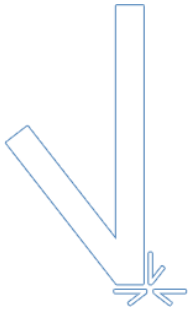
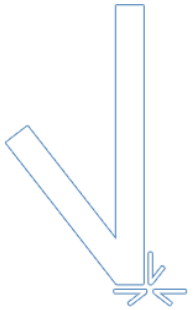
Systèmes d'Exploitation



Objectif :

Comprendre comment un système d’exploitation fonctionne et comment l’utiliser

Sami Radi
sami.radi@virtuoworks.com

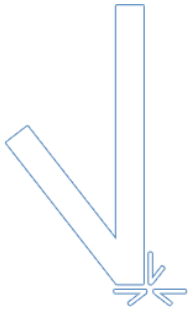


Scripts Shell

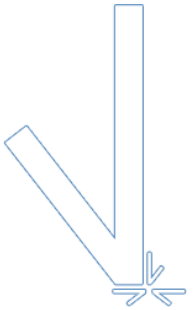
(cours précédent: Systèmes d'Exploitation – Variables d'Environnement)

Sami Radi
sami.radi@virtuoworks.com

I. Systèmes d'Exploitation
I.11. Scripts Shell



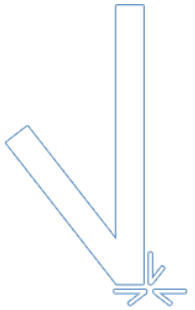
Sami Radi
sami.radi@virtuoworks.com



Shell/BASH

Sami Radi
sami.radi@virtuoworks.com

I. Systèmes d'Exploitation
I.11. Scripts Shell
I.11.a Shell/BASH



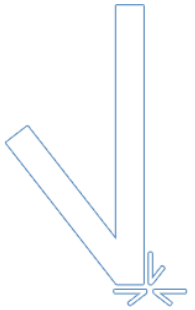
- Sur les systèmes d'exploitation UNIX-Like, « Shell » est le nom du terminal qui est également un Interpréteur de Ligne de Commande (Command Line Interpreter - CLI)

- Le shell original d'UNIX s'appelait *Bourne Shell* d'après le nom de son créateur *Stephen Bourne*.

- Le programme shell est habituellement situé, dans le système de fichier, sous le nom **`/bin/sh`** sur les systèmes d'exploitation UNIX-Like.

sami.radi@virtuoworks.com

I. Systèmes d'Exploitation
I.11. Scripts Shell
I.11.a Shell/BASH

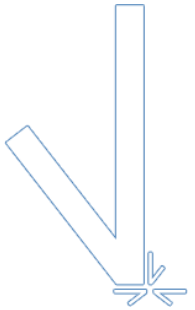


2 autres variantes du Shell peuvent être trouvées sur les SE UNIX-Like :

- **`/bin/bash`** qui est un Shell avec des fonctionnalités additionnelles quand on le compare avec **`sh`** et qui se trouve habituellement sur les distributions de Linux.

- **`/bin/zsh`** qui est un Shell avec des fonctionnalités additionnelles quand on le compare avec **`sh`** et qui se trouve habituellement sur MacOS.

I. Systèmes d'Exploitation
I.11. Scripts Shell
I.11.a Shell/BASH

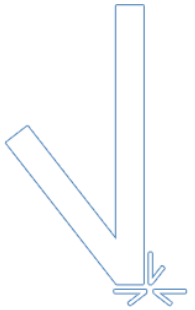


- Le CLI du Shell (ou ses variantes) peut interpréter :
  - **Des lignes de commande écrites par l'utilisateur** sur le CLI;
  - **Des fichiers contenant des commandes Shell**. On appelle ces fichiers des **Scripts Shell**.

- Le langage de commande Shell prend en charge les :
  - Variables;
  - Conditions;
  - Boucles;
  - Fonctions;
  - Remplacements;
  - Commentaires;
  - …

sami.radi@virtuoworks.com

I. Systèmes d'Exploitation
I.11. Scripts Shell
I.11.a Shell/BASH

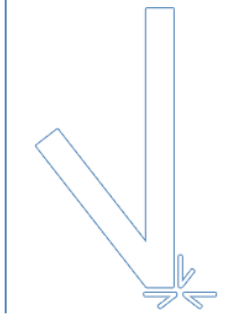


- On peut interpréter et exécuter les scripts Shell de **2 façons**.

1. 1ère sur 2 :
   - **Démarrer l'interpréteur avec le fichier à interpréter en temps qu'argument**. Par exemple :

```
/bin/bash /chemin/vers/le/script/bash
```

I. Systèmes d'Exploitation
I.11. Scripts Shell
I.11.a Shell/BASH



❷ 2éme sur 2 :

- Le fichier de script Shell doit être **exécutable** (le droit `x` doit être positionné sur le fichier de script pour l'utilisateur qui exécute le fichier).

- Le script Shell **doit commencer** par une ligne «**shebang** ». Il s'agit d'une séquence de caractères commençant par `#!` qui indique au système d'exploitation que le fichier doit être interprété comme un programme exécutable avec un interpréteur spécifique.

- Exemple de ligne **shebang** :

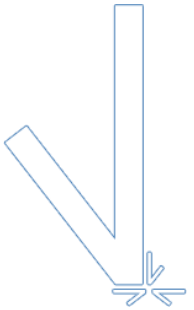
```
#!/absolute/path/to/the/interpreter
```

Par exemple:

```
#!/bin/zsh
```


sami.radi@virtuoworks.com

I. Systèmes d'Exploitation
I.11. Scripts Shell
I.11.a Shell/BASH



! Le langage de commande Shell est un langage **impératif**.

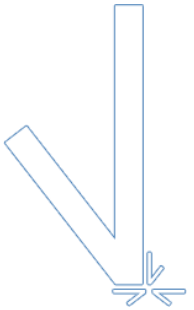
Pour des exemples et des subtilités de programmation en langage bash, veuillez consulter :

https://en.wikibooks.org/wiki/Bash_Shell_Scripting

ou

https://fr.wikibooks.org/wiki/Programmation_Bash

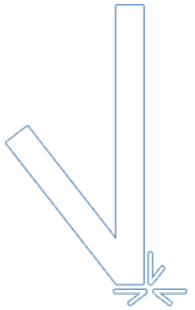
sami.radi@virtuoworks.com



Autres interpréteurs

Sami Radi
sami.radi@virtuoworks.com

I. Systèmes d'Exploitation
I.11. Scripts Shell
I.11.b Autres interpréteurs



- Des **interpréteurs supplémentaires** peuvent être **installés** sur un système d'exploitation de type UNIX.

- Les scripts écrits dans les langages pris en charge par ces interpréteurs peuvent être exécutés de la **même manière** que les scripts shell.

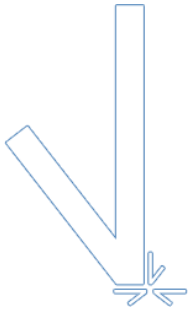
❶ 1er de 2:

- **Démarrez l'interpréteur avec le fichier à interpréter en tant qu'argument**. Par exemple avec l'interpréteur Node dans `/bin` :

```
/bin/node /chemin/vers/le/script/écrit/pour/node
```


sami.radi@virtuoworks.com

I. Systèmes d'Exploitation
I.11. Scripts Shell
I.11.b Autres interpréteurs



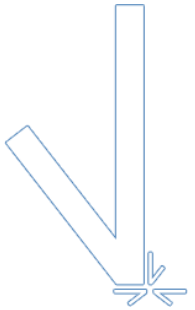
❷ 2éme de 2 :

- Le fichier de script doit être **exécutable** (le droit `x` doit être positionnée sur le fichier de script pour l'utilisateur qui exécute le fichier).

- Le script **doit commencer** par une réplique « **shebang** ». Par exemple avec l'interpréteur Node dans `/bin` :

```
#!/bin/node
```

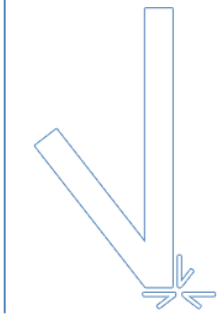

sami.radi@virtuoworks.com



Compilation

Sami Radi
sami.radi@virtuoworks.com

I. Systèmes d'Exploitation
I.11. Scripts Shell
I.11.c Compilation



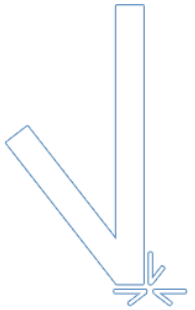
- Pour un programme **écrit en C/C++,** le fichier source (le fichier contenant le code source) doit être **compilé**.

- La **compilation** est un processus par lequel le fichier original sera transformé en un nouveau fichier de **code machine** *binaire*.

- Le *binaire* est spécifique au type de noyau du système d'exploitation sur lequel il est créé. C'est spécifique à la **plate-forme**. Lorsque le *binaire* est chargé dans la mémoire volatile, il peut effectuer des appels système au noyau.

- Le binaire doit être exécutable (le droit `x` doit être positionné sur le fichier de script pour l'utilisateur qui exécute le fichier).

I. Systèmes d'Exploitation
I.11. Scripts Shell
I.11.c Compilation

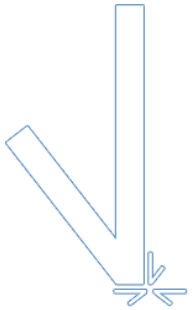


- Pour compiler un programme C/C++, Linux s'appuie sur la **suite logicielle GCC** (GNU Compiler Collection).

- Pour compiler un programme de fichier C/C++, **`gcc`** est l'**utilitaire** à utiliser sous l'interface de ligne de commande.

sami.radi@virtuoworks.com

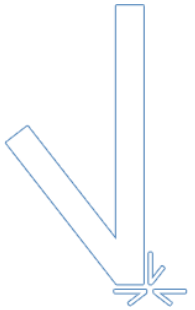
I. Systèmes d'Exploitation
I.11. Scripts Shell
I.11.c Compilation



- Pour compiler un programme de plusieurs fichiers C/C++ dans un ordre spécifique, nous pouvons utiliser :
  - L'**utilitaire `make`**;
  - Avec le **fichier de configuration `makefile`**.

- La configuration de `make` dans le fichier `makefile` définit l'ordre dans lequel compiler les fichiers, les options pour `gcc`, les variables d'environnement, les répertoires à créer, etc...

sami.radi@virtuoworks.com



Gestion des paquets

(voir cours suivant : Systèmes d'Exploitation – Gestion des paquets)

Sami Radi
sami.radi@virtuoworks.com