TP SVN

Préambule

Ce TP a pour but de découvrir les principales commandes SVN.
Il sera réalisé en groupes (binômes voire trinômes s'il y a un nombre impair d'étudiants). Chaque groupe travaillera sur un projet qui lui sera dédié (projet1, projet2, ….) et dont le dépôt est accessible via un serveur.
Chaque utilisateur aura un rôle au sein du groupe (dev1, dev2).
Notes :
- Il est très important d'ajouter des commentaires utiles lors des propagations (commit) svn
- Accès au dépôt : svn+ssh://<user>@<server>/var/svn/projet<X> avec
  - projet<X> est le projet de chaque groupe ; il est déjà créé sur le serveur. <X> prend les valeurs 1, 2, 3, ...
  - <user> : sera le nom de l'utilisateur linux (le login) sur le serveur
    → Note : les étudiants enregistrés ont un login sur le serveur et tous les PC
  - <server> : l'adresse IP du serveur (qui sera donné en TP)

Outils à utiliser

2 possibilités
- Le 'terminal' linux (à privilégier si on a quelques connaissances concernant les commandes de base : ls, pwd, cd, mkdir, mv, rm, ...)
  - les commandes svn seront lancées 'en ligne de commande' ;
  - pour éditer un fichier, on utilisera un des éditeurs de texte disponibles (gedit, kate, ...).
- Un gestionnaire de fichier qui intègre un plugin pour svn ; par exemple dolphin
  - Toutes les opérations sur les fichiers et répertoires (édition, ajout, renommage,…) et les commandes svn seront effectuées via ce gestionnaire.

Note : les indications données tout au long du sujet sont à destination des utilisateurs du terminal linux. Les autres lanceront ces commandes indirectement (via une option de dolphin).

1 Prise en main

1.1 Récupération du projet depuis le référentiel (checkout)

- dev1 et dev2
  - Création d'un répertoire tp-svn (et aller dans ce répertoire)
  - création d'une copie de travail :
    svn checkout svn+ssh://user<XY>@<server>/var/svn/projet<X>

1.2 Modification d'une copie de travail et mise à jour sur le serveur (commit)

- dev2 :
  - édite le fichier data/prenoms.txt
  - ajoute un prénom féminin
  - met à jour cette modification sur le serveur (propage ses modifications) : *svn commit*

1.3 Récupération de la dernière version (update)

- dev1 :
  - met à jour sa version avec celle du dépôt : *svn update*

Vérifier alors que l'ajout de dev2 est intégré à la dernière version

1.4 ça marche dans les 2 sens !

- dev1 :
  - édite le fichier data/prenoms.txt
  - ajoute un prénom masculin
  - met à jour cette modification sur le serveur (propage ses modifications)
- dev2 :
  - met à jour sa version avec celle du dépôt

2 Etat des fichiers, historique : status et log

- dev1 :
  - édite le fichier data/prenoms.txt et ajoute un prénom masculin
  - quel est l'état de sa copie de travail (quelles sont les modifications locales) ? *svn status*
  - propage ses modifications
  - affiche l'historique
    - du fichier data/prenoms.txt : *svn log*
    - du projet
- dev2 :
  - met à jour sa copie de travail (Vérifier que le travail de dev1 est bien intégré au projet)
  - édite le fichier data/prenoms.txt et ajoute un prénom féminin
  - quel est l'état de sa copie de travail (quelles sont les modifications locales) ? *svn status*
  - propage ses modifications
  - affiche l'historique
    - du fichier data/prenoms.txt : *svn log*
    - du projet
  - dev1
    - met à jour son le projet

3 Travail collaboratif

3.1 Modification de fichiers différents

- dev1 :
  - édite, modifie le fichier data/prenoms.txt
  - propage (et met éventuellement à jour) sa version
  - refait (au moins une fois) les 2 étapes précédentes
- dev2:
  - édite, modifie le fichier data/mois.txt
  - propage (et met éventuellement à jour) sa version (plusieurs fois)

Vérifier que les mises à jour se passent bien, que le travail de chacun est bien intégré au projet.

3.2 Modification d'un fichier sans conflit

- dev1 et dev2 doivent avoir une version à jour
- dev1 : modifie un prénom féminin dans data/prenoms.txt, met à jour, propage sa version
- dev2 : modifie un prénom masculin dans data/prenoms.txt, met à jour, propage sa version

Note : l'ordre des mises à jour, propagations ne doit pas avoir d'importance.
Que s'est-il passé ? Quelle information a t-on eu lors des mises à jour ?
Vérifier que les modifications apportées dans un même fichier par 2 utilisateurs ne posent pas de problème.
On peut regarder l'historique de ce fichier

3.3 Modification d'un fichier avec conflit

- dev1 et dev2 doivent avoir une version à jour
- dev1 : modifie le 1er prénom féminin dans data/prenoms.txt, met à jour, propage sa version
- dev2 : modifie le 1er prénom féminin dans data/prenoms.txt, met à jour, propage sa version

Que se passe-t-il ?
Notes :

- choisir l'option 'p' (pour résoudre le conflit plus tard)
- Quels sont les différents fichiers créés ? (prenoms.txt.extension) ?
- Les éditer pour comprendre

- Résoudre le conflit "à la main" (en éditant le fichier)
- Indiquer qu'il est résolu ? *svn resolved data/prenoms.txt*
- Propager

4 Ajouter, supprimer, renommer un fichier, un répertoire : add, mv, mkdir, rm, ...

- dev1 :
  - créé un nouveau fichier (data/jour.txt), l'édite y ajoutant des jours de la semaine
  - créé un nouveau répertoire (doc) dans lequel il ajoute un nouveau fichier (doc.txt)
  - propage sa version
- dev2 :

- met à jour sa version

  A t-il bien récupéré les nouveaux fichiers et le nouveau répertoire ?
- modifie le nom de ce fichier (data/jour.txt en data/jours.txt)
- propage sa version

- dev1 :
  - met à jour sa version

Vérifier que le changement de nom a bien eu lieu sur le dépôt (et sur la copie de dev2).

5 Outils graphiques / Ligne de commande

Pour ceux qui ont utilisés le terminal linux, regarde comment on peut utiliser dolphin….et réciproquement.