UNIVERSITE CHEIKH ANTA DIOP DE DAKAR

ECOLE SUPERIEURE POLYTECHNIQUE

DEPARTEMENT INFORMATIQUE

DIC 1 && LICENCE
LANGAGE C
TP N° 0

Navigation UNIX

ls [options] [chemin]

ls : liste le contenu d'un répertoire

Options communs de ls :
-a, –all : Affiche les fichiers cachés
-l : Affichage au format long (presque tous les champs d'inode)
-h, –human-readable : Affiche les tailles des fichiers de manière adaptée (K pour Kilobyte, M pour Megabyte et G pour Gigabyte)

La sortie de ls associe à chaque type de fichier une lettre
- : Les fichiers ordinaires (tous les fichiers courants, .txt, .doc, .mp3 .jpeg ou sans extensions)
c ou b : Les fichiers spéciaux (fichiers associés aux périphériques, localisés dans le repertoire /dev/)
d : Les répertoires
l : Les liens, un lien permet d'accéder à un même fichier ou répertoire en utilisant un nom différent.
cd : Changedir
La commande cd (change directory) permet de changer le répertoire de travail du Shell, naviguer dans le système de fichier
cd /
> pwd
> cd ~
> pwd
/home/toto
> cd public; pwd
> cd -
/home/toto
> cd .. ; pwd
/home
> cd . ; pwd
La commande touch permet de : Créer un fichier vide
> touch hello_word_file
> ls -l hello_word_file
-rw-r–r– 1 toto toto 0 2011-08-24 18:24 hello_word_file
ou de mettre à jour la date du dernier accès et de la dernière modification
> ls -l hello_word_file
-rw-r–r– 1 toto toto 0 2011-08-24 18:24 hello_word_file
> touch hello_word_file
> ls -l hello_word_file
-rw-r–r– 1 toto toto 0 2011-08-24 18:27 hello_word_file
cp source destination
Quelques options intéressantes :
-a --archive : copie avec préservation des attributs des fichiers et répertoires.
-i, --interactive : demande avant d'écraser la destination
-R, -r, --récursive : copie récursive (de fichiers et répertoires)
-u, --update : Copie seulement si le nouveau fichier est plus récent que la destination
-v, --verbose : Affiche ce qui est en cours de copie
Copie interactive de fichier (verbose)
> cp -iv nom_fichier_original nom_fichier_destination
Copie complète d'un repertoire (écrase les fichiers sans demander)
> cp -var repertoire nom_destination
La commande mv permet de déplacer un fichier le renommage sous Linux qui est le déplacement sur un autre nom dans le même répertoire ;

Déplacement de fichier
> mv file_1 file_2 ... repertoire-destination

Renommage de fichiers
> mv nom-originale-de-fichier nouveau-nom
Déplacement de fichier
mv unfichier.txt unrepertoiredestination/
dirname : permet de récupérer le répertoire dans le chemin complet d'un fichier
dirname : recupère au contraire le nom du fichier
pwd (print working directory) : affiche le repertoire courant
dirname /usr/include/linux/socket.h
/usr/include/linux
> basename /usr/include/linux/socket.h
socket.h
> pwd
/home/toto
La commande rm(remove) permet de supprimer des fichiers ou répertoires
Suppression de fichier ou reperptoire vide
> rm [-i] nom-fichier-ou-repertoire-vide
> rmdir nom-repertoire-vide
Suppression recursive de repertoire non vide
> rm [-i] -rf nom-repertoire
> rmdir –ignore-fail-on-non-empty nom-repertoire
Création d'un repertoire (chemin relatif)
> mkdir un-repertoire un-autre-repertoire un-troisieme-repertoire
Création d'un repertoire (chemin relatif)
> mkdir un-repertoire un-autre-repertoire un-troisieme-repertoire
Créeation d'un repertoire avec un chemin complet
> mkdir /home/toto/Documentes/un-repertoire
Création d'un repertoire (chemin relatif)
> mkdir un-repertoire un-autre-repertoire un-troisieme-repertoire
Créeation d'un repertoire avec un chemin complet
> mkdir /home/toto/Documentes/un-repertoire
Créeation d'un repertoire et créer les repertoires parent s'ils n'existent pas
> mkdir -p /home/toto/Documentes/ancetre/grand-pere/pere/un-repertoire
Création d'un repertoire (chemin relatif)
> mkdir un-repertoire un-autre-repertoire un-troisieme-repertoire
Créeation d'un repertoire avec un chemin complet
> mkdir /home/toto/Documentes/un-repertoire
Créeation d'un repertoire et créer les repertoires parent s'ils n'existent pas
> mkdir -p /home/toto/Documentes/ancetre/grand-pere/pere/un-repertoire
Création d'un repertoire avec spécification des droits
> mkdir -m 755 repertoire
e type des fichiers sous Linux n'est pas determiné par l'extension mais le magic
number
La commande stat affiche le type de fichier en se basant sur le magic number
Gestion des droits
Chaque fichier appartient à un utilisateur et à un ou plusieurs groupes
Chaque utilisateur peut avoir des droits (permissions) de lecture, ecriture et d' execution sur un fichier
Chaque fichier définit des permissions pour trois classes d'utilisateurs :
(u) Utilisateur propriétaire (créateur du fichier)
(g) Groupe propriétaire
(o) Les autres utilisateurs
ls -lh fichier

Compilation

Programmer en C c'est mettre du texte avec le jargon C et utiliser un compilateur pour produire un exécutable

1. Préprocesseur

Interprétation et transformation des directives (#include, #define, ...)
Produit un fichier texte

2. ***Compilation***

Traduit le résultat du préprocesseur en instructions machines (code objet) non prêt pour l'exécution ;

3. ***L'assemblage et la liaison des objets***

Le Linker combine le code objet produit par le compilateur avec les autres bouts de code nécessaires pour crée un exécutable. Exemple d'autres codes : le code de ***printf***

Outil GCC

Le compilateur C utilisé sous Linux est gcc (Gnu Compiler Collection). On peut également l'invoquer sous le nom cc, comme c'est l'usage sous Unix, ou g++ si on compile du code C++.

Différentes façons de compiler ...

Compilation 1

gcc monProg.c : produit un fichier exécutable nommé *a.out*

Compilation 2

gcc monProg.c -o monExe : le fichier exécutable a un nom « *monExe* »

Compilation 3

Etape par étape

1- Preprocesseur

$ gcc -E hello.c -o hello.cpp

2- Production du code objet

$ gcc -x cpp-output -c hello.cpp -o hello.o

3- Création de l'executable (edition des liens)

$ gcc hello.o -o hello

Contrôle de la compilation*

-S : Arrêt de la compilation après production de code Assembleur et produit un fichier assembleur «.S »

-E : Arrêt de la compilation après le préprocesseur et Le produit est affiché sur la sortie standard

-X : Permet de spécifier le langage compilé

-v (en minuscule) : Affiche toutes les commandes exécutées par le compilateur

Autres options

Détection de potentielles erreurs de programmation (Warnings)

-pedantic

– Enlève toutes les extensions non conformes au standard (respect strict à l 'ISO)

-Wformat

– Vérifie si tous les appels de printf et scanf ne contiennent pas d'erreurs de typage

-Wformat-security

– Vérifie l'existence de failles dans l'appel des fonctions printf et scanf

• Et bien d'autres (*man gcc*)

-Wall, -Wunreachable-code, -Wunused , werror...

Le compilateur gcc utilise des conventions sur les suffixes des fichiers pour savoir quel utilitaire invoquer lors des différentes phases de compilation. Ces conventions sont les suivantes :

Suffixe	Produit par	Rôle
.c	Programmeur	Fichier source C, sera transmis à cpp, puis à cc1.
.cc ou .C	Programmeur	Fichier source C++, sera transmis à cpp, puis à cc1plus.
.m	Programmeur	Fichier source Objective C, sera transmis à cpp, puis à cc1obj.
.h	Programmeur	Fichier d'en-tête inclus dans les sources concernées. Considéré comme du C ou du C++ en fonction du compilateur invoqué (gcc ou g++).
.i	cpp	Fichier C prétraité par cpp, sera transmis à cc1.
.ii	cpp	Fichier C++ prétraité par cpp, sera transmis à cc1plus.
.s ou .S	cc1, cc1plus, cc1obj	Fichier d'assemblage fourni par l'un des compilateurs cc1, va être transmis à l'assembleur as.
.o	as	Fichier objet obtenu après l'assemblage, prêt à être transmis à l'éditeur de liens ld pour fournir un exécutable.
.a	ar	Fichier de bibliothèque que l'éditeur de liens peut lier avec les fichiers objet pour créer l'exécutable.

La plupart du temps, on invoque simplement gcc en lui fournissant le ou les noms des fichiers source, et éventuellement le nom du fichier exécutable de sortie, et il assure toute la transformation nécessaire. L'invocation de gcc se fait donc avec les arguments suivants :

Option	Argument	But
-E		Arrêter la compilation après le passage du préprocesseur, avant le compilateur.
-S		Arrêter la compilation après le passage du compilateur, avant l'assembleur.
-c		Arrêter la compilation après l'assemblage, laissant les fichiers objet disponibles.
-W	Avertissement	Valider les avertissements (*warnings*) décrits en arguments. Il en existe une multitude, mais l'option la plus couramment utilisée est `-Wall`, pour activer tous les avertissements.
-pedantic		Le compilateur fournit des avertissements encore plus rigoureux qu'avec `-Wall`, principalement orientés sur la portabilité du code.
-g		Inclure dans le code exécutable les informations nécessaires pour utiliser le débogueur. Cette option est généralement conservée jusqu'au basculement du produit en code de distribution.
-O	0 à 3	Optimiser le code engendré. Le niveau d'optimisation est indiqué en argument (0 = aucune). Il est déconseillé d'utiliser simultanément l'optimisation et le débogage.

Exemple d'application

```
#include <stdio.h>

main()
{

      int op1 = 134, op2 = 275;
      int somme;
      printf("%d+%d=%d\n", op1, op2, somme);

}
```