



UNIX

Cours 10 : Programmation SHELL (bash) (Partie 4)

Filipe Vasconcelos¹

¹ESME, Lille, filipe.vasconcelo@esme.fr

- ① Pattern Matching
- ② Opérations sur les chaînes de caractères



AAV4

Produire, tester et vérifier le résultat des scripts SHELL (en bash) pour réaliser des tâches algorithmiques simples à complexes



1 Pattern Matching

2 Opérations sur les chaînes de caractères



- En bash, il est possible de faire du filtrage par motif (Pattern Matching) en utilisant le mot clé `case`
- Le mot clé `case` commence d'abord par tester une variable, puis essaye de la mettre en correspondance successivement avec chacun des motifs

Syntaxe

```
case MOT in
    motif_1)          # chaque motif se termine par )
        <COMMANDER1>    # si trouvé tout ce qui suit est exécuté ...
        <COMMANDER2> ;;
    motif_2)
        <COMMANDER1>
        <COMMANDER2> ;;
    *)
        <COMMANDER1> ;;
    esac               # termine le bloc
```

Exemple (GitHub) : (exemple_case_cours.sh)

```
#!/bin/bash
case $1 in
"lundi")
    echo "$1 c'est comme un $1 !";;
"mardi")
    echo "$1 c'est permis !";;
"mercredi")
    echo "$1 c'est ravioli !";;
"jeudi")
    echo "$1 c'est juste jeudi !";;
"vendredi")
    echo "$1 c'est bientot fini!";;
"samedi")
    echo "$1 on est déjà samedi !";;
"dimanche")
    echo "$1 ... rien!";;
*)
    echo "hein ?";;
esac
```



Il est aussi possible de faire des « ou » dans un case pour un ensemble de motif en utilisant la barre verticale pour séparer chaque motif.

Exemple (GitHub) : (exemple_case_cours2.sh)

```
#!/bin/bash
case $1 in
"lundi"|"mardi"|"mercredi"|"jeudi"|"vendredi")
    echo "C'est un jour de semaine snif snif:/";;
"samedi"|"dimanche")
    echo "C'est le weekend !!! youpi!";;
esac
```

```
./exemple_case_cours2.sh $(date)
```

C'est un jour de semaine snif snif :/

Les motifs peuvent être construits avec les méta-caractères * ? [...]

Exemple (GitHub) : (exemple_case_cours3.sh)

```
case ${X} in
??/?/?/????)
    echo "j'ai deviné, c'est une date" ;;
[A-Z])
    echo "j'ai deviné, c'est une majuscule" ;;
[a-z])
    echo "j'ai deviné, c'est une minuscule" ;;
[0-9])
    echo "j'ai deviné, c'est un chiffre" ;;
esac
```

./exemple_case_cours3.sh 16/03/2020

j'ai deviné, c'est une date

Il est aussi possible de tester si une chaîne de caractères satisfait une expression régulière en utilisant le mot clé if et l'opérateur =~

Exemple (GitHub) : exemple_romain.sh

```
if [[ $1 =~ ^(V?(I{0,3})?|I[XV])$ ]] && ![[ $1 =~ ^$ ]]
then
    echo "$1 est un chiffre romain"
else
    echo "$1 n'est pas un chiffre romain"
fi
```

```
./exemple_romain.sh VIII
```

VIII n'est un chiffre romain

Exemple classique d'utilisation de case

<MINTED>

① Pattern Matching

② Opérations sur les chaînes de caractères



Modificateur de sous-chaîne

```
$ chaine=filipe
```

Suppresion de la plus courte sous-chaîne à gauche

```
$ echo ${chaine##*i}
```

lipe

Suppresion de la plus longue sous-chaîne à gauche

```
$ echo ${chaine####*i}
```

pe

Suppresion de la plus courte sous-chaîne à droite

```
$ echo ${chaine%?*}
```

fil

Suppresion de la plus longue sous-chaîne à droite

```
$ echo ${chaine%??*}
```

f



Extraction à partir de l'indice 3

```
$ echo ${chaine:3}
```

ipe

Extraction de 2 caractères à partir de l'indice 3

```
$ echo ${chaine:3:2}
```

ip



Remplacement de la première occurrence

```
$ echo ${chaine/ipe/ou}
```

filou

Remplacement de toutes les occurrences

```
$ echo ${chaine//i/u}
```

fulupe



Suppression de la première occurrence

```
$ echo ${chaine/e/}
```

filip

Suppression de toute les occurrences

```
$ echo ${chaine//i/}
```

flpe



Suppression parmi un ensemble de caractères

Suppression de la première occurrence d'un des éléments d'un ensemble

```
$ echo ${chaine/[aeiou]/}
```

flipe

Suppression de toutes les occurrences des éléments d'un ensemble

```
$ echo ${chaine//[aeiou]/}
```

flp



minuscule -> majuscule

```
$ echo ${chaine^^}
```

FILIPE

majuscule -> minuscule

```
$ echo ${chaine,,}
```



Génération de chaîne

```
$ for i in $PWD/{bin,src,lib};do echo $i ;done
```

```
/home/filipe/bin  
/home/filipe/src  
/home/filipe/lib
```

NOM

basename - Éliminer le chemin d'accès et le suffixe
d'un nom de fichier

SYNOPSIS

```
basename NOM [SUFFIXE]  
basename OPTION... NOM...
```

DESCRIPTION

Les paramètres obligatoires pour les options de forme longue
le sont aussi pour les options de forme courte.

-a, --multiple

Accepter plusieurs paramètres et les traiter
chacun comme un NOM

-s, --suffix=SUFFIXE

Retirer le SUFFIXE si présent ; implique -a

-z, --zero

terminer chaque ligne produite par un caractère
NULL plutôt que par un changement de ligne



Exemple d'utilisation de basename

```
$ basename /usr/bin/sort
```

sort

```
$ basename include/stdio.h -s .h
```

stdio

```
$ basename -a nimp/chaîne1 nimp/chaîne2
```

chaîne1
chaîne2

```
$ basename https://fr.wikipedia.org org
```

fr.wikipedia.

```
$ basename https://fr.wikipedia.org/wiki/Richard_Stallman
```

Richard_Stallman

