

Systèmes d'exploitation

Bash

Yves STADLER

Codasystem, UPV-M

11 septembre 2011

1/17

Own the Bash like Brice (be nice)

Variables... encore plus

- `${#chaine}` : longueur d'une chaîne de caractères.
- Tableau : `nom[0]='Bonjour'; nom[1]='Monsieur'; echo $nom[0]`

3/17

Own the Bash like Brice (be nice)

Métacaractères

- `*` correspond à n'importe quel caractère ou ensemble de caractères.
- `?` est équivalent à un caractère quelconque.
- ``` interprète la chaîne de caractères incluse entre deux de ces caractères comme une commande.
- `\` empêche l'interprétation spéciale d'un métacaractère
- `"` début de chaîne.

Dans les chaînes

- dans les simple quote, on met une chaîne de caractère ou les méta-caractères perdent leur signification.
- `" ... "` dans les double quotes, on met une chaîne de caractère ou les méta-caractères perdent leur signification, sauf `\` `"` ``` `$`

2/17

Own the Bash like Brice (be nice)

Les expressions numériques

- `$((expression))`
- `$(expression)`
- `expr expression`
- `==` `!=` `+` `-` `%` `/`
- `let count+=1`

Exemple

- `LAG=$((1234567890 - 1542))`
- `if ((1 + 1 == 2))`

4/17

Own the Bash like Brice (be nice)

Fonctions

```
ma_fonction () {  
    corps  
}
```

Fonctions

- local variable : définit une variable locale, sinon globale.
- \$1 etc...
- return (1-255)

Fonctions

```
#!/bin/bash  
valeur=0  
calcul() {  
    somme=$(( $1 + $2 ))  
    local valeur=$somme  
}
```

5/17

Own the Bash like Brice (be nice)

Découpage en mots

Pour bash un mot est séparé par :
tout élément de \$IFS (défaut : espace, retour à la ligne, tabulation).

Découpage en mots

Peut-être utile pour les boucles for (par exemple)

Own the Bash like Brice (be nice)

Expansion

```
echo sp{e1,i1,a1}l -> donne spell spill spall
```

6/17

Own the Bash like Brice (be nice)

Base/Dir name

- dirname donne le nom de répertoire d'un chemin d'accès
- basename donne le nom du fichier d'un chemin d'accès

Example

```
dirname /home/ystadler/fichier.txt => /home/ystadler/
```

```
basename /home/ystadler/fichier.txt => fichier.txt
```

Own the Bash like Brice (be nice)

Quelques variables d'environnement

- \$PS1 Valeur affichée par le terminal quand il attend une entrée (prompt)
- \$PS2 idem lorsque qu'une commande est en cours d'écriture (écriture d'un for)

Quelques variables d'environnement

```
export PS1="\u@\h \w> "
```

9/17

Own the Bash like Brice (be nice)

Rappels

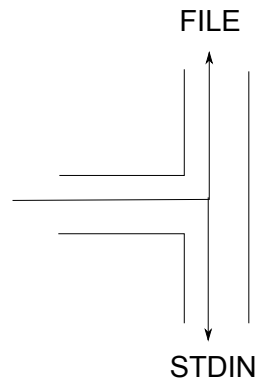
- \$\$ Valeur du PID
- \$# nombre d'argument
- \$? code de retour de la dernière commande
- \$@ concaténation de tous les paramètres (space separated)

10/17

Own the Bash like Brice (be nice)

Tee

- tee dédouble les sorties (envoi sur stdin et un fichier passé en paramètre)
- `ps -l | tee /dev/tty | wc -l` Compte le nombre de ligne renvoyé par la commande ps et l'affiche sur le terminal.



11/17

Own the Bash like Brice (be nice)

Bash comme vous ne l'avait jamais vu

```
while getopts a:bc:d option
do
case $option in
  a) echo "option a : $OPTARG";;
  b) echo "option b ";;
  c) echo "option c : $OPTARG";;
  d) echo "la réponse d ";;
esac
done
```

12/17

Own the Bash like Brice (be nice)

Bash comme vous ne l'avait jamais vu

- Si un : suis l'option, alors on stocke un argument dans OPTARG
- les options peuvent être regroupée -a p1 -bd
- -dba p1
- ...

13/17

Own the Bash like Brice (be nice)

Bash comme vous ne l'avait jamais vu

- `bash -x` : mode debug
- ou `set -x` et `set +x` dans un script

Debug

Permet d'afficher la ligne en cours d'exécution

14/17

Own the Bash like Brice (be nice)

Environnement bash

- `.bash_profile` : personnaliser votre shell.
- `.bashrc` : lorsque l'on lance sans login
- `/etc/profile`
- `/etc/bashrc`

Personnaliser quoi

- PS1, PS2...
- alias de commande
- script de démarrage
- message d'accueil.

15/17

Own the Bash like Brice (be nice)

Un soucis ??

**R.T.F.M.
Read The Fucking Manual**

16/17

Own the Bash like Brice (be nice)

Un soucis ? ?

- `man commande`
- `man man`