TP2 Unix – Module F.A.S.

Redirections – Filtres – Archives

Vincent Berry - `vberry@lirmm.fr`

Préambule

L'application lisant ce pdf vous permet de prendre des notes : faites-le !

Table des matières

1 Test de l'application *Plage* (30mn)

L'inscription doit être faite par chaque étudiant sur la page `https://plage.igpolytech.fr`. Cette inscription déclenche l'envoi d'un email à l'adresse que vous avez indiquée pour vérifier votre identité. Dans ce mail, il faut cliquer sur le lien qui permet de vérifier l'adresse.

A partir de ce moment, vous êtes enregistré dans *Plage* et vous allez recevoir (si ce n'est pas déjà fait) par email un sujet pour ce jour "0" de la semaine *Plage* ou vous explorerez en détail les jolis coquillages (shell) présents sur les machines Unix. [1].

L'email que vous avez dû recevoir pour le jour 0 contient un lien qui vous amène à l'énoncé et vous confie une archive que vous devez récupérer, il y a aussi une zone pour charger vos réponses (sous la forme d'une archive), et même une zone de commentaires pour donner un feedback à vos enseignants – oui, ce sont eux les coupables ;)

Allez, allez, même si l'eau est froide, on y va, on part nager : on teste le système en faisant les exercices du jour.

2 Intégration des notions vues sur le SGF dans une application de taille réelle : l'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 reflexe : 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 nécessaire à l'assemblage du projet. Pour

1. Comment ça c'est tiré par les cheveux cette appellation ? ? ? Et la semaine *Piscine alors ! ? ! :)*

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 d'*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 cette 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 `bin` situé juste sous le répertoire `Musiversal`	

Tout en restant dans le répertoire `Musiversal`, copiez-y les différents modules de **code** développés dans les centres de la façon suivante : pour chacun utilisez la commande `cp` après avoir éventuellement utilisé la commande `ls` pour vous rappeler la structure de l'arborescence du projet.	

Vérifiez combien de fichiers contient maintenant le répertoire `bin` ? 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 `running.pl` 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 `./running.pl` 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 `modules.cfg` dans le même répertoire que `running.pl` : 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 `bin` . 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 `bin`, 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, faîtes les corrections nécessaires pour arriver à un résultat correct.

Documentation du projet (20mn)

Occupons-nous maintenant de la documentation du projet.

Créez un répertoire `doc` dans le répertoire `Musiversal`	

Utlisez le `man` sur la commande `ls` pour trouver comment obtenir en une seule commande la liste de tous les fichiers qui se situent dans le projet `Musiversal`	

Repérez où se situent les fichiers `readme.txt` 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 `readme.txt`, on veut maintenant utiliser la commande `ls` 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 *concaténer* (assembler) leur contenu en un seul fichier `documentation.txt`. Pour cela, utilisez la commande `cat` 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 `cat` (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 (<`TAB`>) et d'agrandir la largeur de la fenêtre Terminal à tout l'écran.	

Utilisez l'éditeur de texte (x)emacs pour créer un fichier `documentation.txt` dans le répertoire `doc` dont le contenu est obtenu par un seul copier-coller de la documentation assemblée dans le terminal.	

Créez pour finir une archive `musiversal.tar.gz` contenant toute l'arborescence du répertoire `Musiversal` et conservez une copie (sur clef USB / par email) de cette archive ! ! !	

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