

# Mise en place d'un serveur subversion + Utilisation de subversion

Document écrit, en  $\text{\LaTeX}$ , par : Frouin Jean-Michel  
Date: 2007-06-18 18:31:52 +0200 (lun, 18 jun 2007)

Rev: 107

Ce document est sous licence :



8 août 2007

## Table des matières

<b>1</b>	<b>Historique</b>	<b>3</b>
<b>2</b>	<b>Introduction</b>	<b>3</b>
<b>3</b>	<b>Installation d'un serveur subversion</b>	<b>3</b>
3.1	Création des dépôts . . . . .	3
3.2	Configuration d'un dépôt . . . . .	3
3.2.1	/usr/svn/nom_du_module/conf/passwd . . . . .	3
3.2.2	/usr/svn/nom_du_module/conf/svnserve.conf . . . . .	4
3.2.3	/usr/svn/nom_du_module/conf/authz . . . . .	4
3.3	Configuration avec apache2 . . . . .	4
3.3.1	Configurer le module subversion pour apache2 . . . . .	4
3.3.2	Droits du dépôt . . . . .	5
3.3.3	Relancer le serveur apache2 . . . . .	5
3.3.4	Test final . . . . .	5
<b>4</b>	<b>Manipulations de dépôt</b>	<b>5</b>
4.1	Importer un répertoire dans un dépôt . . . . .	5
4.2	Obtenir une copie locale d'un projet . . . . .	6
4.3	Manipulation des fichiers . . . . .	6
4.4	Remonter les modifications sur le serveur . . . . .	6
4.5	Mettre à jour sa copie locale d'un projet versionné . . . . .	6
4.6	Récupérer une ancienne version . . . . .	7
4.7	Création d'un tag . . . . .	7
4.8	Création d'une branche . . . . .	7
4.9	Effectuer un merge depuis une branche . . . . .	7
4.10	Informations sur la copie locale . . . . .	7
4.11	Obtenir la liste des fichiers d'un dépôt . . . . .	8
4.12	Afficher le contenu d'un fichier sur le dépôt . . . . .	8
4.13	Nettoyer . . . . .	8
4.14	Editeur de message svn . . . . .	8
<b>5</b>	<b>Propriétés</b>	<b>8</b>
<b>6</b>	<b>Substitution de mot clé</b>	<b>9</b>
<b>7</b>	<b>Références</b>	<b>9</b>
7.1	En ligne . . . . .	9
7.2	Livres . . . . .	9

# 1 Historique

Version 1.00, Juin 2007 : Création du document.

## 2 Introduction

subversion est un logiciel permettant de faire de la gestion de version. A l'instar d'autres outils qui verrouille un fichier lorsque quelqu'un travaille dessus, subversion permet à plusieurs personnes de travailler sur le même fichier. En cas de conflit de version, lors d'une remontée des modifications, subversion vous laisse le soin de résoudre le conflit.

## 3 Installation d'un serveur subversion

Pour installer subversion il suffit (sous GNU/Debian, Ubuntu) de faire : `[sudo] apt-get install subversion`.

Si on prévoit de relier le serveur subversion à un serveur apache, il faut installer aussi le paquet **libapache2-svn**.

Il peut être intéressant d'installer **subversion-tools**.

### 3.1 Création des dépôts

Avant de commencer à configurer le serveur, on va créer les dépôts. Un dépôt représente l'endroit où sont stockés les fichiers, les historiques de version, les branches, les tags et les fichiers de configurations pour ce dépôt. Directement sur le serveur subversion, on crée un répertoire qui contiendra tous les dépôts `mkdir -p /usr/svn`. On se place dans ce dossier `cd /usr/svn`. Enfin on crée autant de dépôts que de modules à gérer sous subversion : `svnadmin create -fs-type fsfs nom_du_module`.

### 3.2 Configuration d'un dépôt

**Si vous voulez configurer subversion avec apache2, la configuration des fichiers passwd et svnserve.conf n'est pas nécessaire, mais le fichier authz devra être renseigné. Dans ce cas, allez directement 3.2.3, page 4.**

La première chose à faire est d'aller dans le répertoire : `/usr/svn/nom_du_module/conf`.

Là il faut éditer 2 fichiers :

Le fichier **passwd** qui contient les noms des utilisateurs, qui utiliseront svn, et leur mot de passe.

Le fichier **svnserve.conf** qui contient la configuration propre du serveur svn.

#### 3.2.1 /usr/svn/nom\_du\_module/conf/passwd

Dans ce fichier rien d'exceptionnel, la syntaxe est la suivante :

```
[users]
nom = mot_de_passe
```

### 3.2.2 /usr/svn/nom\_du\_module/conf/svnserve.conf

Ici un peu plus de travail que dans passwd :

```
1 [general]
2 anon-access = read
3 auth-access = write
4
5 password-db = passwd
6
7 authz-db = authz
8
9 realm = Nom du depot
```

### 3.2.3 /usr/svn/nom\_du\_module/conf/authz

Dans ce fichier, la section *[groups]* permet de définir des groupes, qui seront utilisés juste après pour définir les droits d'accès au différents répertoires du dépôt.

```
1 [groups]
2 dev = snoogie, pierre
3
4 [/]
5 anonymous = r
6
7 [/trunk]
8 @dev = rw
9
10 [/branches]
11 @dev = rw
12
13 [/tags]
14 @dev = rw
```

Si le groupe dev n'existe pas, le créer avec *addgroup dev*. Ensuite ajouter l'utilisateur à dev avec *addgroup utilisateur dev*.

## 3.3 Configuration avec apache2

### 3.3.1 Configurer le module subversion pour apache2

Il peut être utile de faire un *a2enmod dav\_svn*, pour activer le module subversion pour apache2. Moi je n'en ai pas eu besoin.

Il faut, ensuite, ajouter pour chaque module du dépôt dans le fichier */etc/apache2/sites-enabled/000-default* :

```
1 <location /svn/nom_du_module>
2     DAV svn
3     SVNPath /usr/svn/nom_du_module
4
5     AuthType Basic
6     AuthName "Depot SVN pour le module nom"
7     AuthUserFile /usr/svn/svn.htpasswd
8     Require valid-user
9     AuthzSVNAccessFile /usr/svn/nom_du_module/conf/authz
10 </location>
```

Maintenant il faut construire le fichier *svn.htpasswd* :

Par exemple pour ajouter l'utilisateur snoogie : `htpasswd -cm /usr/svn/svn.htpasswd snoogie`.

Pour un utilisateur anonyme : `htpasswd -m /usr/svn/svn.htpasswd anonymous`.

### 3.3.2 Droits du dépôt

Il faut penser à donner les droits d'accès, du dépôt subversion, au serveur apache. On ajoute le dépôt au groupe du serveur apache : `chown -R frouin :www-data /usr/svn/nom_du_module`, puis pour permettre aux utilisateurs de pouvoir ajouter des éléments au dépôt : `chmod -R 775 /usr/svn/nom_du_module`.

### 3.3.3 Relancer le serveur apache2

Enfin on relance le serveur apache2 pour qu'il ai conscience des modifications : `/etc/init.d/apache2 restart`.

### 3.3.4 Test final

Il ne reste plus qu'à tester que tout fonctionne :

(J'ai ajouté serveursvn à la liste des hosts pour ne pas avoir à utiliser l'ip à chaque fois)

```
svn mkdir http://serveursvn/svn/nom_du_module/trunk
svn mkdir http://serveursvn/svn/nom_du_module/branches
svn mkdir http://serveursvn/svn/nom_du_module/tags
```

## 4 Manipulations de dépôt

Il est possible d'ajouter, à presque toutes les commandes qui suivent, l'argument `-message "Message"`, ou sont équivalent `-m "Message"`. Cette option permet de renseigner, directement depuis la ligne de commande, le message associé à l'opération. Une commande très utile pour obtenir plus d'aide : `svn help commande`.

Une autre option très utile est l'option `-r` qui permet de spécifier une révision à la commande. Cette option peut aussi utiliser certains mots clés comme *HEAD* (La dernière révision), *BASE*, *COMMITTED* ou *PREV*. Cette option peut aussi prendre une date : 2007-06-13, une heure 16:15 ou les deux "2007-06-13 16:15".

*Dans les commandes qui suivent, les parenthèses proposent une alternative.*

### 4.1 Importer un répertoire dans un dépôt

Une fois qu'un dépôt a été créé pour héberger un projet, il faut importer les fichiers, du projet, dans le dépôt. En supposant que tous les fichiers du projet se trouvent dans un répertoire nommé *projet*<sup>1</sup>, l'importation se fait en utilisant la commande suivante : `svn import projet http://svnserve/svn/nom_du_module`.

---

<sup>1</sup> En supposant aussi que l'on est dans le répertoire contenant le répertoire *projet*.

## 4.2 Obtenir une copie locale d'un projet

Pour récupérer une copie d'un projet, il faut utiliser la commande `svn checkout (co) http://svnserve/svn/nom_du_projet/trunk/`. Pour récupérer la version 45 d'un projet, il faut utiliser l'option `-r` : `svn co -r 45 http://svnserve/svn/nom_du_projet/trunk/`.

## 4.3 Manipulation des fichiers

Pour pouvoir manipuler les fichiers, subversion fournit les commandes `svn [add — del (rm) — copy (cp) — move (mv)] fichier`, qui permettent d'ajouter, de supprimer, de copier ou de déplacer un fichier. Pour créer un répertoire la commande `svn mkdir repertoire` suffit. **Il est possible d'utiliser `rm`, `cp`, `mv` et `mkdir` directement sur le dépôt, en spécifiant son emplacement en argument de la commande, sans avoir de copie locale du projet.**

Il est possible d'annuler toutes les modifications faites, sur la copie locale, depuis la dernière synchronisation avec le serveur en utilisant `svn revert fichier`. Bien que rarement utilisé, une commande permet de verrouiller un fichier : `svn lock fichier`.

## 4.4 Remonter les modifications sur le serveur

La commande `svn status -u` permet de savoir si lors du commit, des conflits apparaîtront.

Il est possible de remonter les modifications sur le serveur subversion en utilisant la commande `svn commit (ci)`. Dans ce cas toutes les modifications de tous les fichiers (versionnés) seront remontés. En ajoutant le nom d'un fichier en argument de la commande `svn ci`, il est possible de ne remonter que les modifications du fichier.

Si toutefois un conflit apparaît, 3 fichiers seront créés, `fichier.mine`, `fichier.rVieux`, `fichier.rNouveau`. Une fois le conflit corrigé, il faudra le signaler à subversion (sans quoi il ne sera pas possible de remonter les modifications sur le fichier) grâce à `svn resolved fichier`.

## 4.5 Mettre à jour sa copie locale d'un projet versionné

Il se peut<sup>2</sup>, qu'à un moment donné la copie locale d'un projet sur laquelle on travaille soit périmée<sup>3</sup>. Pour mettre à jour la copie locale on utilise la commande `svn update (up)`.

```
snoogie@lenovo:~/Guide$ svn up
D   Article_Libretto.tex
D   Article_Libretto
D   Article_Libretto_us.tex
D   Article_Libretto.pdf
D   Article_Libretto_us.pdf
```

<sup>2</sup>C'est même fort probable.

<sup>3</sup>C'est à dire que la copie locale n'est pas à jour par rapport à la copie disponible sur le serveur.

A chapters/outils/subversion\_corps.tex  
U chapters/outils/subversion.tex  
A code/tag.sh  
A code/build\_a\_dist.sh  
A code/branch.sh  
A code/get\_source.sh  
U Makefile  
A Libretto.tex  
U Guide.pdf  
U notes.txt  
A Libretto\_us.tex  
U Guide.tex  
A Guide.128.pdf  
Actualisé à la révision 75.

## 4.6 Récupérer une ancienne version

Pour récupérer une ancienne version d'un fichier, par exemple la version 7 du fichier *fichier*, il faut passer par `svn update -r 7 fichier`. Pour un dépôt complet : `svn update -r 7`.

## 4.7 Création d'un tag

La création d'un tag se fait via `svn copy (cp) http://serveursvn/svn/nom_du_module/trunk http://serveursvn/svn/nom_du_module/tags/nom_du_tag`.

## 4.8 Création d'une branche

Pour une branche, c'est la même chose : `svn copy (cp) http://serveursvn/svn/nom_du_module/trunk http://serveursvn/svn/nom_du_module/branches/nom_de_la_branche`.

## 4.9 Effectuer un merge depuis une branche

**Cette opération est faite par le mainteneur du trunk.**

Il faut d'abord obtenir la dernière version du trunk : `svn co http://serveur/svn/nom_du_module/trunk`. Ensuite il faut merger les modifications de la branche avec le trunk : `svn merge -r 2 :4 http://serveursvn/svn/nom_du_module/branches/nom_de_la_branche`. Il est possible d'annuler un merge, par exemple : `svn merge -r 4 :2 http://serveursvn/svn/nom_du_module/branches/nom_de_la_branche`.

## 4.10 Informations sur la copie locale

Pour obtenir des informations sur la copie locale d'un projet versionné, il est possible d'utiliser la commande : `svn status (st)`. L'option `-verbose (-v)` rendra `svn st` beaucoup plus verbeux. Le résultat ressemble à cela :

D Article\_Libretto\_us.pdf  
A chapters/outils/subversion\_corps.tex  
M chapters/outils/subversion.tex

La lettre en tête de ligne indique l'opération qui sera effectuée lors d'un *svn ci*. Le tableau suivant indique leur signification :

A	Ajout du fichier.
C	Un conflit existe pour le fichier.
D	Suppression du fichier.
M	Modification du fichier.
U	Fichier mis à jour.

La commande *svn diff fichier* fournit des informations, en utilisant le format de diff unifié, sur les différences entre la version locale et la version du serveur. Il est possible d'utiliser *svn diff fichier* pour produire un patch : *svn diff fichier > fichier.patch*.

Une autre commande est utile, la commande *svn log fichier* qui fournit un affichage complet sur un fichier ou sur l'ensemble du projet (*svn log*).

#### 4.11 Obtenir la liste des fichiers d'un dépôt

Il est possible d'obtenir la liste complète des fichiers contenus dans un dépôt en utilisant la commande *svn list (ls) url\_du\_depot*.

#### 4.12 Afficher le contenu d'un fichier sur le dépôt

Avec la commande *svn cat url\_du\_depot/fichier*, il est possible d'afficher le contenu d'un fichier directement depuis le dépôt.

#### 4.13 Nettoyer

Il est possible d'utiliser *svn cleanup* pour nettoyer une copie locale. (Parfois c'est bien pratique quand plus rien ne va)

#### 4.14 Editeur de message svn

Lors d'un *svn ci*, subversion lance un éditeur vous permettant d'enregistrer un message pour garder une trace de la révision que l'on remonte. Par défaut l'éditeur est nano, si on veut vim il suffit de modifier la variable globale : EDITOR. Par exemple, *export EDITOR=vi*.

## 5 Propriétés

Chaque élément d'un dépôt possède des propriétés qu'il est possible de renseigner en utilisant *svn propset*, de lister *svn proplist*, de récupérer *svn propget* ou de supprimer *svn proptdel*. Il est ainsi possible de modifier le commentaire de log d'un fichier avec : *svn propset svn :log 'nouveau commentaire' -r45 -revprop*.

Quelques propriétés :

copyright	
licence	svn propset -F /usr/share/gnu.txt fichier
countline	
FIXME : svn :ignore	Très pratique pour les projets complexes.
svn :log	
svn :executable	
svn :mime-type	
svn :keywords	

## 6 Substitution de mot clé

subversion possède un mécanisme très pratique, permettant de remplacer dans un fichier une balise : \$Balise\$ par les informations qu'elle représente. Ainsi il est possible d'insérer dans un fichier C++, une page html, le numéro de révision svn du fichier, la date ou le nom de l'auteur de la dernière mise à jour.

Pour pouvoir fonctionner, il faut ajouter le nom du mot clé dans la propriété svn **svn :keywords** du fichier : *svn propedit svn :keywords fichier*. Il est possible de modifier ces propriétés directement *svn propset "Date Rev Id" svn :keywords fichier*.

Mot clé	Complété en :
\$Date : \$	Dernière modification du document.
\$Revision : \$ ou \$Rev : \$	Dernière révision du document.
\$Author : \$	Dernière personne ayant modifié le document.
\$HeadURL : \$	URL du document.
\$Id : \$	Combinaison de Date, Author et Revision.

## 7 Références

### 7.1 En ligne

Le SVN book : <http://svnbook.red-bean.com>

### 7.2 Livres

[1]

## Références

[1] William Nagel. *Subversion version control*. Bruce peren's open source series, 2005.