Module GNU/Linux CEFIPA : TP n°4 - Scripting
Corrigé

Nicolas Burrus

26 Avril 2005

But du TP

Le but de ce TP est de créer un script `trash.sh` permettant d'émuler une "corbeille" sous Linux. Au lieu d'utiliser la commande `rm`, on pourra ensuite utiliser notre script `trash.sh` qui ne supprimera pas vraiment les fichiers ni les répertoires mais les compressera et les stockera dans un répertoire spécial. De cette façon, ils resteront récupérables.

Question 1

Créer le script `$HOME/trash.sh`, qui se contente d'afficher "Vide" pour l'instant. Exemple :

```
$ cd
$ ./trash.sh
Vide
```

Réponse :

```
$ cat ~/trash.sh
#!/bin/sh

echo Vide
```

Question 2

Faire en sorte que le script puisse être appelé de n'importe où :

```
$ cd /tmp
$ trash.sh
Vide
```

Réponse :

```
$ export PATH=$PATH:$HOME
  ou bien
$ alias trash.sh=~/trash.sh
```

La première solution rajoute le répertoire `$HOME` dans la liste des chemins que le shell doit utiliser pour chercher les programmes.

Question 3

La variable spéciale "$@" regroupe tous les arguments du script (man sh). Écrire "$@" est équivalent à `"$1" "$2" ...`. Faire en sorte que la sortie du script soit :

```
$ trash.sh arg1 arg2 toto
Argument: arg1
Argument: arg2
Argument: toto
```

Réponse :

```
$ cat ~/trash.sh
#!/bin/sh

for arg in "$@"; do
  echo Argument: $arg
done
```

Question 4

Pour chaque argument, le script doit maintenant créer une archive contenant le fichier ou répertoire en argument, nommée `argument.tar.gz`, dans `$HOME/trash`. Exemple :

```
$ cd
$ touch file1 file2
$ mkdir dir1
$ touch dir1/file5
$ trash.sh file1 file2 dir1
$ ls ~/trash/
file1.tar.gz
file2.tar.gz
dir1.tar.gz
$ cd /tmp
$ tar xfz ~/trash/dir1.tar.gz
$ ls dir1
file5
```

S'il n'y a pas d'argument, le script doit afficher un message d'erreur et une aide d'utilisation :

```
$ trash.sh
Usage: trash.sh file [file] ...
```

Réponse :

```
$ cat ~/trash.sh
#!/bin/sh

if test $# -eq 0; then
  echo "Usage: trash.sh file [file] ...
  exit 1 # Sort avec le code de sortie 1 pour signaler l'erreur
fi

for arg in "$@"; do
  FILENAME_WITHOUT_PATH=‘echo $arf | sed -e 's/.*\///'‘
  tar cvfz ~/trash/$FILENAME_WITHOUT_PATH.tar.gz $arg
done
```

La variable `FILENAME_WITHOUT_PATH` contient juste le nom du fichier en argument, en enlevant son chemin, par exemple : `/var/log/syslog` devient `syslog`.

Question 5

Ajouter la date dans le nom de l'archive (on s'aidera de la commande `date`), exemple :

```
$ cd
$ trash.sh file1 file2 dir1
$ ls ~/trash/
dir1-25_Apr_05-19h55.tar.gz
file1-25_Apr_05-19h55.tar.gz
file2-25_Apr_05-19h55.tar.gz
```

Réponse :

```
$ cat ~/trash.sh
#!/bin/sh

if test $# -eq 0; then
  echo "Usage: trash.sh file [file] ...
  exit 1 # Sort avec le code de sortie 1 pour signaler l'erreur
fi

DATE='date '+%d_%b_%y-%Hh%M''

for arg in "$@"; do
  FILENAME_WITHOUT_PATH='echo $arf | sed -e 's/.*\///''
  tar cvfz ~/trash/$FILENAME_WITHOUT_PATH-$DATE.tar.gz $arg
done
```

Question 6

Supprimez effectivement les arguments de `trash.sh` :

```
$ cd
$ trash.sh file1 file2 dir1
$ ls
[vide]
$ ls ~/trash/
dir1-25_Apr_05-19h55.tar.gz
file1-25_Apr_05-19h55.tar.gz
file2-25_Apr_05-19h55.tar.gz
```

Réponse :

Idem que Question 5 plus un `rm`.

Question 7 (bonus)

Faire en sorte que si aucun argument n'est donnée à `trash.sh`, il envoie le répertoire courant dans la corbeille :

```
$ cd
$ mkdir dir1 && touch dir1/file5
$ cd dir1
$ trash.sh
$ cd
$ ls
[vide]
$ ls ~/trash/
dir1-25_Apr_05-19h55.tar.gz
```

Réponse :

```
$ cat ~/trash.sh
#!/bin/sh

# Make backup of $1 directory or PWD
OLDPATH="$PWD"

DATE=‘date ’+%d_%b_%y-%Hh%M’‘

if test $# -eq 0; then
    FILENAME_WITHOUT_PATH=‘echo "$PWD" | sed ’s/^.*\///’‘
    ARCHIVE_NAME="$FILENAME_WITHOUT_PATH-$DATE.tar.gz"
    cd ..
    tar cvfz "~/trash/$ARCHIVE_NAME" "$FILENAME_WITHOUT_PATH"
    rm -rf "$FILENAME_WITHOUT_PATH"
else
    for arg in "$@"; do
      FILENAME_WITHOUT_PATH=‘echo $arf | sed -e ’s/.*\///’‘
      tar cvfz ~/trash/$FILENAME_WITHOUT_PATH-$DATE.tar.gz $arg
      rm -r $arg
    done
fi
```