TP 1 : Premier contact

Informatique Fondamentale (IF1)

Semaine du 20 Septembre 2010

Les sujets de TP sont disponibles à l'adresse

`http://www.pps.jussieu.fr/~jch/enseignement/if1/`

Les transparents de cours d'amphi et de Cours/TD sont disponibles sur

`http://www.liafa.jussieu.fr/~picantin/IF1/`

1 Premier contact

Les systèmes de la salle de TP utilisent Unix (Linux ou FreeBSD) et le gestionnaire de fenêtres KDE. Unix et KDE constituent un système multi-tâche : on peut utiliser plusieurs logiciels en même temps.

1.1 Authenti cation

Avant de pouvoir travailler il faut *s'authenti er* auprès du système. Pour cela, il faut entrer son nom d'utilisateur *(login)* et son mot de passe *(password)*.

Normalement, vous avez reçu un nom d'utilisateur et un mot de passe lors de votre inscription auprès du script. Si ce n'est pas le cas, consultez votre chargé de TP.

Exercice 1. Authentifiez vous auprès du système.

1.2 Quelques logiciels

Parmi les logiciels que nous utiliserons, il y a *XEmacs* (éditeur de texte), *Konqueror* (gestionnaire de fichiers et navigateur web), *Konsole* (fenêtre *shell*).

Exercice 2. Lancez *Konqueror*. Déplacez la fenêtre. Changez sa taille. Fermez la.

Exercice 3. Lancez Konqueror de nouveau, et consultez les pages

`http://www.liafa.jussieu.fr/~picantin/IF1/`
`http://www.pps.jussieu.fr/~jch/enseignement/if1/`

1.3 Le shell Unix

Vous aurez besoin de taper des commandes Unix pour plusieurs raisons : lancer un éditeur de texte pour saisir vos programmes, compiler puis exécuter vos programmes, les sauvegarder sur une clé USB, etc. Pour cela, lancez une fenêtre *shell* (sous KDE, cliquez sur l'icône représentant un coquillage devant un écran d'ordinateur).

Le shell permet d'exécuter immédiatement des commandes en tapant leur nom, et éventuellement en précisant sur quoi elles doivent agir. Lorsqu'on le lance, il affiche une *invite*, par exemple :

```
bash-2.04$
```

On peut alors taper une commande, et appuyer sur la touche ■ entrée ■ pour l'exécuter :

```
bash-2.04$ ls -l
```

1.4 La commande `man`

La commande `man` (abréviation de *manual*) permet d'obtenir de la documentation. Par exemple, la ligne de commande `man ls` permet d'obtenir la documentation de la commande `ls`. On peut faire défiler le texte à l'aide de la barre d'espace, et quitter à l'aide de la touche `q`.

Exercice 4. Tapez `man ls` et analysez la structure de la page de manuel. Que fait la commande `ls` ? À quoi sert l'option `-l` ?

1.5 Commandes de manipulation de fichiers et repertoires

Fichiers Un fichier est une suite de données, représentant par exemple un texte, une image etc. Chaque fichier possède un nom, conventionnellement terminé par un point et une suite de caractères indiquant le type de données qu'il contient. Par exemple, le fichier qui contient l'énoncé de ce TP s'appelle `tp1.pdf`, et son nom indique qu'il est au format PDF.

Les systèmes Unix (comme ceux de la salle TP) font une différence entre majuscules et minuscules : `tp0.pdf`, `TP0.pdf` et `TP0.PDF` désignent trois fichiers différents.

Repertoires Sur les systèmes unix, les fichiers sont organisés sous forme d'un *arbre* : chaque fichier est stocké dans un *repertoire*[1], et les répertoires peuvent eux-mêmes contenir d'autres répertoires.

1. Appelé *dossier* par certains.

Le repertoire *home* Le répertoire dit *home* (■ maison ■, ou parfois *repertoire personnel*), noté ■ ~ ■, est l'endroit où vous pouvez stocker vos fichiers personnels. Où que vous soyez, si vous tapez ■ `cd ~` ■, vous vous retrouverez dans le répertoire *home*.

Exercice 5. Allez dans votre répertoire *home*. Contient-il des fichiers ?

Exercice 6. À quoi sert la commande `pwd` ? *(Indication : vous pouvez vous servir de la commande* `man`*.)*

Exercice 7. Quel est le chemin complet de votre répertoire *home* ? Allez à la racine du système de fichiers à l'aide de la commande `cd`. Allez dans le répertoire *home* à l'aide de la commande `cd`.

Exercice 8. A l'invite du shell, tapez sur les touches *eche vers le haut* et *eche vers le bas*. Que se passe-t-il ?

1.6 Quelques aides pour taper les commandes

Edition de ligne Si on se trompe en tapant une commande, et qu'on s'en aperçoit avant d'appuyer sur ■ entrée ■, on peut utiliser les touches *eches gauche et droite* pour déplacer le curseur à l'endroit où est l'erreur.

Historique Si on ne s'aperçoit de l'erreur qu'après avoir démarré la commande, on veut souvent lancer une autre commande corrigée. Au lieu de tout retaper, on peut utiliser la *eche vers le haut*, qui rappelle la commande précédente (puis la commande d'avant, etc., si on appuie plusieurs fois).

Exercice 9. Consultez le manuel de la commande ■ `ls` ■ sans vous servir de la touche ■ `m` ■.

Completion Lorsqu'on veut taper le nom d'un fichier existant, on peut taper le début du nom du fichier puis appuyer sur la touche *tabulation* (marquée *Tab* ou ⇔). Le shell insère alors la fin du nom (s'il y a plusieurs possibilités, le shell complète seulement le plus long préfixe commun). La complétion a deux avantages : elle permet de moins taper, et elle assure que le nom complété existe.

Interrompre une commande Une autre touche à connaître est `^C` (contrôle-C : enfoncer la touche `Ctrl`, puis appuyer ponctuellement sur `C`, et relâcher `Ctrl`). Cette touche interrompt la commande en cours : elle annule la commande en cours, et le shell affiche une nouvelle invite.

2 Premiers pas avec XEmacs

Le système d'exploitation que vous utilisez met à votre disposition différents éditeurs de texte (*kwrite*, *nedit*, *vi*, *Emacs* etc.). Nous utiliserons *XEmacs*, qui est particulièrement bien adapté à la programmation [2].

Lancez le programme XEmacs en tapant ■ `xemacs &` ■.

2.1 Utilisation d'XEmacs

Exercice 10 (Utilisation d'XEmacs).

1. À l'aide d'XEmacs, créez un fichier `poeme.text` ; assurez-vous que le tampon est bien en mode *Text* (regardez en bas de la fenêtre). Tapez votre poème favori (quelques vers suffiront). Sauvegardez, mais ne fermez pas.
2. Toujours dans le même XEmacs, créez un fichier `chanson.text` où vous taperez les paroles de votre chanson favorite (quelques vers suffiront). Sauvegardez.
3. Revenez au tampon du fichier `poeme.text`. *(Indication : regardez dans le menu* Buffers*.)*
4. Pouvez-vous voir les deux fichiers en même temps ? *(Indication : regardez comment scinder la fenêtre dans le menu* View*.)*

3 Premiers pas en Java

Exercice 11. À l'aide du shell, créez un sous-répertoire `IF1_tp1` de votre répertoire *home*.

1. Dans le répertoire `IF1_tp1`, créez un fichier `Menthe.java` , et écrivez-y le programme suivant :

   ```
   import fr.jussieu.script.Deug;
   public class Menthe {
       public static void main(String[] args) {
           Deug.println("Java, c'est pas de la menthe a l'eau.");
       }
   }
   ```

   Sauvegardez le, et vérifiez à l'aide de la commande `ls` que le fichier a bien été créé (vous pouvez également utiliser les commandes `cat`, `more` et `less` pour visualiser directement son contenu sans passer par un éditeur de texte).
2. Tapez la commande suivante :

2. Pour le moment, nous vous imposons d'utiliser *XEmacs*.

```
javac Menthe.java
```

Quels fichiers ont été créés ?

3. La commande `javac` *compile* le *chier source* `.java` en un fichier `.class`. Ce processus sera étudié en détail en cours. Le fichier nommé `Menthe.class`, dit *chier bytecode*, contient un code dit *bytecode*, qui peut être exécuté par la commande `java`. Exécutez-le :

   ```
   java Menthe
   ```

4. Enlevez le point-virgule de la ligne 4 du fichier `Menthe.java` et compilez-le à nouveau. Que se passe-t-il ?

5. Toujours dans le répertoire `IF1_tp1`, créez un fichier `Division.java` qui contient le code suivant :

   ```
   import fr.jussieu.script.Deug;
   public class Division {
       public static void main(String[] args) {
           int n, r;
           Deug.println("Entrez un entier");
           n = Deug.readInt();
           r = 42 / n;
           Deug.println("Le resultat est : " + r);
       }
   }
   ```

   Ce programme demande à l'utilisateur d'entrer un nombre entier n, puis affiche la partie entière de $42/n$.

   Compilez ce programme, vérifiez que le fichier *bytecode* a été créé, puis testez le programme. Que se passe-t-il si vous entrez 0 ?