# Scripting Shell

Module 05 – Les caractères spéciaux du script



1



### Objectifs

- Conditionner une commande au résultat d'une autre
- Colorier la console

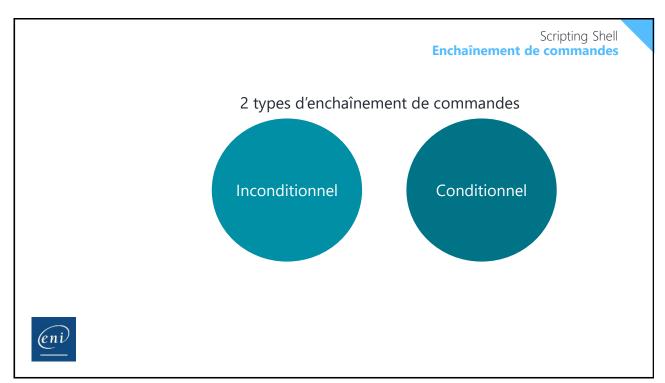


Scripting Shell

# Enchaînement de commandes



3



#### L'enchaînement sans condition

Pour enchaîner des commandes quel que soit le résultat de leur exécution, on utilise le « ; » entre celles-ci.



\$ mkdir /data; touch /data/file1; cat /home/bonjour.txt
mkdir: cannot create directory `./data': Permission denied
touch: cannot touch `./data/file1': No such file or directory
Bonjour le monde !

- Dans cet exemple, l'utilisateur n'a pas le droit de créer un répertoire à la racine (mkdir en erreur).
- La commande de création d'un fichier dans le répertoire échoue aussi (touch en erreur).
- La dernière commande (sans lien avec les précédentes) aboutit.



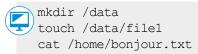
Scripting Shell **Enchaînement inconditionnel** 

#### L'enchaînement sans condition

Dans un script, le traitement des commandes est séquentiel. Le « ; » est substitué par un retour à la ligne.



\$ mkdir /data ; touch /data/file1 ; cat /home/bonjour.txt



Traitement séquentiel



## Scripting Shell **Enchaînement conditionnel**

### Le "et" logique

L'opérateur « && » conditionne l'enchaînement de commandes au résultat d'exécution positif de la commande précédente.

• Cette structure peut se traduire par :

Si la commande aboutit alors je fais la suivante.





7

Scripting Shell **Enchaînement conditionnel** 

#### && Le "et" logique

Le système s'appuie sur le code retour de la commande précédente (variable réservée \$?) :

- Si \$? = 0 (pas d'erreur) : alors la commande suivante est exécutée
- Si \$? ≠ 0 (erreur) : alors la commande suivante n'est pas exécutée



# Scripting Shell **Enchaînement conditionnel**

#### Le "ou" logique

- L'opérateur « | | » conditionne l'enchaînement de commandes à l'échec d'exécution de la commande précédente.
- Cette structure peut se traduire par :
   Si la commande n'a pas abouti alors je fais la suivante.





9

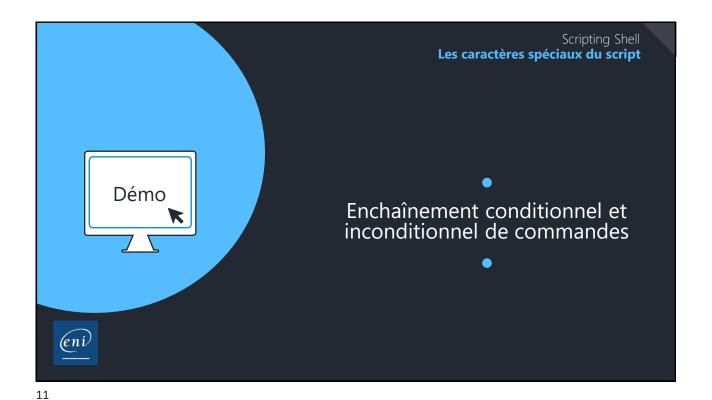
Scripting Shell Enchaînement conditionnel

#### && Le "et" logique

Le système s'appuie sur le code retour de la commande précédente (variable réservée \$?) :

- Si \$? = 0 (pas d'erreur) : alors la commande suivante n'est pas exécutée
- Si \$? ≠ 0 (erreur) : alors la commande suivante est exécutée





Regroupements de commandes

#### Les regroupements de commandes

- Les regroupements de commandes peuvent être utilisés pour :
  - Exécuter plusieurs commandes dans un même environnement.
  - Rediriger le résultat de plusieurs commandes vers un fichier ou un tube.
- Deux syntaxes de regroupement de commandes sont utilisables : (« regroupement ») et {« regroupement »}.



13

Scripting Shell **Les regroupements de commandes** 

#### Regroupement avec création de sous-shell

• Pour <u>regrouper dans un environnement enfant</u> l'exécution de plusieurs commandes, on utilise la syntaxe suivante :

```
$ (cmd1 ; cmd2)
```

• Dans un script, pour plus de lisibilité, le regroupement sera écrit ainsi :

```
Exemple:
pour afficher le nombre total de fichiers
contenus dans /bin et /usr/bin

$ (ls /bin/; ls /usr/bin) | wc -l
```



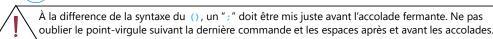
Scripting Shell

#### Les regroupements de commandes

#### Regroupement sans création de sous-shell

• Pour <u>regrouper dans un environnement enfant</u> l'exécution de plusieurs commandes, on utilise la syntaxe suivante :





Dans un script, pour plus de lisibilité, le regroupement sera écrit ainsi :





15

Scripting Shell

# Utiliser les couleurs dans le Shell



#### **Utiliser les couleurs dans le Shell**

- Il est possible d'utiliser quelques couleurs dans le Shell ou dans les scripts.
- Elles peuvent être utiles pour mettre en valeur des informations importantes.

Couleur	Caractères	Fond
Noir	30	40
Rouge	31	41
Vert	32	42
Jaune	33	43
Bleu	34	44
Magenta	35	45
Cyan	36	46
Gris clair	37	47





17

Scripting Shell **Utiliser les couleurs dans le Shell** 

- Pour coloriser une information, les codes couleur seront à encadrer pour :
  - Annoncer la couleur : \033[ ou \e[
    Finir la définition de couleur : m
    Remettre la valeur initiale : \033[0m

```
$ echo -e "voici du \033[1;32mvert\033[0m" voici du vert
```



