2 - Le Shell
Découverte

Laurent Tichit

5 avril 2011

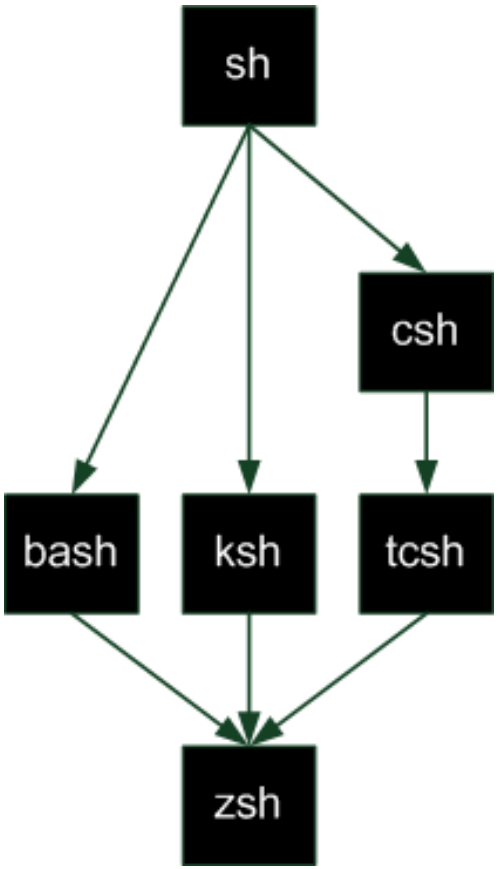
Plan

Shell

L'interpréteur de commandes : le Shell

- Logiciel faisant partie du système d'exploitation.
- Joue le rôle d'intermédiaire entre l'usager et les commandes.

Il est aussi appellé **interpréteur de commandes**. Il en existe plusieurs.



Le shell par défaut sous Linux est **bash**.

Intérêt :

- Pouvoir effectuer des opérations complexes sur plusieurs fichiers...
- Pouvoir écrire des *scripts Shell* et automatiser des tâches

Le Shell, atouts

- Le shell est un langage interprété.
  - Erreurs peuvent être facilement localisables.
  - Modifications faciles.
  - Pas de compilation ou d'édition de liens.
- Le shell manipule, à la base, des chaînes de caractères.
  - Raisonnement uniforme en termes de chaînes.
- Le shell est adapté au prototypage rapide.
  - Communication entre processus (tubes).
  - Substitutions de commandes et de variables.
- Le shell est un langage ≪ glu ≫.
  - Connection de composants écrits dans des langages différents.

Le Shell, lacunes

- Syntaxe quelque peu « ésotérique » et d'accès difficile.
- L'oubli ou l'ajout d'une espace provoque facilement une erreur de syntaxe.
- Plusieurs syntaxes pour implanter la même fonctionnalité.
  - Substitution de commande.
  - Écriture d'une chaîne à l'écran.

  Car besoin de compatibilité ascendante avec le Bourne shell (**sh**), shell historique
- Certains caractères spéciaux ont des significations différentes suivant le contexte. Par exemple, les parenthèses :
  - Liste de commandes.
  - Définition de fonction.
  - Ordre d'évaluation d'une expression arithmétique.

Environnement de travail

Grâce au Shell il est possible

- De rappeler des commandes au moyen de l'historique
- D'interagir avec la commande courante
- De gérer des travaux lancés en arrière-plan
- D'initialiser les variables d'environnement (configuration)
- D'exécuter un *pipeline* de commandes

On exécute une commande

- en mode interactif ; ou
- par l'intermédiaire d'un *script*

Le prompt

En mode interactif, le **prompt** (ou invite de commande) est affiché à l'écran.

`tichit@iml230:~$`

`login @ nom-de-la-machine : répertoire-de-travail $`

Cette chaîne est paramétrable : il suffira de modifier la valeur de la variable prédéfinie PS1 (Prompt Shell 1).

L'interpréteur de commandes : le Shell

L'interpréteur de commandes effectue les opérations suivantes :

- Ecriture sur l'écran d'une invite de commande : le **prompt**.
  `tichit@iml230:~$`
- Lecture au clavier d'une ligne tapée par l'utilisateur et se terminant par un *return*.
  `tichit@iml230:~$` **cal 6 2009**
- Interprétation de la commande, le premier mot étant le nom de la commande désirée, les autres étant ses arguments.
- Exécution de la commande désirée et affichage des résultats.

```
      juin 2009
di lu ma me je ve sa
    1  2  3  4  5  6
 7  8  9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30
```

- Retour à la première étape.

Commandes

Commandes

Il existe des centaines de commandes. Exécutez-en quelques-unes :

```
tichit@iml230:~$ date
 mar sep 16 12:36:11 CEST 2008
tichit@iml230:~$ cal
     avril 2009
di lu ma me je ve sa
           1  2  3  4
 5  6  7  8  9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30
tichit@iml230:~$ whoami
tichit
tichit@iml230:~$ who am i
tichit   pts/0        2009-04-15 14:58 (:0)
```

Commandes

Certaines commandes demandent des paramètres qui lui sont passés sur la ligne de commmande : `tichit@iml230:~$` **cal 2009**

```
                                2009

      janvier               février                 mars
di lu ma me je ve sa  di lu ma me je ve sa  di lu ma me je ve sa
                1  2   1  2  3  4  5  6  7   1  2  3  4  5  6  7
 4  5  6  7  8  9 10   8  9 10 11 12 13 14   8  9 10 11 12 13 14
11 12 13 14 15 16 17  15 16 17 18 19 20 21  15 16 17 18 19 20 21
18 19 20 21 22 23 24  22 23 24 25 26 27 28  22 23 24 25 26 27 28
25 26 27 28 29 30 31                       29 30 31

       avril                  mai                  juin
di lu ma me je ve sa  di lu ma me je ve sa  di lu ma me je ve sa
          1  2  3  4                 1  2      1  2  3  4  5  6
 5  6  7  8  9 10 11   3  4  5  6  7  8  9   7  8  9 10 11 12 13
12 13 14 15 16 17 18  10 11 12 13 14 15 16  14 15 16 17 18 19 20
19 20 21 22 23 24 25  17 18 19 20 21 22 23  21 22 23 24 25 26 27
26 27 28 29 30        24 25 26 27 28 29 30  28 29 30
                      31
```

Commandes

- **echo [argument]** : écriture sur la sortie standard.
  `tichit@iml230:~$` **echo Bonjour Laurent**

  `Bonjour Laurent`
- **echo** sans argument : affiche un retour à la ligne.
  `tichit@iml230:~$` **echo**

- **cat [fichier...]** : liste le contenu des fichiers passés en paramètre.

Syntaxe d'une commande

Rappelons la syntaxe générale d'une commande.

```
[ chemin/]nom_cmd [ option ... ] [ argument ... ]
```

- Suite de mots séparés par un ou plusieurs séparateurs.
- Un séparateur est une tabulation horizontale ou un espace.
- Un mot est une suite de caractères non blancs.
- Plusieurs caractères ont une signification spéciale pour le Shell : ce sont les méta-caractères (ex : |, <).

Une commande comporte généralement un certain nombre de paramètres spécifiques appelés **options**. Elles permettent de modifier le comportement de base de la commande. Les options sont indiquées par le caractère -.
Exemple : `ls -l`

Historique

Historique

- Naviguer dans l'historique des commandes : *flèches HAUT* et *BAS*

- Appeler l'historique des commandes : `tichit@iml230:~$` **history**

```
494  deborphan
495  deborphan --guess-all
496  deborphan --all
497  apt-get update && apt-get --purge dist-upgrade && apt-get --pur
498  apt-get install synaptic
499  apt-get update && apt-get --purge dist-upgrade
500  history
```

- Rappeler l'une des dernières commandes tapées : **!numero** ou **!début_de_commande**. Exemple :
  `tichit@iml230:~$` **!499**
  ou
  `tichit@iml230:~$` **!deb**
  Dernière commande commençant par *deb*, donc la numéro *496* : `deborphan --all`.

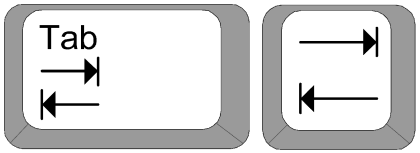
Exercice

Maîtrisez bien l'historique, flèches HAUT et BAS, et l'opérateur **!**.

Complétion

La complétion

Sous le *Shell* La touche <TAB> (tabulation) permet d'effectuer une complétion.



Le premier mot est **toujours** supposé être une *commande*, les suivant, des chemins vers des *fichiers* accessibles. Exemples :

- `tichit@iml230:~$` **ec<TAB>**
  echo
- `tichit@iml230:~$` **da<TAB>**
  date
- `tichit@iml230:~$` **who<TAB><TAB>**
  `who  whoami`
- `tichit@iml230:~$` **whoa<TAB>**
  whoami

Exercice

Maîtrisez bien la complétion pour taper plus vite. Vous pouvez aussi tester les raccourcis clavier :

- CTRL-A : retour en début de ligne de commande
- CTRL-E : aller en fin (*end*) de ligne de commande
- CTRL-U : effacer la ligne de commande
- CTRL-C : annuler la ligne de commande (et considérer qu'elle a échoué)

Manuel

Le Manuel

Documentation sur les commandes Unix, les fonctions C, Perl...
`tichit@iml230:~$` **man [section] 'nom de commande'**
Les sections traditionnelles du manuel sont :

1. **Commandes usager** : commandes utilisateur Unix.
2. **Appels systèmes** : interface entre les programmes et le système d'exploitation.
3. **Fonctions de bibliothèques** : fonctions C qu'un programme peut appeler.
4. **Périphériques** : interfaces avec les différentes ressources matérielles.
5. **Fichier** : formats de fichier les plus importants.
6. **Jeux** : jeux standards de Unix.
7. **Divers** : informations diverses (impression, édition).
8. **Administration** : commandes d'administration.

Le Manuel

Pour quitter le manuel, taper : **q**
Pour rechercher un mot 'mot' dans le manuel, taper : **/mot**
Tapez **n** pour afficher les occurrences suivantes de 'mot'.
Exemple :

- Cherchez l'occurrence de 'include' dans **man 2 time**
- Cherchez l'occurrence de 'example' dans **man 1 time**

Le Manuel - options

Pour la commande **man** :

- L'option -f permet d'obtenir un rapide survol de l'action d'une commande :
  `tichit@iml230:~$` **man -f nom-de-commande**
- L'option -k permet d'obtenir les différentes pages de manuel contenant un mot clé :
  `tichit@iml230:~$` **man -k mot-clé**

Le Manuel

```
tichit@iml230:~$ man -k printf

fprintf (3)          - Formatage des sorties
vfprintf (3)         - Formatage des sorties

tichit@iml230:~$ man -k vesa

cvt (1)              - calculate VESA CVT mode lines
gtf (1)              - calculate VESA GTF mode lines
setvesablank (8)     - Turn VESA screen blanking on or off
vesa (4)             - Generic VESA video driver

tichit@iml230:~$ man -f write

write (2)            - Écrire dans un descripteur de fichier
write (1)            - send a message to another user

tichit@iml230:~$ man -f clock

clock (3)            - Déterminer la durée d'utilisation du proces
clock (8)            - query and set the hardware clock (RTC)
```

Exercice

- Testez le manuel.
- A quoi sert la commande **echo** ?
- A quoi sert l'option **-n** de la commande **echo** ?