Projet « mydiff »

ESIEA- Mastère SI&S

27 novembre 2012

Version : 1.00

1 Objectif du projet

L'objectif de ce projet est d'écrire un script shell « mydiff.sh » réalisant la comparaison de tous les fichiers et répertoires de deux sous-arborescences.

2 Description du projet

2.1 Fonctionnalités

Le script doit prendre un ensemble de paramètres :

« -s » : répertoire source

« -d » : répertoire destination

« -c » : liste de drapeaux indiquant les types de comparaisons à effectuer :

« d » : comparaison « diff » : utilise la commande « diff » pour comparer les deux entités

« m » : comparaison « md5 » : compare les checksums md5 des deux entités

« p » : comparaison « permission » : compare les permissions, utilisateur et groupe des deux entités

« t » : comparaison « time » : compare la date de dernière modification des deux entités

Pour les répertoires, seule la comparaison « permission » est effectuée.
Il est possible de combiner plusieurs comparaisons en positionnant une liste de drapeaux. Par exemple « -c "d m p" » effectuera :

pour chaque sous-répertoire, un test d'existence et la comparaison « permission »

pour chaque fichier, un test d'existence puis les comparaisons « diff », « md5 » et « permission » Si l'un des tests retourne une différence, les entités sont considérées comme différentes.

« -e » : liste d'items é ne pas considérer. Par exemple « -e "\.txt$ logs" » permettra de ne pas tenir compte des entités se terminant par « .txt » et de celles contenant le mot clef « logs » (la recherche portant sur le chemin complet de l'entité et non pas juste sur son nom)

« -f » : filtre de fichiers. Par exemple « -f "*.txt" » indiquera au script de ne comparer que les fichiers se terminant par « .txt » (les répertoires étant eux tous comparés)

« -v » : définit le niveau de log. Quatre valeurs sont possibles :

0 : n'affiche rien

1 : affiche le nom des entités différentes

2 : affiche le détail des différences

3 : affiche le détail de tous les tests

« -S » : effectue la synchronisation des sous-arborescences. Le script effectue alors les tests de comparaison en fonction des critéres demandés ; si deux entités sont déclarées différentes, l'entité du répertoire destination est synchronisée á partir celle du répertoire source :

pour un répertoire, synchronisation de l'utilisateur, du groupe et des permissions

pour un fichier, synchronisation de l'utilisateur, du groupe, des permissions et du contenu
A noter que la synchronisation de la derniére date de modification ne sera pas effectuée.

2.2 Langage

Le programme est é développer exclusivement en script shell (bash) et devra utiliser les commandes shell standards. L'utilisation d'autres langages de script comme perl, python ou ruby est interdite (méme en inlinant une simple commande) : le script devra

pouvoir s'exécuter sur une machine ne comportant pas d'interpréteur pour ces langages.

2.3 Structure du script

Le script sera basé sur le squelette suivant :

```
#!/bin/sh

function PrintUsage {
  echo "
Usage : ./mydiff.sh -s <src dir> -d <dst dir> [-c <comparison flags>] [-e <skip
     items>] [-f <file filter>] [-v <verbose level>] [-S] [-h]

    -h                      = display this help
    -s <src dir>            = source directory
    -d <dst dir>            = destination directory
    -c <comparison flags>   = properties to compare :
         d = diff
         m = md5
         p = permission (and user/group)
         t = last modification date
         Use a list of flags separated by a space; e.g. : \"d p m\"
         Default : \"d\"
    -e <skip items>         = list of items to skip
         e.g. : -e \"\.txt\$ logs\"
    -f <file filter>        = file pattern to find
         e.g. : -f \"*.txt\"
    -v <verbose level>      = verbose level
         0 = display nothing
         1 = display entity name if different
         2 = display differences details
         3 = display all tests
         Default : \"1\"
    -S                      = synchronize dst from src
"
}

function PrintMsg {
  level=$1
  msg=$2
  if [ $level -le $VERBOSE_LEVEL ]
  then
    echo $msg
  fi
}

#
    ============================================================================

# Comparison functions
#
    ============================================================================
```

```
function DoCompare {
  res=0
  src=$1
  dst=$2

  # TO DO...

  return $res
}

function DoSynchronize {
  src=$1
  dst=$2

  # TO DO...

}

#
    ================================================================================

# Beginning of script
#
    ================================================================================


############################## ARGUMENTS DEFAULT VALUES
    ################################

DIRPATH_SRC=""
DIRPATH_DST=""
COMP="d"
COMP_DIFF=1
COMP_MD5=0
COMP_PERM=0
COMP_DATE=0
SYNCHRONIZE=0
EXCLUDE=""
FILTER=""

VERBOSE_LEVEL=1
VERBOSE_LEVEL_ERROR=0
VERBOSE_LEVEL_DIFF=1
VERBOSE_LEVEL_DIFF_DETAIL=2
VERBOSE_LEVEL_ALL=3

############################## ARGUMENTS ANALYSIS
    ################################

# TO DO...

cd $DIRPATH_SRC
for dirname in `find . -type d`
do
  # TO DO...
done

for filename in ...
```

```
do
  # TO DO
done
```

Fonction « PrintUsage »
Cette fonction permet d'afficher l'aide du script.
Fonction « PrintMsg »
Cette fonction permet d'afficher un message de log si le niveau du message est inférieur ou égal é celui fixé par l'utilisateur.
Fonction « DoCompare »
Cette fonction effectue la comparaison entre deux entités fournies en paramétre. Elle retourne 0 si les entités sont identiques, 1 dans le cas contraire.
Fonction « DoSynchonize »
Cette fonction effectue la synchronisation entre deux entités fournies en paramétre.
Variables
Cette partie contient l'initialisation des variables utilisées dans mon script. Elle n'est fournie qu'é titre indicatif et peut bien sur étre modifiée.

Elle contient également l'initialisation de la variable « VERBOSE_LEVEL » représentant le niveau de trace (par défaut é « 1 ») ainsi que la définition des quatre niveaux de traces :

VERBOSE_LEVEL_ERROR

VERBOSE_LEVEL_DIFF

VERBOSE_LEVEL_DIFF_DETAIL

VERBOSE_LEVEL_ALL

Les boucles for
La fin du script est constituée de deux boucles : la premiére parcourt l'ensemble des sous-répertoires et la seconde l'ensemble des fichiers. Ces boucles devront appeler les fonctions « DoCompare » et « DoSynchonize » pour effectuer les opérations demandées.

2.4 Quelques pistes...

Voici une liste de quelques commandes qui pourraient bien vous étre utiles : diff, sdiff, md5sum, stat, cp, mkdir, chown, chmod, getopts, find et les tests "-z, -e, -d, . . . "

2.5 Tests de votre script

Une archive tar « test_mydiff.tar » contenant deux sous-arborescences « src » et « dst » sera fournie la semaine prochaine. Ces deux sous-arborescences contiennent un ensemble de répertoires et fichiers avec de légéres différences. Elles vous permettront de tester votre script.

3 Cadre du projet

3.1 Réalisation du projet

Le projet est é faire de maniére individuelle. L'utilisation de code copié/collé depuis le web est é proscrire.

3.2 Evaluation du rendu

Votre script sera évalué sur les critéres suivants :

fonctionnalités : le script doit exécuter les opérations demandées

propreté du code : le code devra étre lisible, commenté et utiliser des fonctions. Il devra se baser sur le squelette fourni.

sortie du script : la sortie du script devra étre au maximum identique é celle du script de référence. Pour cela, vous pourrez vous aider des fichiers de sortie générés par mon script sur les arborescences de l'archive « test_mydiff.tar ».

3.3 Date du rendu

Le script (et uniquement le script) devra être envoyé par mail sur « erra@esiea.fr » avant le dimanche 2 décembre 2012 23h59.
ATTENTION : suivant le principe « one shot » seul le premier envoi sera considéré. Inutile donc d'envoyer deux scripts.