiant que vous pouvez utiliser l'un ou l'autre. Pour certains modules d'enseignements que vous suivrez cette année, il est préférable que vous utilisiez xemacs, donc nous vous recommandons dès maintenant de choisir cet éditeur pour effectuer les TPs Unix afin de vous acclimater doucement. Cependant, nous invitons les étudiants les plus aguerris et les plus intrépides à essayer *aussi* emacs.

(x)emacs est un puissant éditeur destiné à l'écriture de programmes (de tous les principaux langages de programmation).

Attention, cet éditeur n'est pas dévolu à la bureautique (vous pouvez faire de la bureautique en utilisant **OpenOffice** (équivalent de Microsoft Office), **Lyx** ou **Latex** (pour les esthètes)).

Bon, passons à un parcours de quelques fonctions.

Démarrez l'éditeur par un clic sur le raccourci que nous avons créé précédemment ou par le lanceur d'applications (<ALT>-<F2>). Délaissez le **buffer scratch** dans lequel il vous positionne peut-être par défaut au lancement de l'application (c'est une zone pour les brouillons). Demandez une nouvelle zone d'édition (c-a-d un nouveau *buffer*). Il est possible que l'éditeur vous demande alors un nom de fichier sous lequel sauvegarder ce que vous allez faire. Si ce n'est pas le cas, C-x c-s tapé dans la nouvelle zone d'édition devrait déclencher une telle demande.

L'éditeur emacs formule cette demande dans la zone qui lui sert à dialoguer avec l'utilisateur, c-a-d le minibuffer, en bas de la fenêtre de l'application. Xemacs préfère peut-être lui ouvrir une fenêtre graphique. Dans les deux cas, indiquez que vous voulez que le fichier édité s'appelle **essai.txt** et qu'il se trouve dans le répertoire **Unix**

Une fois cette indication donnée, notez que le nom de votre buffer est maintenant indiqué (dans la barre d'état en bas de ce buffer).

Couper - Coller - Copier

Copiez-collez dans le buffer les rubriques *Fonctionnalités, Modes d'édition et Personnalisation* de l'article de Wikipedia sur Emacs. Lisez rapidement le texte obtenu.

Dans la rubrique <i>Fonctionnalités</i> du texte, repérez le paragraphe citant des fonctions moins habituelles : Calc, ..., Tetris. Pour chacune ajoutez manuellement une étoile à la place du point virgule final. Cette étoile devra être celle que vous avez copié (M-w) au début de la ligne. Au fait, quel raccourci permet de vous placer en fin de ligne ?	
--	--

Remarque : dans emacs, il est possible que seuls les flèches du clavier et les raccourcis clavier vous permettent de changer la position du curseur. Dans xemacs, il est vraisemblable qu'un clic gauche de la souris permette aussi de le déplacer.

Parmi ces quelques lignes, supprimez la ligne correspondant à ERC et celle correspondant à MULE, tout ceci en utilisant les raccourcis pour :	
<ul style="list-style-type: none">- se placer en début de ligne- supprimer la fin de la ligne (tout ce qui suit le curseur)	

Notez que la barre d'état du buffer (juste en dessous de la zone d'édition) contient des étoiles sur la gauche, indiquant que des modifications ont été effectuées depuis la dernière sauvegarde. Sauvez le fichier et regarder disparaître les étoiles.

Couper-Coller

Sélectionnez maintenant la 1ère phrase du 2ème paragraphe de votre texte (C-<code><espace></code> puis déplacez le curseur avec les flèches du clavier) et couper cette phrase. Quel est le raccourci pour faire cette dernière action ? Placez ensuite la phrase que vous avez coupé au début du 3ème paragraphe.	
--	--

Nature du lien entre les contenus du buffer (mémoire) et du fichier (disque).

Enregistrez le contenu de votre buffer (C-x C-s) et dans une fenêtre Terminal, demandez à voir le contenu du fichier que vous venez de créer (commande cat).	
---	--

En listant le contenu du répertoire contenant votre fichier, vous remarquerez peut-être un fichier au nom très similaire mais suivi par un symbole `~`. Il correspond à une sorte de sauvegarde du fichier dans un état antérieur à la version actuelle. Ce fichier est généré automatiquement par l'éditeur et peut vous être d'un grand secours si vous effectuez des modifications malheureuses dans votre fichier.

Supprimez du disque le ou les fichiers correspondant au buffer que vous êtes en train d'éditer dans l'éditeur (commande rm). Après avoir vérifié que le(s) fichier(s) n'existe(nt) plus sur disque, vérifiez le contenu du buffer dans emacs. Que peut-on en conclure ?	
---	--

Revenez maintenant dans l'éditeur de textes et sauvegardez votre texte à nouveau. Dans le Terminal regardez ce qui c'est produit.

@ Chercher - remplacer

Combien d'occurrences du mot "bibliothèque" (singulier ou pluriel) contient votre texte ? Pour le savoir expérimentez la combinaison **C-s** pour chercher ce mot (puis **C-s** pour les prochaines occurrences). Notez la coloration syntaxique des occurrences voisines de l'occurrence courante.

Nous allons maintenant supprimer toutes les occurrences du texte "[modifier]". Pour cela, placez-vous en début de document et sélectionnez la première occurrence de ce mot. Copiez-la dans le presse-papier de l'éditeur (**M-w**) puis déclenchez une recherche/remplacement : **M-%**. Utilisez alors Couper depuis le presse-papier (**C-y**) pour coller le motif que nous voulons supprimer. Quand l'éditeur vous demande par quoi remplacer (*replace with :*) tapez simplement la touche **<Entrée>** afin de ne remplacer par rien de particulier. A chaque occurrence du motif à remplacer que l'éditeur trouve, il vous faut répondre par "y" ou "n" pour remplacer effectivement cette occurrence, il passe ensuite automatiquement à l'occurrence suivante. Pour lui dire de remplacer systématiquement (sans vous demander) toutes les occurrences restantes, utilisez la touche "!".

Vérifiez. Sauvegardez ensuite votre texte.

@ Shell dans emacs

Dans le menu **Tools** vous trouverez la possibilité d'exécuter un shell (c-a-d un interpréteur de commandes), directement dans un buffer d'emacs. Ceci est très utile quand on fait de l'édition de code avec cet éditeur. Essayez.

2 Le terminal de commandes (25 mn)

Utilisez les commandes appropriées pour répondre aux questions suivantes. Les commandes que vous n'avez pas encore vues en cours sont données entre parenthèses. N'oubliez pas que pour avoir la documentation d'une commande vous pouvez utiliser la commande **man nomCommande** dans un Terminal.

Nous allons expérimenter maintenant les commandes qui permettent de manipuler les répertoires et les fichiers. Ces commandes vous seront utiles non seulement en tant qu'utilisateur de système Unix/Linux, mais aussi en tant qu'informaticien, quand vous créez des scripts ou des programmes qui feront pour vous (ou vos clients) un certain nombre de travaux automatiquement. Dans cette partie vous n'avez pas le droit d'utiliser l'explorateur graphique de fichiers (**Nautilus**). Fermez donc toutes les fenêtres de cette application qui sont actuellement ouvertes dans votre session¹.

2.1 Les parties de l'arborescence du SGF situées en dehors de votre espace personnel (10mn)

Lancez un terminal de commande et utilisez les commandes de base pour répondre aux questions suivantes. Notez les réponses dans la colonne de droite

Initialement dans quel répertoire êtes-vous ? Comment le savoir plus précisément que ce qu'indique l'invite de commande) ?	
Déplacez-vous dans le répertoire /tmp	
Comment vérifier que vous êtes bien arrivé dans le bon répertoire ?	
Fermez maintenant le terminal et ouvrez-le à nouveau. Dans quel répertoire vous situez-vous ? qu'en concluez-vous ?	
Comment savoir si vous avez les droits pour écrire dans le répertoire /tmp ? Par exemple, quel argument faut-il ajouter à la commande <code>ls -l</code> pour obtenir cette information ? Essayez de deux façons possibles : en désignant l'endroit intéressant de façon relative, puis en le désignant de façon absolue	
Dans la liste des répertoires et fichiers présents à cet endroit, comment pouvez-vous distinguer les répertoires des fichiers ?	
Revenez dans votre répertoire d'accueil en utilisant le raccourci <code>~</code> (oui, mais pas tout seul : indiquer d'abord le nom de la commande permettant de changer de dossier).	
Essayez de trouver le répertoire d'accueil de votre enseignant.	
Pouvez-vous vous déplacer dans le répertoire d'accueil de votre enseignant ? Pourquoi ?	
Pouvez-vous créer un fichier dans son répertoire d'accueil (par exemple par la commande <code>touch coucou.txt</code> une fois dans son répertoire) ? Pourquoi ? Comment prouver ce que vous dites ?	

¹A l'extrême limite, s'il vous arrive d'être complètement perdu dans la suite, vous pourrez rouvrir une de ces fenêtres, mais uniquement pour connaître la forme de l'arborescence, pas pour agir dessus. J'insiste sur le fait que même ceci peut être réalisé par des commandes.

2.2 Expérimentation dans votre espace de travail (15mn)

Revenez dans l'application Terminal, où nous allons faire quelques essais de commandes.

Note : dans le Terminal, il est possible de naviguer dans les commandes que vous avez tapées précédemment : on y accède par l'intermédiaire des touches flèches du clavier (haut, bas, gauche, droite). Ceci permet souvent de gagner du temps dans la saisie des commandes futures.

Replacez-vous dans votre répertoire d'accueil	
En utilisant le manuel ci-besoin (commande <code>man</code>), trouvez l'option de la commande <code>ls</code> qui vous permet de connaître tous les fichiers qui se trouvent dans ce répertoire, y compris les fichiers cachés (préfixés par un "point" dans les systèmes Unix) ?	
Pouvez-vous deviner les usages de certains fichiers cachés que vous observez ?	
Créez un répertoire nommé <code>UNIX</code> (commande <code>mkdir</code>) dans votre répertoire d'accueil	
Déplacez-vous dans ce répertoire (<code>cd</code>)	
Créez un sous-répertoire <code>TP2</code> dans <code>Unix</code> et déplacez-vous y	
Créez un fichier <code>titi</code> en utilisant <code>xemacs</code> . Vous pouvez par exemple y saisir : <i>Unix, j'ai même pas peur de toi, et d'abord je connais déjà plein de commandes</i>	
Quittez l'éditeur, revenez dans le Terminal et demandez à visualiser le contenu de ce fichier (commandes <code>cat</code> et <code>more</code>)	
Faites une copie de <code>titi</code> sous le nom <code>toto</code> (commande <code>cp</code>)	
Zut, il devait en fait s'appeler <code>tutu</code> . Renommez la copie du fichier en conséquence (commande <code>mv</code>)	
Double-zut, en fait, on voulait pas mettre le fichier dans le répertoire <code>TP2</code> , mais plutôt à la racine de votre arborescence personnelle. Déplacez le fichier (commande <code>mv</code>)	
En utilisant le mode absolu, changez les droits de <code>titi</code> de sorte que son propriétaire et le groupe aient les droits d'écriture et d'exécution, et les autres seulement le droit d'exécution (<code>chmod</code>)	
Quelle option et quel argument de la commande <code>ls</code> vous permettent de vérifier l'attribution des droits sur ce fichier ?	
Ajoutez le droit de lecture pour les autres en utilisant le mode symbolique (c-a-d en utilisant un/des symbole/s <code>r,w,x</code>)	
Essayez de visualiser <code>titi</code> (commande <code>cat</code>). Que faut-il faire pour y parvenir ?	
Remontez à votre répertoire d'accueil	
Tapez <code>cd U</code> puis appuyez sur la touche <code>TAB</code> afin d'expérimenter la complétion automatique. Validez	

Depuis le répertoire Unix créez un répertoire Ursule dans votre répertoire d'accueil (c-a-d que Unix et Ursule doivent être répertoires frères l'un de l'autre) en utilisant un <i>chemin relatif</i>	
---	--

Revenez dans votre répertoire d'accueil (commande cd) et tapez à nouveau ls U suivi de la touche magique TAB Que se passe-t-il ? Pourquoi ?	
---	--

Essayez à nouveau en tapant maintenant deux fois sur la touche magique	
--	--

Avant de passer à la suite, remettez tous les fichiers texte manipulés ci-dessus dans le répertoire TP2	
--	--

3 Intégration des notions vues sur le SGF – assemblage d'un projet (50 mn)

Nous allons intégrer l'ensemble des notions vues ci-dessus dans une application type. Vous venez de rejoindre l'entreprise *IG+* où vous avez pour rôle d'assembler le travail des différentes équipes de développeurs sur le projet *Musiversal*.

Mise en place (10 mn)

Récupérez l'archive **Musiversal.tar** (depuis le répertoire commun) ou depuis la page web des TPs. Déplacez l'archive dans votre répertoire **TP2**. Décompressez l'archive à l'aide de l'explorateur de fichiers (dans le doute sur cette opération, un réflexe : demander le menu contextuel), mais sans explorer le résultat de cette opération.

Fermez juste après la fenêtre de l'explorateur de fichiers (pour la suite de cet exercice, vous n'y avez plus droit du tout). Pour réaliser votre mission sur ce projet, vous allez utiliser l'application **Terminal**, parfois pour vous déplacer dans la hiérarchie des répertoires du projet et parfois pour taper les commandes nécessaires à l'assemblage du projet. Pour distinguer ces deux utilisations, nous allons recourir à la possibilité qu'offre le Terminal d'utiliser différents **onglets** dans la même fenêtre graphique. Explorez cette possibilité en regardant les menus de cette application. Demandez à disposer de deux onglets et nommez-les **Assemblage** et **Exploration** respectivement. Le premier vous servira à effectuer les tâches décrites ci-dessous, le deuxième à vous promenez dans l'arborescence du projet quand vous le jugerez nécessaire.

Dans l'application Terminal, placez-vous dans le répertoire **Musiversal**. Explorez son contenu pendant quelques minutes (commandes **cd,ls,cat,more**).

Pourquoi est-il nécessaire de faire précéder par un symbole "\" l'espace qui apparaît dans le nom de certains répertoires du projet ?	
--	--

Le projet *Musiversal* est découpé en modules dont la réalisation a été confiée à trois équipes de *IG+*, situées dans des villes différentes. Dans chaque centre de développement, une à deux personnes ont travaillé sur le projet, chacune réalisant un module de l'application finale. L'application que vous devez assembler est composée de 5 modules.

A l'aide du fichier roadmap.lst et des fichiers de sites team.txt déterminez quel développeur est en charge de quel module. Inscrivez la correspondance dans la case de droite.	
---	--

Assemblage du code des modules (20 mn)

Chaque module développé est composé de deux fichiers, dont le nom respecte une convention établie dans l'entreprise IG+ : le code se nomme `module.pl` et sa documentation se nomme `readme.txt`

Créez un répertoire <code>bin</code> situé juste sous le répertoire <code>Musiversal</code>	
Tout en restant dans le répertoire <code>Musiversal</code> , copiez-y les différents modules de <code>code</code> développés dans les centres de la façon suivante : pour chacun utilisez la commande <code>cp</code> après avoir éventuellement utilisé la commande <code>ls</code> pour vous rappeler la structure de l'arborescence du projet.	
Vérifiez combien de fichiers contient maintenant le répertoire <code>bin</code> ? Pourquoi observez-vous ce résultat ?	
Recommencez donc la copie des fichiers de code en ajoutant dans le nom de chacun quelque chose qui le caractérise (le numéro du module, les initiales de son développeur, ...). Evitez les caractères spéciaux dans les noms choisis ainsi que les espaces.	
Pour enchaîner l'exécution des différents modules, le programme de code <code>running.pl</code> vous a été fourni, il se trouve dans le répertoire principal du projet. Placez-vous dans ce répertoire principal, puis demandez l'exécution de l'application en tapant <code>./running.pl</code> Que se passe-t-il ?	
Remédiez à cette situation en donnant les droits nécessaires pour que toute entité puisse exécuter ce fichier.	

Redemandez l'exécution de ce fichier. L'exécution doit maintenant démarrer, mais être vite arrêtée par l'absence d'un fichier de configuration que vous devez composer :

Avec l'éditeur (x)emacs, créez le fichier de configuration <code>modules.cfg</code> dans le même répertoire que <code>running.pl</code> : ce fichier de configuration devra contenir les noms que vous avez choisis pour les fichiers de code des modules que vous avez placé dans le répertoire <code>bin</code> . Chaque ligne du fichier de configuration ne doit contenir que le nom du fichier module (sans chemin) et doit être clôturée par un passage à la ligne (<Entrée>). Les modules doivent être listés dans le bon ordre. Inscrivez dans la case de droite le contenu du fichier de configuration obtenu.	
---	--

Demandez une exécution de l'application pour voir si tout est en place. Le programme est conçu pour vous dire si l'un des modules déclaré est manquant ou pas.

Même si le fichier de configuration est bien établi, et que l'application trouve bien le code des modules dans le répertoire <code>bin</code> , il manque encore un petit quelque chose pour que l'application fonctionne. Pouvez-vous deviner ce dont il s'agit et y remédier ?	
--	--

Bien, maintenant tout doit être en place pour que l'application s'exécute en utilisant le code des différents modules. Faites un essai. Si cela fonctionne correctement, 5 messages doivent apparaître à l'écran, chacun produit par l'exécution d'un des modules, et l'enchaînement des messages doit former une phrase intelligible. Si tel n'est pas le cas, faites les corrections nécessaires pour arriver à un résultat correct.

Documentation du projet (20mn)

Occupons-nous maintenant de la documentation du projet.

Créez un répertoire <code>doc</code> dans le répertoire <code>Musiversal</code>	
Utilisez le <code>man</code> sur la commande <code>ls</code> pour trouver comment obtenir en une seule commande la liste de tous les fichiers qui se situent dans le projet <code>Musiversal</code>	
Repérez où se situent les fichiers <code>readme.txt</code> qu'il va nous falloir assembler pour générer la documentation du projet. Quelles caractéristiques communes partagent les emplacements de ces fichiers dans la hiérarchie ?	
En profitant de l'emplacement symétrique de ces fichiers <code>readme.txt</code> , on veut maintenant utiliser la commande <code>ls</code> mais pour n'obtenir que la liste de ces fichiers. Quelle commande taper ?	
Bien, maintenant que nous savons désigner exactement ces fichiers (sans les autres), nous allons <i>concaténer</i> (assembler) leur contenu en un seul fichier <code>documentation.txt</code> . Pour cela, utilisez la commande <code>cat</code> avec comme argument la désignation calculée ci-dessus.	
Est-ce que les différentes parties de la documentation apparaissent dans le bon ordre ? Pourquoi ?	
Pour palier ce problème, et en vous servant de la correspondance entre numéro de modules et développeurs établie plus haut, vous allez utiliser la commande <code>cat</code> (une seule fois) en lui donnant 5 arguments dans l'ordre pour obtenir à l'écran la documentation dans le bon ordre (pour cela, bien-sûr il est recommandé d'utiliser autant que possible la touche de complétion automatique des noms de répertoires et fichiers (<code><TAB></code>) et d'agrandir la largeur de la fenêtre Terminal à tout l'écran.	
Utilisez l'éditeur de texte (x)emacs pour créer un fichier <code>documentation.txt</code> dans le répertoire <code>doc</code> dont le contenu est obtenu par un seul copier-coller de la documentation assemblée dans le terminal.	

Vous venez de remplir votre première mission au sein de la société *IG+* : bravo !