



Ressourcesinformatiques

Programmation shell **sous Unix/Linux** **sh, ksh, bash**

(avec exercices corrigés)

4^{ième} édition

Christine DEFFAIX RÉMY

Téléchargement
www.editions-eni.fr



Les éléments à télécharger sont disponibles à l'adresse suivante :
<http://www.editions-eni.fr>
Saisissez la référence ENI de l'ouvrage **RI4PRO** dans la zone de recherche
et validez. Cliquez sur le titre du livre puis sur le bouton de téléchargement.

Avant-propos

Chapitre 1 Introduction

1. Définition du shell	19
2. Caractéristiques d'un interpréteur de commandes	19
3. Interpréteurs de commandes (shells)	20
3.1 Historique	20
3.2 Avec quel shell faut-il programmer ?	21
3.2.1 Scripts de démarrage	21
3.2.2 Autres scripts	21
3.3 Nom des exécutable	21
4. Shells abordés dans cet ouvrage	22

Chapitre 2 Mécanismes essentiels du shell

1. Commandes internes et externes	23
1.1 Les commandes externes	23
1.2 Les commandes internes	25
1.3 Implémentation interne et implémentation externe	26
2. Affichage à l'écran	27
2.1 La commande echo	27
2.1.1 Le caractère "\n"	27
2.1.2 Le caractère "\c"	28
2.1.3 Le caractère "\t"	29
2.1.4 Liste des caractères d'échappement	29
2.2 Les commandes print et printf	30

2 _____ Programmation shell

sous Unix/Linux, sh, ksh, bash

3. Le caractère ~ (tilde)	30
4. La commande interne cd	31
5. Substitution de noms de fichiers	32
5.1 Expressions basiques	32
5.1.1 Le caractère *	32
5.1.2 Le caractère �	32
5.1.3 Les caract�res []	33
5.2 Expressions complexes	34
5.2.1 �(expression)	34
5.2.2 *(expression)	35
5.2.3 +(expression)	35
5.2.4 @(expression)	35
5.2.5 !(expression)	36
5.2.6 Alternatives	36
5.3 Interpr�tation du shell	37
6. S�parateur de commandes	38
7. Redirections	38
7.1 Entr�e et sorties standard des processus	38
7.1.1 Entr�e standard	38
7.1.2 Sortie standard	39
7.1.3 Sortie d'erreur standard	39
7.2 H�ritage	39
7.3 Redirection des sorties en �criture	40
7.3.1 Sortie standard	40
7.3.2 Sortie d'erreur standard	42
7.3.3 Sortie standard et sortie d'erreur standard	43
7.3.4 Se prot�ger d'un �crasement involontaire de fichier	44
7.3.5 �liminer les affichages	45
7.3.6 M�canisme interne	45
7.4 Redirection de l'entr�e standard	46

7.5	Redirections avancées.	48
7.5.1	Rediriger les descripteurs 1 et 2 vers le même fichier.	48
7.5.2	La double redirection en lecture.	54
7.5.3	Fermeture d'un descripteur	55
8.	Tubes de communication.	55
8.1	Commandes ne lisant pas leur entrée standard	57
8.2	Commandes lisant leur entrée standard	58
8.2.1	Exemples triviaux	58
8.2.2	Cas des filtres	58
8.3	Compléments	63
8.3.1	Enchaîner des tubes	63
8.3.2	Dupliquer les sorties.	63
8.3.3	Envoyer la sortie standard et la sortie d'erreur standard dans le tube	64
9.	Regroupement de commandes.	65
9.1	Les parenthèses	65
9.2	Les accolades	70
9.3	Conclusion	73
10.	Processus en arrière-plan	74
11.	Exercices	74
11.1	Fonctionnalités diverses	74
11.1.1	Exercice 1 : commandes internes et externes	74
11.1.2	Exercice 2 : génération de noms de fichiers.	74
11.1.3	Exercice 3 : séparateur de commandes	75
11.2	Redirections	75
11.2.1	Exercice 1.	75
11.2.2	Exercice 2	75
11.2.3	Exercice 3	76
11.2.4	Exercice 4.	76
11.2.5	Exercice 5	76
11.2.6	Exercice 6.	76

4 _____ Programmation shell

sous Unix/Linux, sh, ksh, bash

11.3 Tubes de communication	77
11.3.1 Exercice 1	77
11.3.2 Exercice 2	77
11.3.3 Exercice 3	77
11.3.4 Exercice 4	77

Chapitre 3

Paramétrage de l'environnement de travail

1. Variables d'environnement	79
1.1 Liste des variables	79
1.2 Affichage de la valeur d'une variable	80
1.3 Modification de la valeur d'une variable	80
1.4 Principales variables	81
1.4.1 HOME	81
1.4.2 PATH	81
1.4.3 PWD	83
1.4.4 PS1	83
1.4.5 PS2	87
1.4.6 TMOUT	87
1.4.7 TERM	88
1.4.8 LOGNAME	88
1.4.9 Processus et variables d'environnement	88
1.5 Exportation des variables	89
1.5.1 Liste des variables exportées	89
1.5.2 Variables devant être exportées	90
1.5.3 Exporter une variable	90
2. Les options du shell	94
2.1 Activer et désactiver une option du shell	94
2.2 Visualiser la liste des options	94
2.3 Principales options	95
2.3.1 ignoreeof	95
2.3.2 noclobber	95

2.3.3	emacs et vi	96
2.3.4	xtrace	97
3.	Les alias	97
3.1	Définir un alias	97
3.2	Visualiser la liste des alias	98
3.2.1	Visualiser tous les alias	98
3.2.2	Visualiser un alias en particulier	98
3.3	Supprimer un alias	98
4.	Historique de commandes	99
4.1	Paramétrer le rappel de commandes en ksh	100
4.1.1	Option vi	100
4.1.2	Option emacs	101
4.2	Paramétrer le rappel de commandes en bash	105
4.3	La complétion de noms de fichiers	105
4.3.1	La complétion du bash	105
4.3.2	La complétion du ksh	106
4.3.3	Tableau récapitulatif	108
5.	Les fichiers d'environnement	108
5.1	Caractéristiques des fichiers d'environnement	108
5.1.1	Shell de connexion	109
5.1.2	Fichiers d'environnement lus par le shell de connexion	109
5.2	Session utilisant un Bourne Shell	113
5.3	Session utilisant un Korn Shell	114
5.4	Session utilisant un Bourne Again Shell	116
6.	Exercices	118
6.1	Variables d'environnement	118
6.1.1	Exercice 1	118
6.1.2	Exercice 2	118
6.2	Alias de commande	119
6.2.1	Exercice 1	119
6.2.2	Exercice 2	119

6 _____ Programmation shell

sous Unix/Linux, sh, ksh, bash

Chapitre 4

Les bases de la programmation shell

1. Les variables utilisateur	121
1.1 Nommer une variable	121
1.2 Définir une variable	121
1.2.1 Affecter une valeur à une variable	122
1.2.2 Affecter une valeur contenant au moins un espace ...	122
1.2.3 Variable indéfinie	122
1.2.4 Retirer la définition d'une variable	123
1.2.5 Isoler le nom d'une variable	123
1.2.6 Variables numériques	124
1.2.7 Variables complexes	125
1.3 Substitution de variables	126
2. Substitution de commandes	128
3. Caractères de protection	129
3.1 Les simples quotes	129
3.2 Le caractère \	131
3.3 Les guillemets	132
4. Récapitulatif des caractères spéciaux	132
5. Interprétation d'une ligne de commande	133
6. Écriture et lancement d'un script shell	134
6.1 Définition	134
6.2 Exécution d'un script par un shell enfant	135
6.3 Exécution d'un script par le shell courant	141
6.4 Commentaires	143
7. Variables réservées du shell	144
7.1 Les paramètres positionnels	144
7.2 La commande shift	146
7.2.1 Syntaxe	146
7.2.2 Principe	146

7.3	Code de retour d'une commande	148
7.3.1	La variable \$?.	148
7.3.2	La commande exit	149
7.4	Autres variables spéciales	150
7.4.1	PID du shell interpréteur	150
7.4.2	PID du dernier processus lancé en arrière-plan	151
8.	La commande read	153
8.1	Syntaxe	153
8.2	Lectures au clavier	153
8.3	Code de retour	155
8.4	La variable IFS	156
9.	Exécution de tests	157
9.1	Introduction	157
9.2	La commande test	157
9.2.1	Syntaxe	158
9.2.2	Tests sur les fichiers	158
9.2.3	Tests sur les chaînes de caractères	161
9.2.4	Tests sur les nombres	163
9.2.5	Les opérateurs	164
9.2.6	Exemple concret d'utilisation	165
9.3	La commande [[]]	166
10.	Les opérateurs du shell	170
10.1	Évaluation de l'opérateur &&	171
10.2	Évaluation de l'opérateur 	172
11.	L'arithmétique	173
11.1	La commande expr	173
11.1.1	Syntaxe	173
11.1.2	Opérateurs	173
11.2	La commande (())	177
11.2.1	Syntaxe	177
11.2.2	Utilisation	177
11.3	La commande let	180

8 — Programmation shell

sous Unix/Linux, sh, ksh, bash

11.4 Arithmétique sur les flottants	180
11.4.1 ksh93	180
11.4.2 Autres shells	181
12. Substitution d'expressions arithmétiques	182
13. Mise au point d'un script	183
13.1 Option -x	183
13.2 Autres options	186
14. Les structures de contrôle	187
14.1 if	187
14.2 case	191
14.2.1 Syntaxe	191
14.2.2 Principe	191
14.2.3 Utilisation	193
14.3 Boucle for	196
14.4 Boucle while	200
14.4.1 Syntaxe	200
14.4.2 Utilisation	200
14.4.3 Boucle infinie	201
14.5 until	204
14.5.1 Syntaxe	204
14.5.2 Utilisation	204
14.6 break et continue	208
15. Exercices	210
15.1 Variables, caractères spéciaux	210
15.1.1 Exercice 1 : variables	210
15.1.2 Exercice 2 : variables	210
15.1.3 Exercice 3 : substitution de commande	211
15.1.4 Exercice 4 : caractères de protection	211
15.2 Variables, affichages et lectures clavier	212
15.2.1 Exercice 1 : variables	212
15.2.2 Exercice 2 : paramètres positionnels	212
15.2.3 Exercice 3 : lectures clavier	212

15.3 Tests et arithmétique	213
15.3.1 Exercice 1 : tests sur des fichiers	213
15.3.2 Exercice 2 : tests de chaînes de caractères	213
15.3.3 Exercice 3 : tests numériques	214
15.3.4 Exercice 4 : arithmétique	214
15.3.5 Exercice 5 : opérateurs logiques des commandes [], [[]] et opérateurs logiques du shell	214
15.4 Structures de contrôle if, case, boucle for	215
15.4.1 Exercice 1 : les commandes [] et [[]], la structure de contrôle if	215
15.4.2 Exercice 2 : structures de contrôle case, boucle for	215
15.5 Boucles	216
15.5.1 Exercice 1 : boucle for, commande tr	216
15.5.2 Exercice 2 : boucle for, arithmétique	216
15.5.3 Exercice 3 : boucles for, while	217

Chapitre 5

Aspects avancés de la programmation shell

1. Comparatif des variables \$* et \$@	219
1.1 Utilisation de \$* et de \$@	219
1.2 Utilisation de "\$*"	221
1.3 Utilisation de "\$@"	222
2. Substitution de variables	223
2.1 Longueur de la valeur contenue dans une variable	223
2.2 Manipulation de chaînes de caractères	223
2.2.1 Retirer le plus petit fragment à gauche	224
2.2.2 Retirer le plus grand fragment à gauche	224
2.2.3 Retirer le plus petit fragment à droite	225
2.2.4 Retirer le plus grand fragment à droite	225
3. Tableaux	226
3.1 Assigner un élément	226
3.2 Référencer un élément	226

3.3	Assignment globale d'un tableau	227
3.4	Référencer tous les éléments d'un tableau	228
3.5	Obtenir le nombre d'éléments d'un tableau	228
3.6	Obtenir la longueur d'un élément d'un tableau	228
3.7	Tableaux associatifs	229
4.	Initialisation des paramètres positionnels avec set	230
5.	Les fonctions	230
5.1	Définition d'une fonction	231
5.2	Code de retour d'une fonction	233
5.3	Portée des variables	235
5.4	Définition de variables locales	236
5.5	Passage d'arguments	237
5.6	Exploiter l'affichage d'une fonction	240
5.7	Programme complet de l'exemple	241
6.	Commandes d'affichage	243
6.1	La commande print	243
6.1.1	Utilisation simple	243
6.1.2	Suppression du saut de ligne naturel de print	243
6.1.3	Afficher des arguments commençant par le caractère "-"	243
6.1.4	Écrire sur un descripteur particulier	244
6.2	La commande printf	245
7.	Gestion des entrées/sorties d'un script	246
7.1	Redirection des entrées/sorties standard	246
7.2	Gestion de fichiers	250
7.2.1	Ouverture de fichier	250
7.2.2	Lecture à partir d'un fichier	250
7.2.3	Écriture dans un fichier	250
7.2.4	Fermeture d'un fichier	251
7.3	Traitement d'un fichier	252
7.3.1	Informations préalables	252
7.3.2	Les différentes façons d'exploiter un fichier	253

7.3.3	Découper une ligne en champs	258
7.3.4	Modifier le séparateur de ligne.	260
8.	La commande eval	261
9.	Gestion des signaux	263
9.1	Principaux signaux	263
9.2	Ignorer un signal	264
9.3	Modifier le traitement associé à un signal	265
9.4	Repositionner le traitement par défaut du shell vis-à-vis d'un signal.	266
9.5	Utiliser trap à partir d'un script shell	267
10.	Gestion de menus avec select.	268
11.	Analyse des options d'un script avec getopt	270
12.	Gestion d'un processus en arrière-plan	276
13.	Script d'archivage incrémental et transfert sftp automatique.	278
13.1	Objectif	278
13.2	Le fichier uploadBackup.sh	281
13.3	Le fichier fonctions.inc.sh	283
14.	Exercices	287
14.1	Fonctions.	287
14.1.1	Exercice 1 : fonctions simples	287
14.1.2	Exercice 2 : fonctions simples, statut de retour	288
14.1.3	Exercice 3 : passage de paramètres, retour de valeur	289
14.1.4	Exercice 4 : fichiers	290
14.1.5	Exercice 5 : fichiers, fonctions, menu select	291
14.1.6	Exercice 6 : fichiers, tableaux associatifs (bash 4, ksh93)	292

Chapitre 6

Les expressions régulières

1.	Introduction	293
2.	Caractères communs aux ERb et ERe	294

12 _____ Programmation shell

sous Unix/Linux, sh, ksh, bash

3. Caractères spécifiques aux ERb	296
4. Caractères spécifiques aux ERe	297
5. Exploitation des expressions régulières par les commandes	299
5.1 La commande vi	299
5.2 La commande grep	299
5.3 La commande expr	302
5.4 sed et awk	305
6. Exercices	305
6.1 Expressions régulières.	305
6.1.1 Exercice 1 : expressions régulières avec vi	305
6.1.2 Exercice 2 : grep	306

Chapitre 7

La commande sed

1. Utilisation de la commande sed.	307
2. Exemples	310
2.1 Utilisation de sed en ligne de commande	310
2.1.1 La commande d (delete)	310
2.1.2 La commande p (print)	311
2.1.3 La commande w (write)	312
2.1.4 Négation d'une commande (!)	312
2.1.5 La commande s (substitution)	313
2.2 Script sed	314
3. Exercices	316
3.1 Expressions régulières.	316
3.1.1 Exercice 1 : insertion de balises dans un fichier	316
3.1.2 Exercice 2 : formatage de fichier	317

Chapitre 8

Le langage de programmation awk

1. Principe	319
1.1 Syntaxe	319
1.2 Variables spéciales	320
1.2.1 Variables prédéfinies dès le lancement de awk	320
1.2.2 Variables initialisées lors du traitement d'une ligne	321
1.2.3 Exemples simples	322
1.3 Critères de sélection	323
1.3.1 Expressions régulières	324
1.3.2 Tests logiques	325
1.3.3 Intervalles de lignes	326
1.4 Structure d'un script awk	326
1.4.1 BEGIN	326
1.4.2 Sections intermédiaires	326
1.4.3 END	326
1.4.4 Commentaires	326
1.4.5 Variables	327
1.4.6 Exemple	327
2. Opérateurs	328
3. La fonction printf	330
4. Redirections	331
5. Lecture de la ligne suivante : next	333
6. Structures de contrôle	334
6.1 if	334
6.2 for	335
6.3 while	336
6.4 do-while	336
6.5 break	336
6.6 continue	336
7. Terminer un script	337

14 _____ Programmation shell

sous Unix/Linux, sh, ksh, bash

8. Tableaux	337
8.1 Tableaux indicés par un entier	337
8.2 Tableaux associatifs	338
8.2.1 Définition	338
8.2.2 Tester l'existence d'un élément	340
8.2.3 Supprimer un élément	340
9. Les arguments de la ligne de commande	341
10. Fonctions intégrées	343
10.1 Fonctions travaillant sur les chaînes	343
10.2 Fonctions mathématiques	344
10.3 Autres fonctions	344
10.3.1 La fonction getline	344
10.3.2 La fonction close	348
10.3.3 La fonction system	349
11. Fonctions utilisateur	350
12. Exercices	352
12.1 awk en ligne de commande	352
12.1.1 Exercice 1 : awk et autres filtres	352
12.1.2 Exercice 2 : critères de sélection	352
12.1.3 Exercice 3 : critères de sélection, affichage de champs, sections BEGIN et END	353
12.2 Scripts awk	354
12.2.1 Exercice 4 : fonctions	354
12.2.2 Exercice 5 : analyse d'un fichier de log	355
12.2.3 Exercice 6 : génération d'un fichier d'étiquettes	357

Chapitre 9

Les commandes filtres

1. Introduction	359
2. Syntaxe d'appel des commandes filtres	359

3.	Visualisation de données	360
3.1	Consultation de données, création de fichiers : cat	360
3.2	Valeur des octets d'un flux de données : od	362
3.3	Filtrage de lignes : grep	363
3.4	Dernières lignes d'un flux de données : tail	366
3.5	Premières lignes d'un flux de données : head	367
3.6	Duplication de la sortie standard : tee	368
3.7	Numérotation de lignes : nl	369
3.8	Présentation d'un flux de données : pr	370
4.	Traitement de données	372
4.1	Comptage de lignes, de mots et de caractères : wc	372
4.2	Extraction de caractères : cut	374
4.3	Tri de données : sort	375
4.4	paste	378
4.5	split	379
4.6	Transformation de caractères : tr	381
4.7	Dédoublonnage : uniq	382
5.	Compressions, archivages et conversions	384
5.1	Compressions : gzip, bzip2	384
5.2	Archives tar	386
5.3	Archives cpio	388
5.4	Copie physique, transformations : dd	390
5.5	Changement d'encodage : iconv	392
6.	Commandes réseau sécurisées	393
6.1	Connexion à distance : ssh	393
6.2	Transfert de fichier : sftp	395
6.2.1	Commandes de sftp s'exécutant sur la machine locale	396
6.2.2	Commandes s'exécutant sur la machine distante	397
6.2.3	Commandes de transfert	398
6.2.4	Connexion automatique sans mot de passe	399

16 _____ Programmation shell

sous Unix/Linux, sh, ksh, bash

7. Autres commandes	401
7.1 La commande xargs	401
7.2 Comparer deux fichiers : cmp	403
7.3 Lignes communes à deux fichiers : comm	404

Chapitre 10 Solutions des exercices

1. Solutions du chapitre Mécanismes essentiels du shell	407
1.1 Fonctionnalités diverses	407
1.1.1 Exercice 1 : commandes internes et externes	407
1.1.2 Exercice 2 : génération de noms de fichiers	407
1.1.3 Exercice 3 : séparateur de commandes	409
1.2 Redirections	409
1.2.1 Exercice 1	409
1.2.2 Exercice 2	409
1.2.3 Exercice 3	410
1.2.4 Exercice 4	410
1.2.5 Exercice 5	410
1.2.6 Exercice 6	410
1.3 Tubes de communication	411
1.3.1 Exercice 1	411
1.3.2 Exercice 2	411
1.3.3 Exercice 3	411
1.3.4 Exercice 4	411
2. Solutions du chapitre Paramétrage de l'environnement de travail	412
2.1 Variables d'environnement	412
2.1.1 Exercice 1	412
2.1.2 Exercice 2	412
2.2 Alias de commande	413
2.2.1 Exercice 1	413
2.2.2 Exercice 2	414

3. Solutions du chapitre Les bases de la programmation shell.	415
3.1 Variables, caractères spéciaux	415
3.1.1 Exercice 1 : variables.	415
3.1.2 Exercice 2 : variables.	415
3.1.3 Exercice 3 : substitution de commande.	416
3.1.4 Exercice 4 : caractères de protection	416
3.2 Variables, affichages et lectures clavier	417
3.2.1 Exercice 1 : variables.	417
3.2.2 Exercice 2 : paramètres positionnels	418
3.2.3 Exercice 3 : lectures clavier.	419
3.3 Tests et arithmétique	420
3.3.1 Exercice 1 : tests sur des fichiers	420
3.3.2 Exercice 2 : tests de chaînes de caractères.	421
3.3.3 Exercice 3 : tests numériques.	422
3.3.4 Exercice 4 : arithmétique	423
3.3.5 Exercice 5 : opérateurs logiques des commandes [], [[]] et opérateurs logiques du shell	424
3.4 Structures de contrôle if, case, boucle for	425
3.4.1 Exercice 1 : les commandes [] et [[]], la structure de contrôle if.	425
3.4.2 Exercice 2 : structures de contrôle case, boucle for	426
3.5 Boucles.	427
3.5.1 Exercice 1 : boucle for, commande tr	427
3.5.2 Exercice 2 : boucle for, arithmétique.	428
3.5.3 Exercice 3 : boucles for, while	429
4. Solutions du chapitre Aspects avancés de la programmation shell. 430	
4.1 Fonctions.	430
4.1.1 Exercice 1 : fonctions simples	430
4.1.2 Exercice 2 : fonctions simples, statut de retour	432
4.1.3 Exercice 3 : passage de paramètres, retour de valeur . . .	433
4.1.4 Exercice 4 : fichiers	436
4.1.5 Exercice 5 : fichiers, fonctions, menu select	437

18 _____ Programmation shell

sous Unix/Linux, sh, ksh, bash

4.1.6 Exercice 6 : fichiers, tableaux associatifs (bash 4, ksh93)	439
5. Solutions du chapitre Les expressions régulières	440
5.1 Expressions régulières.	440
5.1.1 Exercice 1 : expressions régulières avec vi.	440
5.1.2 Exercice 2 : grep	441
6. Solutions du chapitre La commande sed.	442
6.1 Expressions régulières.	442
6.1.1 Exercice 1 : insertion de balises dans un fichier	442
6.1.2 Exercice 2 : formatage de fichier	443
7. Solution du chapitre Le langage de programmation awk	444
7.1 awk en ligne de commande	444
7.1.1 Exercice 1 : awk et autres filtres	444
7.1.2 Exercice 2 : critères de sélection.	445
7.1.3 Exercice 3 : critères de sélection, affichage de champs, sections BEGIN et END	445
7.2 Scripts awk	446
7.2.1 Exercice 4 : fonctions	446
7.2.2 Exercice 5 : analyse d'un fichier de log	448
7.2.3 Exercice 6 : génération d'un fichier d'étiquettes	450

Annexes

1. Caractères spéciaux du shell	453
2. Commandes internes au shell	454
3. Ordre d'interprétation d'une commande.	457
Index	459

Chapitre 3

Paramétrage de l'environnement de travail

1. Variables d'environnement

Les thèmes abordés dans ce chapitre permettront à l'utilisateur de paramétrer son environnement de travail en tenant compte du shell utilisé.

Un certain nombre de variables sont définies dans l'environnement du shell. Elles contiennent des informations nécessaires au fonctionnement de l'interpréteur et/ou des commandes lancées à partir de celui-ci.

1.1 Liste des variables

La commande **set** donne la liste des variables définies dans le shell courant.

Exemple

```
$ set
HOME=/home/christie
LOGNAME=christie
PATH=/usr/bin:/bin
PS1='$ '
PS2='> '
TERM=vt100
...
```

1.2 Affichage de la valeur d'une variable

Le caractère spécial **\$** du shell permet d'accéder au contenu d'une variable.

Exemple

```
$ echo $HOME
/home/christie
$
```

1.3 Modification de la valeur d'une variable

Le shell permet d'initialiser ou de modifier des variables.

Exemple

```
$ variable=valeur
$ echo $variable
valeur
$$
```

Si la valeur contient des caractères spéciaux du shell (\$, >, espace...), il faut empêcher le shell d'interpréter ceux-ci en entourant la valeur avec des simples quotes.

■ Remarque

Utiliser des simples quotes est l'une des trois manières de masquer des caractères en shell. Ce point sera détaillé ultérieurement.

Exemple

Le symbole ">" (redirection) doit être masqué, l'espace (séparateur de mots sur la ligne de commande) également :

```
$ variable='mot1 mot2 =>'
$ echo $variable
mot1 mot2 =>
$
```

■ Remarque

Il ne faut pas mettre d'espace autour du signe =. Le shell ne comprendrait pas qu'il s'agit d'une affectation.

1.4 Principales variables

Les variables présentées ci-dessous possèdent une valeur au niveau du shell de connexion. D'autres variables peuvent être définies ultérieurement.

1.4.1 HOME

Cette variable contient la valeur du répertoire d'accueil de l'utilisateur. Elle ne doit pas être modifiée.

1.4.2 PATH

La variable PATH contient une liste de répertoires qui sont explorés par le shell lorsqu'il doit lancer une commande externe.

■ Remarque

En aucun cas, une commande n'est recherchée dans le répertoire courant si celui-ci ne figure pas dans la variable PATH.

Exemples

```
$ echo $PATH
/usr/bin:/bin
$
```

La commande `date` est trouvée :

```
$ date
Tue Jan 28 17:51:23 MET 2014
$
```

En effet, elle se situe dans le répertoire `/usr/bin` :

```
$ find / -name date 2> /dev/null
/usr/bin/date
$
```

82 — Programmation shell

sous Unix/Linux, sh, ksh, bash

La commande **ping** n'est pas trouvée :

```
$ ping localhost
ksh: ping: not found
$
```

La commande est située dans le répertoire **/usr/sbin** qui n'est pas cité dans la variable **PATH** :

```
$ find / -name ping 2> /dev/null
/usr/sbin/ping
$
```

Le répertoire courant n'est pas exploré s'il n'est pas cité dans **PATH** :

```
$ cd /usr/sbin
$ ping localhost
ksh: ping: not found
$
```

Modifier le contenu de la variable **PATH** :

```
$ PATH=$PATH:/usr/sbin
$ echo $PATH
/usr/bin:/bin:/usr/sbin
$
```

La commande **ping** est trouvée :

```
$ ping localhost
localhost is alive
$
```

Rechercher une commande dans le répertoire courant

Pour qu'une commande soit recherchée dans le répertoire courant, il faut ajouter en fin de variable **PATH** la chaîne **":"** ou simplement le caractère **"."**.

Exemple

```
PATH=/usr/bin:/usr/local/bin:/home/christie/bin:.
```

Équivalent à :

```
PATH=/usr/bin:/usr/local/bin:/home/christie/bin:
```

1.4.3 PWD

ksh	bash
-----	------

Cette variable contient la valeur du répertoire courant. Elle est mise à jour par le shell dès que l'utilisateur change de répertoire. Cette variable peut être utilisée en ksh pour faire apparaître la valeur du répertoire courant dans le prompt.

1.4.4 PS1

Cette variable contient la chaîne de caractères représentant le prompt principal.

Exemple

```
$ echo $PS1
$
$ PS1='Entrez une commande => '
Entrez une commande => date
Thu Jan 30 17:27:51 MET 2014
Entrez une commande =>
```

Avec le ksh et le bash, il est possible de paramétrer son prompt de telle façon qu'il contienne en permanence la valeur du répertoire courant.

Faire apparaître la valeur du répertoire courant dans le prompt en ksh

Il faut se servir de la variable PWD.

Exemple

Ici, le prompt est composé de deux caractères : le symbole "\$" suivi d'un espace (cf. Figure 1) :

ksh

Variable	Valeur
PS1	\$
PWD	/home/christie
....	

Figure 1 : Initialisation de PS1 avec le répertoire courant (1)

```
$
$ echo -$PS1-
-$ -
$
```

Le répertoire courant est **/home/christie** :

```
$
$ pwd
/home/christie
$
```

Initialisation de PS1 avec la chaîne de caractères '\$PWD\$' ; il faut empêcher le shell de substituer \$PWD par sa valeur au moment de l'affectation, donc il faut protéger l'expression avec des quotes (cf. Figure 2) :

```
$ PS1='$PWD$─'
```