



Scripts sous Linux

420-1S6-EM Systèmes d'exploitation



Qu'est-ce qu'un script? (petit rappel)

Contrairement aux langages de programmation comme C++, les langages de scripts ont généralement ces caractéristiques :

- Centrés sur une tâche précise
- Interprétés (ne nécessitent pas de compilation)
- Ne possèdent pas d'interface graphique
- Utilisent des commandes ou des bibliothèques du système
- Mais possèdent tout de même des structures de contrôle

Souvent, les scripts sont liés à un shell. Ils exploitent les commandes de celui-ci d'une manière programmatique.



Qu'est-ce qu'un script Linux?

Un script Linux est un fichier texte contenant une suite de commandes **shell** qui s'exécutent comme si elles étaient inscrites directement en ligne de commande.

C'est l'équivalent du fichier batch en Windows.

Dans un script Linux, on peut gérer :

- > les entrées-sorties et de leur redirection
- > les variables (système, environnement et usager)
- > le passage de paramètres
- > les structures conditionnelles et itératives
- > les fonctions internes



Qu'est-ce qu'un shell?

Quand on parle de ligne de commande, on se réfère vraiment au shell. Le shell est un programme qui prend les commandes clavier et les transmet au système d'exploitation pour qu'il les exécute.

Presque toutes les distributions Linux fournissent un programme shell du projet GNU appelé **bash**.

Saviez-vous que ...

Le nom bash est un acronyme pour **B**ourne **A**gain **S**hell, une référence au fait que bash est un remplacement amélioré pour sh, le programme shell Unix original écrit par **Steve Bourne**.





Les types de scripts sous Linux



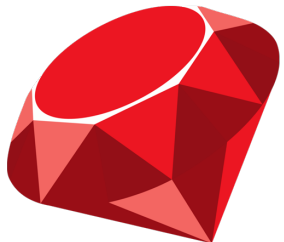
Bash (extension .sh)



Python (extension .py)



Perl (extension .pl)



Ruby (extension .rb)

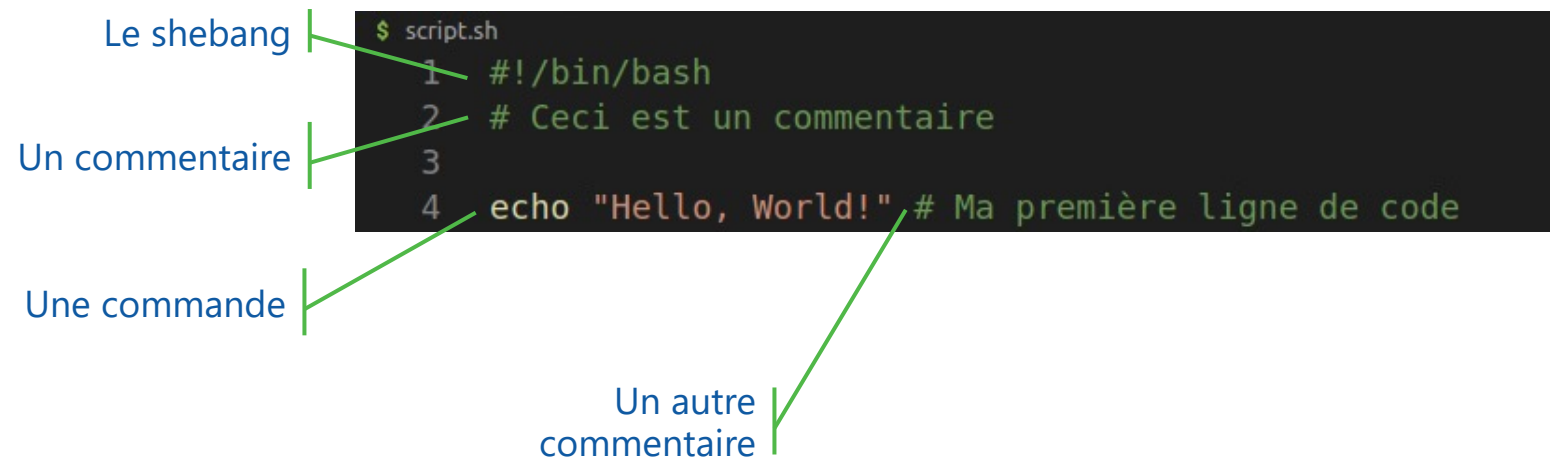


Structure de base des scripts bash



Le premier script : "Hello, World!"

Il est impératif de commencer l'apprentissage d'un nouveau langage avec ce chef-d'œuvre intemporel... 🤗



Première exécution



Pour qu'un script puisse être exécuté, il faut lui donner les **droits d'exécution**.

```
vincent@VD-Ubuntu:~/Bureau/Scripts$ ls -l
total 0
-rw-rw-r-- 1 vincent vincent 0 Oct  7 21:42 script.sh
vincent@VD-Ubuntu:~/Bureau/Scripts$ chmod a+x script.sh
vincent@VD-Ubuntu:~/Bureau/Scripts$ ls -l
total 0
-rwxrwxr-x 1 vincent vincent 0 Oct  7 21:42 script.sh
```

Le fichier n'a pas les droits d'exécution

Cette commande lui donne les droits d'exécution

Le fichier a maintenant les droits d'exécution

À retenir :

chmod a+x nomdufichier.sh

Première exécution



Il faut écrire le nom du script pour l'exécuter en spécifiant son chemin relatif ou absolu :

```
vincent@VD-Ubuntu:~/Bureau/Scripts$ ./script.sh
```

```
Hello, World!
```

```
vincent@VD-Ubuntu:~/Bureau/Scripts$ /home/vincent/Bureau/Scripts/script.sh
```

```
Hello, World!
```

En utilisant ./ (chemin relatif)

En utilisant le chemin absolu

Affichage à l'écran



Il faut utiliser la commande **echo** pour afficher à l'écran :

Script

```
$ script.sh
1  #!/bin/bash
2  # Ceci est un commentaire
3
4  echo "Hello, World!" # Ma première ligne de code
```

Exécution

```
vincent@VD-Ubuntu:~/Bureau/Scripts$ ./script.sh
Hello, World!
```

Caractères spéciaux



Faire précéder par « \ » un caractère spécial permet de l'afficher normalement :

```
# Insertion du caractère spécial « " » dans un chaîne
echo "une \"ligne\""
# Affiche : une "ligne"

# Insertion des caractères spéciaux « " » et « \ » dans un chaîne
echo "\\une \"ligne\"\\\"
# Affiche : \une "ligne"
```

Le shebang #!



Le shebang, représenté par **#!**, est un en-tête d'un fichier texte qui indique au système d'exploitation que ce fichier est un script.

Sur la même ligne, on précise l'interpréteur permettant d'exécuter ce script (i.e. le **langage**).

#!/bin/bash

↑ ↑

shebang interpréteur

Pour plus d'information...

<http://fr.wikipedia.org/wiki/Shebang>

Exemples de shebang



```
#!/bin/bash
# Ceci est un commentaire

echo "Hello, world!" # Un autre commentaire
```

```
#!/usr/bin/python
# Ceci est un commentaire

print('Hello, world!') # Un autre commentaire
```



```
#!/usr/bin/perl
# Ceci est un commentaire

print "Hello, world!\n"; # Un autre commentaire
```

```
#!/usr/local/bin/ruby
# Ceci est un commentaire

puts 'Hello, world!' # Un autre commentaire
```





Édition des scripts

L'édition des scripts

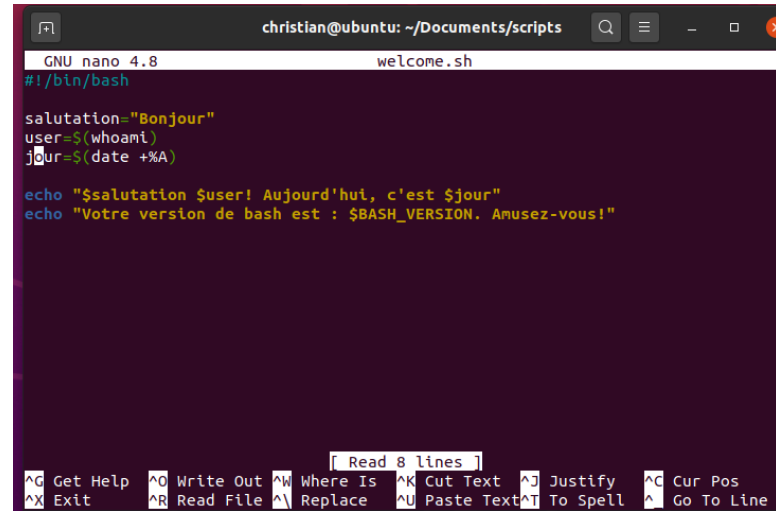


Il est possible d'éditer les scripts dans tous les éditeurs de texte.

Vous pouvez utiliser **nano**, que nous avons déjà utilisé par le passé. Il permet de faire la coloration syntaxique.

Tout comme sur Windows, **Visual Studio Code** est aussi une option. Il permet d'utiliser la souris.

Il faut par contre en faire l'installation :
`sudo apt install code`



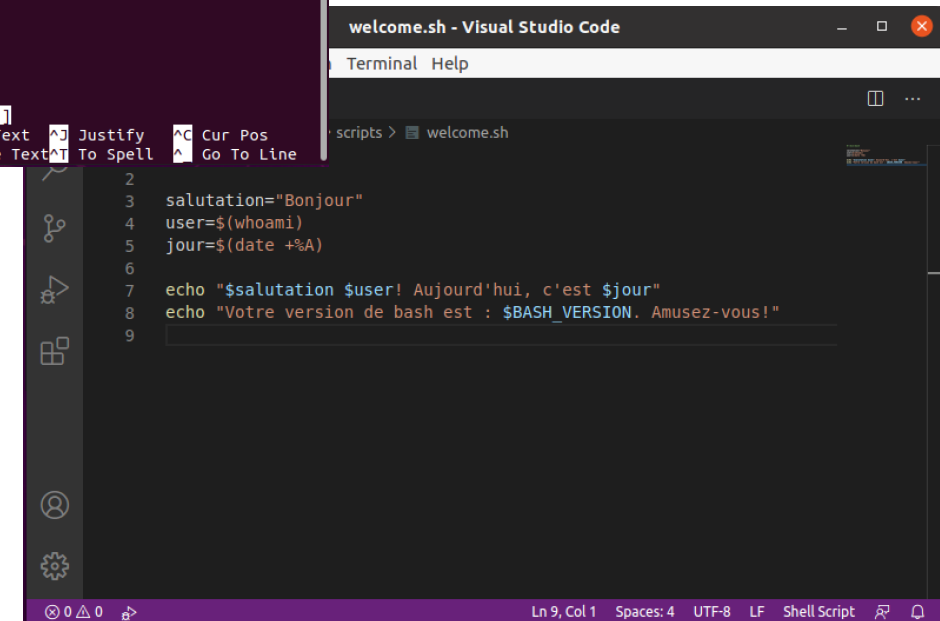
```
christian@ubuntu: ~/Documents/scripts
GNU nano 4.8 welcome.sh
#!/bin/bash

salutation="Bonjour"
user=$(whoami)
jour=$(date +%A)

echo "$salutation $user! Aujourd'hui, c'est $jour"
echo "Votre version de bash est : $BASH_VERSION. Amusez-vous!"
```

nano

code



```
welcome.sh - Visual Studio Code
Terminal Help

scripts > welcome.sh

2 salutation="Bonjour"
3 user=$(whoami)
4 jour=$(date +%A)
5
6
7 echo "$salutation $user! Aujourd'hui, c'est $jour"
8 echo "Votre version de bash est : $BASH_VERSION. Amusez-vous!"
9
```



Quelques concepts utiles

Avant de passer aux choses sérieuses...



Création d'un fichier texte avec contenu

On utilise l'opérateur de redirection **>** pour créer un fichier texte avec un contenu.

L'opérateur de redirection **>>** permet d'ajouter du contenu à un fichier existant.

```
vincent@VD-Ubuntu:~/Bureau/Scripts$ echo "J'aime les hamburgers" > manger.txt
vincent@VD-Ubuntu:~/Bureau/Scripts$ cat manger.txt
J'aime les hamburgers
vincent@VD-Ubuntu:~/Bureau/Scripts$ date +%F >> manger.txt
vincent@VD-Ubuntu:~/Bureau/Scripts$ cat manger.txt
J'aime les hamburgers
2021-10-07
```

Écriture dans un fichier



On peut aisément générer des fichiers texte.

contenu > unfichier.x	contenu >> unfichier.x
# Crée un fichier avec le contenu s'il n'existe pas ou l'écrase (remplace) s'il existe.	# Crée un fichier avec le contenu s'il n'existe pas ou lui ajoute le contenu s'il existe déjà.

```
rejean@rejean-vm:~$ ls
Bureau      Images      Musique      Public      snap      Vidéos
Documents   Modèles     obsession.txt q1.sh       Téléchargements
rejean@rejean-vm:~$ ls > home.txt
rejean@rejean-vm:~$ cat home.txt
Bureau
Documents
home.txt
Images
Modèles
Musique
obsession.txt
Public
q1.sh
snap
Téléchargements
Vidéos
```



Les pipelines (rappel)

Les commandes peuvent être enchaînées pour que le résultat d'une commande soit utilisé par la commande suivante.

```
vincent@VD-Ubuntu:~/Bureau/Scripts$ cat Superheros.txt | grep man
```

```
superman  
batman  
spiderman  
ironman  
wonderwoman  
aquaman  
antman  
greenarrow  
greenlantern  
hawkeye  
hulk  
ironman  
professorx  
spiderman  
supergirl  
superman  
thor  
wolverine  
wonderwoman
```



Les variables d'environnement



Les variables d'environnement

Les variables d'environnement sont des variables déterminées par le système d'exploitation.

On peut obtenir la liste avec la commande **env**

Les principales variables d'environnement utilisées :

`$PATH` # La liste des répertoires séparés par « : » qui sont
recherchés lorsque vous entrez le nom d'un programme exécutable.
`$HOME` # Le chemin du répertoire personnel
`$USER` # Le nom de l'utilisateur
`$PS1` # La chaîne de l'invite
`$SHELL` # Le nom du shell utilisé
`$PWD` # Le répertoire courant

```
mathieu@mathieu-virtual-machine:~$ echo $USER
mathieu
mathieu@mathieu-virtual-machine:~$ echo "Bonjour $USER !"
Bonjour mathieu !
```