BASH CHEAT SHEET

Informations générales

- Les mots-clés entre chevrons (<mot-clé>) représentent des paramètres ou options obligatoires
- Les mots-clés entre crochets ([mot-clé]) représentent des paramètres ou options facultatives
- Pour chaque commande, une liste des options non exhaustive est proposée. Ces options sont les versions raccourcies. Voir le “man” pour plus d’information.

Redirections d’entrée/sortie

`commande < fichier`	redirige le fichier comme entrée standard
`commande > fichier`	redirige la sortie standard vers un nouveau fichier
`commande >> fichier`	redirige la sortie standard et l’ajoute à un fichier
`commande 2> fichier`	redirige la sortie d’erreur vers un nouveau fichier
`commande 2>> fichier`	redirige la sortie d’erreur et l’ajoute à un fichier
`commande &> fichier`	redirige toutes les sorties vers un nouveau fichier
`commande &>> fichier`	redirige toutes les sorties et l’ajoute à un fichier
`commande 1>&2`	redirige la sortie standard dans la sortie d’erreur
`commande 2>&1`	redirige la sortie d’erreur dans la sortie standard

Remarque : il est possible de cumuler plusieurs redirections
commande > fichier.out 2> fichier.err
commande < fichier.in > fichier.out 2> fichier.err

Pipes

commande1 | commande2	redirige la sortie standard de la commande 1 vers la commande 2
commande1 |& commande2	redirige la sortie d’erreur de la commande 1 vers la commande 2

Répertoires

Répertoires spéciaux :

`.`	répertoire courant
`..`	répertoire parent
`/`	répertoire racine ou séparateur de chemins
`~`	répertoire personnel de l’utilisateur
`pwd`	affiche le répertoire courant
`cd [dir]`	change le répertoire courant - sans paramètre, cd retourne au répertoire personnel - paramètre spécial “-” qui permet de retourner au répertoire précédent
`ls [-1|l|a|R|r] [dir]`	liste le contenu d’un répertoire
`mkdir [-p] <dir>`	crée un ou des répertoires
`rmdir [-p] <dir>`	efface un ou des répertoires vides

Fichiers et répertoires

`cp [-r|i|u] <src> <dest>`	copie la source vers la destination
`mv [-r|i|u] <src> <dest>`	déplace (ou renomme) la source vers la destination
`rm [-r|f] <fichier>`	efface le fichier ou répertoire

Permissions sur les fichiers et répertoires

`rwxrwxrwx` `(u) (g) (o)`	r = lecture w = écriture x = exécution u = propriétaire g = groupe o = les autres
`chmod <perm> <fichier>`	met à jour les permissions du fichier
`chown [util.][:[groupe>]` `<fichier>`	met à jour le propriétaire et/ou le groupe du fichier

Opérations sur les fichiers

`touch <fichier>`	met à jour les dates d’accès et de modification d’un fichier créé le fichier s’il n’existe pas
`cat [-e] <fichier>`	affiche le contenu d’un fichier
`echo [-e] <texte>`	affiche le texte

`cut`
`[-f <champ> [-d <del>]]`
`[-c [N][[-][M]]`
`<fichier>`
	découpe les lignes en sections

`sort [-r|n|f] <fichier>`	trie les lignes
`head [-n <n>] <fichier>`	affiche les n premières lignes
`tail [-f] [-n <n>]` `<fichier>`	affiche les n dernières lignes
`uniq [-c] <fichier>`	supprime les lignes dupliquées
`grep [[-e] <pattern>]` `[-v] <fichier>`	filtre les lignes
`paste [-s] [-d]` `<fichier(s)>...`	fusionne les lignes d’un ou plusieurs fichiers
`wc [-w|c|l] <fichier>`	compte le nombre de lignes, caractères ou mots
`tr [-d] <set1> <set2>`	transpose ou élimine des caractères

`split [-l|b]`
`[--filter=<commande>]`
`<fichier>`
	découpe un fichier

`find`
`[-[i]name <fichier>]`
`[-[a|c|m]time <fichier>]`
`[-[a|c|m]newer <fichier>]`
	trouve un fichier dans un répertoire

BASH & SCRIPTING CHEAT SHEET

Wildcards (voir "man glob")

`*`	remplace 0 ou plusieurs caractères
`?`	remplace 1 seul caractère
`[]`	remplace 1 caractère parmi l'ensemble spécifié
`{}`	génére une liste de l'ensemble spécifié

Peuvent remplacer ou compléter `<fichier>`.

Processus

`top / htop`	Affiche la liste des processus par ordre décroissant d'utilisation CPU (par défaut) Rafraichi automatiquement toutes les 2 secondes
`ps [w|a|x|u]`	Affiche la liste des processus en cours d'exécution
`CTRL-C`	arrête un processus
`CTRL-Z`	stoppe un processus (mais ne l'arrête pas)
`fg`	reprend l'exécution d'un processus en avant-plan
`bg`	reprend l'exécution d'un processus en arrière-plan
`nice`	exécute une commande avec une priorité différente de la priorité par défaut
`renice`	change la priorité d'un processus en cours d'exécution

Exécution

`$(<cmd>)` ou `` `<cmd>` ``	Exécute la commande <cmd> et place le résultant dans CWD

Exemple:

`CWD=`pwd``	place le résultat de pwd dans la variable CWD

Variables

`NAME="Samuel"`	Assigner une variable
`echo $NAME`	Afficher/utiliser le contenu
`echo "$NAME"`	d'une variable (ie: fonctionne
`echo "${NAME}!"`	avec toute
commande)	
`echo '$NAME!'`	Le contenu de la chaîne est affiché sans traitement ou interprétation par Bash

Boucles

Boucle FOR :

`for <var> in <ens>; do`	boucle sur l'ensemble ENS
`  <cmd>`	dans la variable VAR
`done`	

Exemples:

`for file in /bin/*; do`	liste les fichiers du répertoire
`  echo $file`	/bin et affiche un à un leurs
`done`	noms
`for i in {1..20}; do`	compte et affiche de 1 à 20
`  echo $i`	(voir le glob "range" `{}`)
`done`	
`for i in {1..20..2}; do`	compte et affiche de 1 à 20
`  echo $i`	par pas de 2
`done`	

Boucle WHILE:

`while <condition>; do`	boucle tant que la condition
`  <cmd>`	est vraie
`done`	

Exemples:

`while true; do`	boucle infinie affichant "infini"
`  echo "infini"`	(ctrl-c pour
interrompre)	
`done`	
`i=1`	compte et affiche de 1 à 20
`while [[ $i -le 20 ]]; do`	
`  echo $((i+=1))`	
`done`	

Fonctions

`<nom-de-fonction>() {`	Déclare une fonction
`  <cmd(s)>`	Code de la fonction
`  return <valeur>`	Retour (facultatif)
`}`	
`<nom-de-fonction>()`	Appel de la fonction

Conditions

`if <condition1>; then`	Si la condition1 est vraie
`  <cmd(s)>`	exécute la/les commande(s)
`elif <condition2>; then`	Sinon si la condition2 est
`  <cmd(s)>`	vraie exécute la/les commande(s) (facultatif)
`else`	Sinon exécute la/les
`  <cmd(s)>`	commande(s) (facultatif)
`fi`	

Les conditions suivantes renvoient VRAI si :

Chaînes de caractères :

`[[ -z STR ]]`	chaîne vide
`[[ -n STR ]]`	chaîne non vide
`[[ STR1 == STR2 ]]`	chaînes égales
`[[ STR1 != STR2 ]]`	chaînes différentes

Nombres :

`[[ NB1 -eq NB2 ]]`	nombres égaux
`[[ NB1 -ne NB2 ]]`	nombres différents
`[[ NB1 -lt NB2 ]]`	NB1 < NB2
`[[ NB1 -le NB2 ]]`	NB1 <= NB2
`[[ NB1 -gt NB2 ]]`	NB1 > NB2
`[[ NB1 -ge NB2 ]]`	NB1 >= NB2

Fichiers :

`[[ -e STR ]]`	fichier existe
`[[ -s STR ]]`	fichier non vide
`[[ -r STR ]]`	droits en lecture (r--)
`[[ -w STR ]]`	droits en écriture (-w-)
`[[ -x STR ]]`	droits en exécution (--x)
`[[ -f STR ]]`	fichier
`[[ -d STR ]]`	répertoire

Opérateurs booléens :

`[[ ! condition ]]`	Not
`[[ cond1 ]] && [[ cond2 ]]`	And
`[[ cond1 ]] || [[ cond2 ]]`	Or

Case

`case "$<variable>" in`	Teste la <variable>
`  <val1> [|val2 …])`	Si c'est égal à une ou plusieurs valeurs
`    <cmd(s)>`	exécute les commandes
`    ;;`	fin de bloc
`  *)`	bloc par défaut
`    <cmd(s)>`	exécute les commandes
`    ;;`	fin de bloc
`esac`	fin de case