Licence d'informatique
Module de C/Unix

La commande `install`

Examen – Partie Unix

Philippe MARQUET

Septembre 1999

Documents de cours et TD autorisés
On rendra deux copies séparées pour la partie Unix et pour la partie langage C.
La partie Unix est à rendre sur une *copie de couleur*

Important Plusieurs commandes ou codes Bourne-shell sont à développer. On veillera particulièrement à détailler pour chacune des commandes, chacun des codes, un algorithme avant de donner le code Bourne-shell lui-même. La notation sera répartie entre ces deux composantes de vos réponses.

La commande Unix `install` est dérivée de la commande `cp`. Cette commande est particulièrement utilisée dans des `Makefile` pour copier des programmes dans leur répertoire final. L'examen consiste à réécrire cette commande sous forme d'un programme Bourne-shell. Quelques commandes pouvant être utiles et n'ayant pas ou peu été présentées sont décrites dans une annexe à la fin du sujet.

Manuel de la commande `install`

Un manuel de la commande `install` est résumé et traduit ici.

NAME `install` — copie des fichiers et positionne leurs attributs

SYNOPSIS

`install` [options] [backup-options] [-s] [--strip] source dest (1er format)
`install` [options] [backup-options] [-s] [--strip] source... directory (2^{e} format)
`install` [options] [-d,--directory] directory... (3^{e} format)
Options :
[-c] [-g group] [-m mode] [-o owner] [--group=group] [--mode=mode] [--owner=owner] [--help] [--version]

DESCRIPTION `install` copie des fichiers, positionne leurs droits et, si possible, leur propriétaire et groupe propriétaire. `install` est utilisé de manière similaire à `cp` pour copier des programmes dans leur répertoire final. `install` peut aussi être utilisé pour créer ces répertoires et positionner leurs attributs. `install` refuse de copier des fichiers sur eux-mêmes.

OPTIONS

-c Ignorée ; pour compatibilité avec d'anciennes versions de la commande.

-d, --directory Crée chacun des répertoires et leurs répertoires parents si ils n'existent pas. Positionne les propriétaire, groupe, et droits comme spécifié sur la ligne de commande ou avec les valeurs par défaut.

-g, --group group Positionne le groupe propriétaire des fichiers installés ou des répertoires créés au groupe **group** (par défaut le groupe actuel de l'utilisateur).

Id : xam99-sep.tex,v 1.7 2000/11/28 18 :34 :55 marquet Exp

-m, --mode mode Positionne les droits des fichiers installés et répertoires créés à **mode** qui peut être exprimé sous forme d'un nombre octal ou d'un mode symbolique comme dans `chmod`. Le mode par défaut est 0755.

-o, --owner owner Si la commande est exécutée par le super-utlisateur (root), positionne le propriétaire des fichiers installés à **owner** (root par défaut).

-s, --strip Supprime la table des symboles des programmes installés.

--help Affiche un message d'usage de la commande sur la sortie standard et termine sur un succès.

--version Affiche les informations de version sur la sortie standard et termine sur un succès.

Les options de sauvegarde, dites **backup options**, seront détaillées plus loin.

Question 1 (Analyse des arguments)
Donner une liste de variables qui vont mémoriser les valeurs des attributs pouvant être spécifiés par la ligne de commande et le format choisi pour mémoriser ces informations.

En particulier, une variable mémorise lequel des trois formats de la commande est utilisé.

On ne demande pas de donner le code Bourne-shell qui initialise et met à jour ces variables.

Question 2 (Quel format)
Donner un algorithme d'analyse de la ligne de commande permettant de déterminer lequel des trois formats de la commande `install` est utilisé.

Question 3 (Premier et deuxième formats)
Donner le code Bourne-shell de la commande `install` qui traite des premier et deuxième formats de la commande. On suppose bien entendu que la ligne de commande a été analysée et que les valeurs des variables décrites à la question 1 ont été positionnées en conséquence.

Question 4 (Troisième format)
Donner le code Bourne-shell de la commande `install` qui traite du troisième format.

Les options de sauvegarde

La version GNU de plusieurs commandes Unix (au moins `cp`, `install`, `ln`, `mv`) peut créer des sauvegardes des fichiers avant d'écrire une nouvelle version. Les options détaillées ici et la valeur de variables d'environnement contrôlent ces sauvegardes.

-b, --backup Crée des sauvegardes des fichiers qui vont être détruits ou modifiés. Sans cette option, les versions originelles sont remplacées et perdues.

-S, --suffix suffix Ajoute le suffixe à chaque sauvegarde créée par l'option **-b**. Si cette option n'est pas spécifiée, la valeur de la variable d'environnement `SIMPLE_BACKUP_SUFFIX` est utilisée. Si cette variable n'est pas positionnée, la valeur par défaut est '~' (tilde, comme sous Emacs).

-V, --version-control method Utilise la méthode **method** pour déterminer le type de sauvegarde effectuée par **-b**. Si cette option n'est pas spécifiée, la valeur de la variable d'environnement `VERSION_CONTROL` est utilisée. Si cette variable n'est pas positionnée, le type de sauvegarde par défaut est '*existing*'.

Les valeurs valides de **method** sont (les abréviations non ambiguës sont acceptées) :

'*t*' ou '*numbered*' Toujours créer des sauvegardes numérotées.

'*nil*' ou '*existing*' Créer des sauvegardes numérotées des fichiers qui en ont déjà, sinon créer des sauvegardes simples.

'*never*' ou '*simple*' Toujours créer des sauvegardes simples.

Il apparaît donc qu'il existe deux types de sauvegardes : les sauvegardes simples et les sauvegardes numérotées.

Une sauvegarde simple crée un fichier contenant la version précédente du fichier ; les versions plus anciennes sont perdues. Par défaut, pour le fichier de nom `file`, ce fichier de sauvegarde est `file~`.

Une sauvegarde numérotée permet de garder toutes les versions antérieures du fichier. Ces versions successives sont rangées dans les fichiers `file.~1~`, `file.~2~`, etc.

Question 5 (Analyse des options de sauvegarde)
Donner une liste de variables qui vont synthétiser les valeurs des options de sauvegarde données sur la ligne de commande et des variables d'environnement `SIMPLE_BACKUP_SUFFIX` et `VERSION_CONTROL`.

Question 6 (`exists-numbered-backup-file`)
Écrire une fonction `exists-numbered-backup-file` qui retourne un succès si et seulement si il existe une sauvegarde numérotée d'un fichier dont le nom est passé en paramètre.

Question 7 (`next-numbered-backup-file`)
Écrire une fonction `next-numbered-backup-file` qui retourne le nom du fichier de sauvegarde numéroté devant être créé pour un fichier dont le nom est donné en paramètre.

Question 8 (`backup-file`)
Écrire une fonction `backup-file` qui retourne le nom du fichier de sauvegarde devant être créé pour le fichier passé en paramètre en fonction des variables définies à l'exercice 5.

Annexe

Les fonctionnalités de quelques commandes pouvant être utiles sont données ici.

La commande `id` affiche les numéros d'utilisateur (option **-u, --user**) et de groupe actuel (option **-g, --group**). Les noms d'utilisateur et de groupe peuvent être affichés avec l'option **`-n, --name`**. Exemples :

```
% id --user
5408
% id -un
marquet
% id --group --name
enseign
```

La commande `strip` supprime la table des symboles d'un exécutable. Cette table des symboles mémorise les différents noms (variables...) utilisés dans le programme source de l'exécutable. `strip` modifie le fichier passé en argument plutôt que de produire la copie modifiée sous un autre nom.