Mécanismes de base

semaine 3

Personnalisation du shell

- Fichier de profile :
  - Emplacement : /etc/profile
  - Standard POSIX
  - Point de passage obligatoire après un login
  - Fournit par le système, modifiable par l'administrateur
  - Permet de configurer le Shell (couleurs, terminal, variables PATH, PS1…)
  - Il concerne tous les utilisateurs
  - Il peut avoir des compléments (ex. dans /etc/profile.d/)
  - Il peut faire appel (sourcing) à d'autres compléments shell spécifiques (ex. /etc/bash*)

Personnalisation du shell

Autres scripts d’initialisation :

- bash:
  - ~/.profile, ~/.bash_profile: initialisations en tant que shell de connexion
  - ~/.bashrc : à l’invocation du shell (sans fonction de connexion)
  - ~/.bash_logout : quand l’utilisateur se déconnecte du système (nettoyage de l’environnement de travail)
- Bourne shell, Korn shell & POSIX: ~/.profile
- C-shell: ~/.login, ~/.cshrc, ~/.logout
- Zsh: ~/.zshrc
- T-csh: ~/.tcshrc
- Modifiables par l’utilisateur : Personnalisation du shell
- Peuvent être utilisés pour:
  - Redéfinir les variables de /etc/profile
  - Définir de nouvelles variables
  - Définir des fonctions
  - Personnaliser l’apparence (PS1…)
  - Définition d’alias pour des commandes complexes

Couleurs de texts (terminal compatible)

- Afficher des message en couleurs
- Redéfinir les variables $PS1 et $PS2
- Coloriage du texte par la séquence : \033[XXm
  - XX est le code de la couleur
  - Le code couleur « 0 » ramène à la couleur par défaut
    - ex. `echo -e "\033[31mUn texte en rouge et\033[0m retour a la normale"`
  - Les codes de couleurs peuvent se cumuler :
    - `$ echo -e "\033[31m\033[44mHello world rouge sur fond bleu\033[0m"`
  - On peut cumuler plusieurs couleurs/effets séparés par ;
    - `$ echo -e "\033[31;44;1;4;5;7mLe texte\033[0m"`
- le caractère \033 peut être remplacé par \e

Couleur Text		Couleur du fond	
30	Noir	40	Noir
31	Rouge	41	Rouge
32	Vert	42	Vert
33	Orange	43	Orange
34	Bleu	44	Bleu
35	Magenta	45	Magenta
36	Cyan	46	Cyan
37	Blanc	47	Blanc

Nombre	Effet
01	**Gras**
04	Sous-ligné
05	Clignotant
07	Sur-ligné/ Inversé

Personnalisation du prompt

———

- On peut utiliser des caractères spéciaux

```
\d  la date actuelle au format
"Weekday Month Date" ("Tue May 26")
\h  le nom de l'ordinateur
\H  le nom complet de l'ordinateur
\n  saut de ligne
\s  nom du shell ("bash")
\[ démarre une séquence de
caractères non imprimable
(couleurs…),
\] ferme une séquence non
imprimable
```

```
\u  le nom de login de l'utilisateur
courant
\v  la version du bash ("2.00")
\V  la version release du bash, version
+ patchlevel ("2.00.0")
\W  le répertoire de travail actuel
\w  le répertoire de travail actuel
depuis la racine
\!  numéro de commande courant
\@  heure actuelle au format 12h am/pm
\T  heure actuelle au format 12h
HH:MM:SS
\A  heure actuelle au format 24h HH:MM
\t  heure actuelle au format 24h
HH:MM:SS
\$  si l'UID=0 affiche # sinon $
……
```

Les variables

- Types de variables :
  - Variables globales :
    - Le shell en fournit une copie à sa descendance
    - Dans les scripts d'initialisation
      - appelées "Variables d'environnement"
      - commande "env" pour les lister
    - Commande export pour les déclarer (setenv sous csh)
  - Variables locales
    - Créées par l'utilisateur dans le shell actuel
    - Utilisables dans l'environnement actuel
    - Liste de toutes les variables disponibles dans le shell actuel (locales + environnement) : commande "set"
  - Variables de substitution (variables spéciales)
    - Variables positionnelles $1, $2 ...
    - Autres variables spéciales

Les variables

- Déclaration :
  - variable=valeur
  - declare variable=valeur (bash v.2+)
  - typeset variable=valeur (bash, ksh..)
- Contenu d'une variable :
  - $variable
  - ${variable}
- Suppression d'une variable:
  - unset variable

- Exemple :

```
$ nom=ALI
$ adresse="2 bd de France"
$ echo $nom
ALI
$ echo $adresse
2 bd de France
$ unset adresse
$ set
HOME=/home/ali
PATH=/bin:/usr/bin
PS1=$
nom=ALI
```

Les variables : exemples

```
---
$ fleur=rose
$ echo "une $fleur, des $fleurs"
une rose, des
$ echo "une $fleur, des ${fleur}s"
une rose, des roses
$ transport="air mer terre"
$ echo $transport
air mer terre
$ unset fleur transport
$ echo "fleur = $fleur et transport = $transport"

fleur = et transport =
```

Les tableaux (bash)

- tab[i]=valeur : définir un emplacement mémoire pour la i^{eme} case du tableau, et la remplir avec valeur.
- ${tab[i]} : pour consulter la valeur de la i^{eme} case du tableau.
  - Les accolades sont obligatoires pour éviter les ambiguïtés avec ${tableau}[i]
  - ${tableau[0]} est identique à $tableau . N'importe quelle variable peut être considérée comme le rang zéro d'un tableau qui ne contient qu'une case.

- On peut initialiser le tableau entier en une seule expression :
  - $ tab=(zero un deux trois quatre)
  - $ tab=("fry" "leela" [42]="bender" "flexo")
  - $ tab=(['un']="one" ['deux']="two" ['trois']="three")
- L'expression ${tab[@]} fournit une liste de tous les membres du tableau.
- ${!tab[@]} fournit la liste des clés du tableau
- ${#tab[i]} fournit la longueur du ieme membre du tableau
- ${#tab[@]} , donne le nombre de membres du tableau

Les variables (suite)

- Une variable peut être déclarée en lecture seule :
  - `$ readonly variable=valeur`
- Le contenu d'une variable peut être directement lu depuis l'entrée standard (clavier) :
  - `$ read variable`
- Si vous avez des espaces dans la valeur à affecter :
  - employez des guillemets droits : `$ variable="Bonjour Monsieur"`
  - employez des quotes (apostrophes) : `$ variable='Bonjour Monsieur'`
- Les simples quotes groupent les mots et suppriment toute évaluation, les guillemets permettent l'évaluation :
  - `$ nom="Kamel" ; echo 'Bonjour Monsieur $nom'` ⇒ `Bonjour Monsieur $nom`

Variable vide vs variable non déclarée

- Par défaut, une variable qui n’a jamais été affectée est traitée par défaut comme une chaîne vide :
  - `$ echo '-'$inex'-'`
  - `--`
- Une variable à laquelle on affecte une chaîne vide existe quand même. Ce n’est pas comme si on la supprime avec “unset”
- On peut différencier les variables inexistantes des variables vides en utilisant l’option -u à l'exécution du shell (ou avec la commande set) :
  - `$ set -u`
  - `$ echo $inexistante`
  - `bash: inexistante : variable sans liaison`

TD/TP 3

1. Identifiez les fichiers d’initialisation de sessions propres à votre compte.
2. Créez un alias permanent pour remplacer la commande “ls -la” par “ls”
3. Écrivez un script qui compte à rebours de 5 à 0, avec une pause d’une seconde à chaque affichage. (en rouge)
4. Explorez la variable PS1 et redéfinissez de manière permanente votre propre prompte en utilisant des couleurs et en affichant les informations suivantes :

<user>@<host>/<pwd>_<time>_$_