Système d'exploitation

Jean-Sébastien Coron

Université du Luxembourg

October 25, 2009

Scripts shell

- Un script shell est un fichier qui contient une liste de commandes.
  - En exécutant le fichier, ces commandes sont exécutées automatiquement.
  - Utile pour automatiser des taches répétitives.
- Nouvelles instructions:
  - Instruction `if`: exécution conditionnelle.
  - Instruction `for`: répétition d'une suite d'instructions.

Execution conditionnelle

- Il est possible d'exécuter une suite d'instruction sous une certaine condition.
  - ```
    if [ test ]
    then
      commande-si-vrai
    else
      commande-si-faux
    fi
    ```
- Le test peut être une condition sur un fichier, ou une comparaison de chaine ou de valeur.
  - Ne pas oublier les espaces dans `if [ test ] then`
    ```

Conditions sur les fichiers

- `[ -s file ]`: vrai si le fichier existe et est non vide.
- `[ -f file ]`: vrai si c'est un fichier ordinaire.
- `[ -d file ]`: vrai si c'est un répertoire.
- `[ -r file ]`: vrai si le fichier est libre en lecture.
- `[ -w file ]`: vrai si le fichier est libre en écriture.
- `[ -x file ]`: vrai si le fichier est exécutable.

Exemple

- Afficher le contenu d'un fichier s'il existe.
  - ```
    #!/bin/bash
    if [ -f $1 ]
    then
      cat $1
    fi
    ```
- Créer un répertoire `toto` s'il n'existe pas déjà.
  - ```
    #!/bin/bash
    if [ !  -d toto ]
    then
      mkdir toto
    fi
    ```

Comparison de valeurs et chaines

- Entiers
  - `[ $X -eq $y ]`: vrai si X=Y.
  - `[ $X -ne $Y ]`: vrai si X<>Y.
  - `[ $X -lt $Y ]`: vrai si X<Y.
  - `[ $X -le $Y ]`: vrai si X<=Y.
- Chaîne:
  - `[ "$A" = "$B" ]`: vrai si chaine A = chaine B.
  - `[ "$A" != "$B" ]`: vrai si chaine A <> chaine B.

Exemple

- Script `pos` déterminant si un nombre est positif.
  - ```
    #!/bin/bash
    if [ $1 -gt 0 ]
    then
      echo "$1 est positif"
    fi
    ```
  - ```
    $ pos 2
    2 est positif
    ```

Exemple

- Script `quitter` demandant si l'utisateur veut quitter.
  - ```
    #!/bin/bash
    echo "Voulez vous quitter ?"
    read rep
    if [ "$rep" = y ]
    then
      echo "exiting..."
      exit 0
    fi
    ```
  - ```
    $ quitter
    Voulez-vous quitter ?
    y
    exiting...
    $
    ```

Exemple

- Script `fichtxt` déterminant si un fichier a l'extension .txt
  - ```
    #!/bin/bash
    if [ "${1##*.}" = "txt" ]
    then
      echo $1 est un fichier .txt
    else
      echo $1 n'est pas un fichier .txt
    fi
    ```
  - ```
    $ fichtxt toto.txt
    toto.txt est un fichier .txt
    ```

Combinaison de tests

- Expressions:
  - `[ $E -a $F ]`: vrai si les expressions E et F sont vraies.
  - `[ $E -o $F ]`: vrai si E ou F est vraie.
- Example: script demandant une valeur en 0 et 100.
  - ```
    #!/bin/bash
    if [ \( $1 -gt 100 \) -o \( $1 -lt 0 \) ]
    then
      echo "out of range"
    fi
    ```
    ```

Boucle `for`

- La boucle `for` permet d'exécuter une suite d'instruction avec une variable parcourant une suite de valeurs.
  - `for` *variable* `in` *list*
    `do`
    statement
    `done`
- La liste peut être un ensemble de fichier.
  - `for x in *.txt`
    `do`
    statement
    `done`

Boucle `for`

- Exemple:
  - Le script:

```
for x in chien chat canard
do
  echo "x=$x"
done
```

  - Affiche:

```
x=chien
x=chat
x=canard
```

Traitement de fichier

- La boucle for permet d'appliquer la même suite d'instruction à un ensemble de fichiers:
  - ```
    for fichier in *
    do
       if [ -d "$fichier" ]
       then
         echo "$fichier:  répertoire"
       fi
    done
    ```
  - Affiche la liste des répertoires dans le répertoire courant.
    ```