

Installation de Subversion sous Debian

par Gildas Cuisinier ([Hikage](#)) ([Blog](#))

Date de publication : 07/08/2007

Dernière mise à jour :

Cet article présente l'installation d'un serveur Subversion sur une distribution Linux Debian ainsi que les bases de gestion d'un projet sur ce même serveur. Il présentera ensuite une application Web, USVN, qui propose une interface simple de gestion des dépôt Subversion.

Qu'est-ce que subversion ?

Qu'est-ce qu'un VCS ?

Pourquoi Subversion ?

Atomicité

Gestion des déplacements / renommages

Gestion des répertoires

Révisions

Gestion des accès

II - Subversion sous debian

Installation

Gestion d'un dépôt

II-C - Utilisation du client

Check out

Ajout de fichiers/répertoires

Intégration des modifications sur le serveur

Intégration, le retour

Utilisation en réseau

SVNServe

SSH

Module apache

Installation

Configuration

Gestion des utilisateurs

Relancement du serveur

III - Interface web de gestions

Présentation d'USVN

Installation

Prérequis

Ecrans d'installations

Installation - Accueil

Installation - Choix de la langue

Installation - Licence

Installation - Configuration des répertoires

Installation - Configuration de la base de données

Installation - Création de l'administrateur

Installation - Fin

Utilisation

IV - Conclusion

Remerciements

Ressources

Qu'est-ce que subversion ?

Qu'est-ce qu'un VCS ?

Subversion est un VCS, Version Control System, ou logiciel de gestion de version. C'est un logiciel qui permet à plusieurs développeurs de travailler sur un même projet, voir sur un même fichier simultanément.

Il est responsable de la gestion des accès concurrents, ou plus exactement des modifications concurrentes sur un même fichier. De plus si les modifications ne sont pas conflictuelles, Subversion assure la fusion de celles-ci afin d'avoir au final une version du projet qui comprend toutes les modifications.

Dans les cas où les modifications portent sur les même lignes, Subversion aura besoin d'une intervention d'un utilisateur afin de résoudre manuellement ces conflits.

Pourquoi Subversion ?

Subversion a été développé dans le but de remplacer à terme CVS en comblant quelques lacunes inhérentes à ce dernier :

- Atomicité
- Gestion du renommage / déplacement
- Gestion des révisions
- Gestion des accès

Atomicité

Dans CVS les modifications envoyées par un développeur étaient intégrées dans le dépôt une par une dès leur réception. Si pour une raison ou l'autre, l'envoi de ces modifications était interrompu (problème réseau, plantage du système d'exploitation du coté client, ...), les modifications déjà reçues étaient intégrées.

Dans ce cas précis, le dépôt est dans un état incohérent, et le projet n'est plus forcément compilable.

Afin d'éviter cela, Subversion intègre les modifications uniquement lorsque toutes celles-ci ont été reçues intégralement et qu'aucun problème n'est survenu.

Le cas échéant, aucune modification n'aura été intégrée, et le développeur devra faire une autre tentative, après résolution des problèmes éventuels.

Ce système d'intégration est comparable au système de transactions des bases de données [commit/rollback](#).

Gestion des déplacements / renommages

Une autre faiblesse de CVS est la gestion, ou plus exactement l'absence de gestion, du renommage/déplacement des fichiers. En effet, CVS perçoit ces actions comme la suppression du fichier *source*, et la création du fichier *destination*. Ce qui a pour conséquence que l'historique du fichier original n'est pas lié au nouveau fichier.

Dans Subversion, si un fichier est déplacé ou renommé à l'aide de la commande **svn move**, l'historique reste associé au fichier destination, ce qui permet de voir toutes ses modifications depuis le début.



Attention, si le déplacement n'est pas fait avec la commande svn, l'historique sera perdu de la même manière que dans CVS.

Gestion des répertoires

Un autre avantage de Subversion est sa gestion des répertoires. Pour chaque révision, une liste des fichiers contenus dans un répertoire est gardé.

Cela a pour avantage de pouvoir comparer le contenu d'un répertoire par rapport à une ancienne version rapidement.

Révisions

La notion des révisions entre Subversion et CVS est différente. Dans ce dernier une révision était associée à un fichier en particulier.

Cela peut paraître logique, mais cela pose un problème : Comment récupérer une copie du dépôt à un moment donné ? Un fichier pouvant être à la révision 3, tandis qu'un autre à la révision 10, et un plus récent à la révision 1.

Dans Subversion, la notion de version est appliquée sur l'ensemble du dépôt. Dès qu'une modification est apportée à un fichier dans le dépôt, c'est l'ensemble des données qui passe au numéro de révision suivant.

Il est dès lors plus simple de retrouver l'état d'un dépôt à un moment donnée, en gardant une trace de la révision à ce moment là.

Gestion des accès

Comme il sera démontré plus loin dans cet article, la gestion des accès peut être très fine, spécialement lorsque Subversion est lié à Apache.

Il est dès lors possible de spécifier différents accès par utilisateur, par groupe, et ce, pour chaque fichier/répertoire dans le dépôt, et non pas un accès global à celui-ci.

II - Subversion sous debian

Installation

L'installation de Subversion se fait simplement par la commande :

```
aptitude install subversion
```

Cette commande installe le client Subversion, différentes commandes de gestion de dépôt ainsi qu'un exécutable serveur.

Sur une distribution Debian Etch, la version est 1.4.2.

Gestion d'un dépôt

Cela fait, il est nécessaire de créer un premier dépôt.

Dans le cadre de cet article, le répertoire de base des dépôts sera /var/subversion/depot/

La création d'un dépôt pour un projet nomdemonprojet se fait via la commande :

```
svnadmin create /var/subversion/depot/nomdemonprojet
```

Le résultat devrait être un répertoire nomdemonprojet crée dans /var/subversion/depot/

II-C - Utilisation du client

Dès lors, il est déjà possible de travailler sur celui-ci localement (sur la même machine que le dépôt lui même). L'accès à distance nécessitant un peu de configuration, cela sera expliqué par après.

En attendant, voici un petit descriptif des commandes de bases du client Subversion.

Check out

La première étape dans l'utilisation de Subversion est de récupérer une copie locale du dépôt :

```
svn co file:///var/subversion/depot/nomdemonprojet
```

Si tout c'est bien déroulé, un répertoire nomdemonprojet a du être créé dans le répertoire courant.

Ce répertoire doit contenir les fichiers de la dernière révision disponible sur le dépôt.

Dans ce cas-ci, le projet venant d'être créé, ce répertoire est vide.

Ajout de fichiers/répertoires

Dans ce répertoire, vous êtes libre de travailler normalement : création de répertoire, ajout de fichier.

Dans le cadre de cet article, la hiérarchie de fichiers/répertoires suivante a été créée :

```
nomdemonprojet
+ src
+ java
+ com
+ developpez
+ hikage
+ TestMain.java
+ Application.java
+ webapp
+ WEB-INF
+ web.xml
```

Il est ensuite nécessaire d'informer Subversion que ces fichiers doivent être pris en compte lors de la prochaine publication des modifications sur le dépôt :

```
svn add src
```

Le résultat de cette commande est :

```
debian:~/nomdemonprojet# svn add src
A      src
A      src/webapp
A      src/webapp/WEB-INF
A      src/webapp/WEB-INF/web.xml
A      src/java
A      src/java/com
A      src/java/com/developpez
A      src/java/com/developpez/hikage
A      src/java/com/developpez/hikage/TestMain.java
A      src/java/com/developpez/hikage/Application.java
```

Intégration des modifications sur le serveur

La dernière commande n'a fait que mettre une "étiquette" sur des fichiers, mais ceux-ci n'ont pas encore été envoyés vers le dépôt.

Cette action est réalisée via la commande :

```
svn commit src -m "Message expliquant le modification effectuées"
Ajout      src
Ajout      src/java
Ajout      src/java/com
Ajout      src/java/com/developpez
Ajout      src/java/com/developpez/hikage
Ajout      src/java/com/developpez/hikage/Application.java
Ajout      src/java/com/developpez/hikage/TestMain.java
Ajout      src/webapp
Ajout      src/webapp/WEB-INF
Ajout      src/webapp/WEB-INF/web.xml
Transmission des données ...
Révision 1 propagée.
```

Le paramètre **-m "Mon Message"** spécifie un message qui sera associé à la publication (et donc à la nouvelle révision).

Il est courant d'expliquer comme message le but des modifications réalisées : ajout d'une fonctionnalité, correction d'un bug, ...

Intégration, le retour

Imaginons maintenant que du code a été ajouté dans le fichier TestMain.java (qui était jusqu'ici un simple fichier vide), et que lors de la publication des modifications, le résultat affiche :

```
debian:~/nomdemonprojet/src/java/com/developpez/hikage# svn commit TestMain.java -m "Ajout du code
par developpeur 1"
Envoi      TestMain.java
svn: Échec de la propagation (commit), détails :
svn: Le chemin '/src/java/com/developpez/hikage/TestMain.java' est obsolète dans la transaction
'3-1'
```

Le message informe que le fichier du dépôt est plus récent (une révision supérieure) que celle disponible en locale.

Il est donc nécessaire de mettre à jour cette dernière grâce à la commande :


```
svn update TestMain.java
C      TestMain.java
Actualisé à la révision 3.
```

La mise à jour a été effectuée mais un conflit (le **C** à coté du nom du fichier) est apparu.

Lors d'un conflit, Subversion crée trois nouveaux fichiers dans le répertoire, ici TestMain.java.mine (qui est une copie de la version modifiée localement), TestMain.java.r1 (qui est la version locale avant modification, c'est à dire celle récupérée lors du check out ou de la dernière mise à jour), et TestMain.java.r3 qui est la dernière version sur le serveur.

Le fichier TestMain.java quand à lui possède des informations sur les différences entre chaque fichiers.

Il est donc nécessaire de réaliser la fusion manuellement, en gardant les bonnes lignes.

 *Dans le cas de la ligne de commande, la fusion doit être réellement faite à la main.*

Lors de l'utilisation de client graphique, il est fréquent d'avoir une fenêtre qui affiche les différences entre les fichiers et propose une interface simple pour réaliser la fusion.

Une fois la fusion effectuée, il faut dire à Subversion que le problème a été résolu :

```
svn resolved TestMain.java
Conflit sur 'TestMain.java' résolu
```

En plus de supprimer l'étiquette 'Conflit' (qui empêche le fichier d'être envoyé sur le dépôt), cette commande supprime les trois fichiers temporaires cités plus haut.

Le cas du "conflit" est le seul qui nécessite l'intervention manuelle. Lors d'une mise à jour de la copie locale (svn update), les informations possibles associées au fichier peuvent être :

C	Conflit, c'est le cas présenté dans cet article
A	Ajout, un nouveau fichier a été ajouté sur le dépôt et est donc récupéré dans la copie locale
D	Suppression, un fichier a été marqué comme supprimé, et donc la copie locale est supprimée aussi
G	Fusion, des modifications ont été réalisées sur un fichier qui a été modifié dans la copie locale, mais les modifications ne sont pas conflictuelle. Subversion intègre donc ces modifications dans la copie locale
M	Modification, des modifications ont été réalisées sur un fichier qui n'a pas été modifié localement, les modifications sont donc intégrées dans la version locale

Voilà qui finit la présentation des commandes de base de Subversion. Il en existe beaucoup d'autre afin de créer des patches entre version (svn diff), annuler des modifications locales (svn revert), et bien d'autres encore qui nécessiteraient un article à part afin d'être présenté correctement.

Utilisation en réseau

Jusqu'ici le serveur était accessible uniquement en local et son utilité est donc restreinte.

Pour mettre à disposition les dépôts à des clients distants, il existe différents moyens possédant chacun leurs avantages et leurs inconvénients.

SVNServe



La première méthode est d'utiliser SVNServe, un démon fourni dans le package Subversion. Celui-ci peut être configuré en serveur Standalone ou via InetD.

L'avantage de cette méthode est qu'il ne nécessite pas d'autre dépendance.

L'inconvénient est qu'il écoute sur un port propre (qui est bien sur configurable), et qu'il risque donc d'être bloqué par des pare-feux.

SSH

Il est aussi possible d'accéder au dépôt via un tunnel SSH. Cela nécessite bien sûr un serveur SSH installé sur la machine qui héberge le dépôt, mais c'est souvent le cas sur des machines Linux.

Du côté du client, il est nécessaire que celui-ci gère ce type d'accès. Parmi ceux-ci, citons  **Tortoise SVN**,  **SmartSVN** ou encore **Subclipse**

Module apache

La dernière méthode, et la plus intéressante, est d'utiliser un module Apache pour Subversion.

Le principal intérêt par rapport à l'accès via SvnServe ou SSH est que le port HTTP est très rarement filtré par les pare-feux, et donc est accessible de partout.

De plus l'authentification étant effectuée via les modules d'Apache, les moyens d'authentification de celui-ci (fichier d'utilisateur, annuaire LDAP, base de données) sont disponibles.

Comme il a été dit au début de cet article, l'intégration Apache / Subversion fournit un dispositif de gestion des droits d'accès plus fine.

Dans le cadre de ce tutoriel, c'est cette méthode qui sera expliquée.

Installation

La première chose à faire est d'installer Apache si ce n'est pas encore le cas :

```
aptitude install apache2
```

Ensuite, il faut installer le module svn en lui même.

Sous debian Sarge :

```
aptitude install libapache2-mod-dav libapache2-svn
```

Tandis que sous debian Etch, cela se fait via :

```
aptitude install libapache2-mod-svn
```

Configuration

Une fois le module installé, il faut configurer Apache afin d'intégrer le dépôt.

Pour ce faire, nous allons configurer Apache afin que l'URL de type `http://monserveur/svn` fournisse un accès au dépôt.

Tout d'abord, il faut informer Apache qu'il doit prendre en compte le module SVN :

```
a2enmod dav_svn
```

Ensuite, le fichier de configuration `/etc/apache2/mod-available/dav_svn.conf` doit être édité :

```
<Location /svn>
  DAV svn
  Require valid-user
  SVNParentPath /var/subversion/depot/
  AuthType Basic
  AuthName "Mon dépôt"
  AuthUserFile /var/subversion/conf/htpasswd
  AuthzSVNAccessFile /var/subversion/conf/access
</Location>
```

Ligne	Description
Location /svn	Spécifie que le dépôt sera accessible via <code>http://ip/svn</code> Ou ip représente l'adresse ip du serveur
DAV svn	Active la gestion SVN sur ce répertoire
Require valid-user	Nécessite un utilisateur valide afin d'accéder à ce répertoire
SVNParentPath	Configure le chemin vers le répertoire parent des dépôts
AuthType	Spécifie le type d'authentification
AuthName	Spécifie le nom qui sera affiché dans la boîte de demande de mot de passe
AuthUserFile	Spécifie le fichier des utilisateurs
AuthzSVNAccessFile	Spécifie le fichier de gestion des accès

Gestion des utilisateurs

Cela fait, l'étape suivante est de créer le fichier `/var/subversion/conf/htpasswd` qui contiendra les utilisateurs et leurs mots de passe.

La création du premier utilisateur est réalisée via la commande :

```
htpasswd -c /var/subversion/conf/htpasswd hikage
```

hikage représente le nom de l'utilisateur, et son mot de passe sera demandé par la commande.

Le paramètre **-c** spécifie qu'il est nécessaire de créer le fichier. L'ajout des prochains utilisateurs se fera donc via la commande :

```
htpasswd /var/subversion/conf/htpasswd utilisateur2
```

Une fois les utilisateurs créés, il reste encore à configurer leurs accès. Cela est réalisée dans le fichier `/var/subversion/conf/access`.

Le contenu de ce fichier dans le cadre de cet article pourrait être :


```
[groups]
developpez = hikage, utilisateur2
```

```
[nomdemonprojet:/]  
@developpez = rw  
* = r  
  
[projetprivehikage:/]  
hikage = rw  
* = r  
  
[projetprivehikage:/documentation/utilisateur]  
auteurdoc = rw
```

La section [groups], comme son nom le laisse sous entendre, permet de définir des groupes. Ici un seul groupe (**developpez**), dont les membres sont *hikage* et *utilisateur2*, est créé

La deuxième section ([nomdemonprojet:/]) définit les accès globaux au projet **nomdeprojet**. Ici le groupe developpezcom (@developpezcom) possède un droit en lecture/écriture, tandis que tout les autres (représenté par *) ont un accès en lecture seule.

La troisième section définit les accès globaux à un deuxième projet (projetprivehikage), accessible en écriture uniquement par hikage mais lisible par tous.

 *Remarquez que le fichier de configuration des droits d'accès peut être commun à plusieurs projets.*

La dernière section redéfinit les droits d'accès au répertoire documentation/utilisateur du projet projetprivehikage. Celui-ci sera accessible en écriture par auteurdoc.

Relancement du serveur

Ces étapes réalisées, il est nécessaire de relancer Apache afin qu'il prenne ces changements en compte :

```
/etc/init.d/apache2 restart
```

III - Interface web de gestions

Présentation d'USVN

Jusqu'ici, la gestion des dépôts était réalisée en console.

Si cela reste très utile, la console n'est pas forcément appréciée par tout le monde. De plus, afin de gérer les dépôts il est nécessaire d'avoir un compte administrateur, ou équivalent.

Pour résoudre cela, un projet de fin d'études de l'Epitech viens nous aider : USVN.

Celui-ci fournit une interface Web en PHP afin de prendre en charge la gestion de plusieurs dépôts.

USVN est fourni sous licence **CeCILL**.

Installation

Prérequis


Afin d'utiliser USVN, il faut s'assurer d'avoir toutes les dépendances requises :

- Apache 2
- PHP5
- SQLite
- mod_dav_svn
- mod_rewrite
- subversion
- mod_authz_svn

```
aptitude install libapache2-mod-php5 php5-sqlite
a2enmod rewrite
/etc/init.d/apache2 reload
```

Afin de poursuivre l'installation, l'archive de USVN doit être récupérée sur le site, et décompressée dans un répertoire accessible à Apache :

```
wget --no-check-certificate http://www.usvn.info/download/get/?file=0.6.3%20Mike.tgz
tar -zxvf usvn-0.6.3.tgz
mv usvn /var/www
chown -R www-data: /var/www/usvn
```

 **wget** est un outil en ligne de commande permettant de récupérer des fichiers distants. S'il n'est pas encore installé : **aptitude install wget**.

Cela fait, il est nécessaire de s'assurer que la configuration pour ce site est correcte. Pour ce faire, ajoutez les lignes suivantes dans le fichier `/etc/apache2/sites-available/default` (ou dans le fichier de votre nom de domaine) :

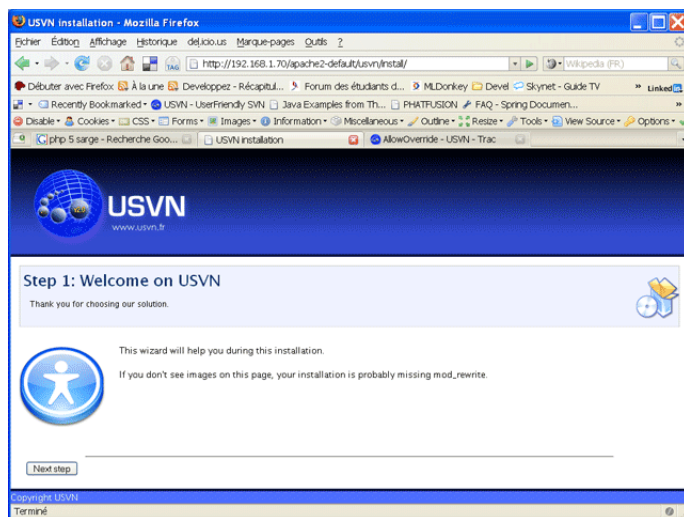
```
<Directory /var/www/usvn>
AllowOverride all all
</Directory>
```

Cela permet au site de définir des redirections via le fichier `.htaccess`.

Ecrans d'installations

La suite de l'installation est réalisée directement par l'interface fournie par USVN :

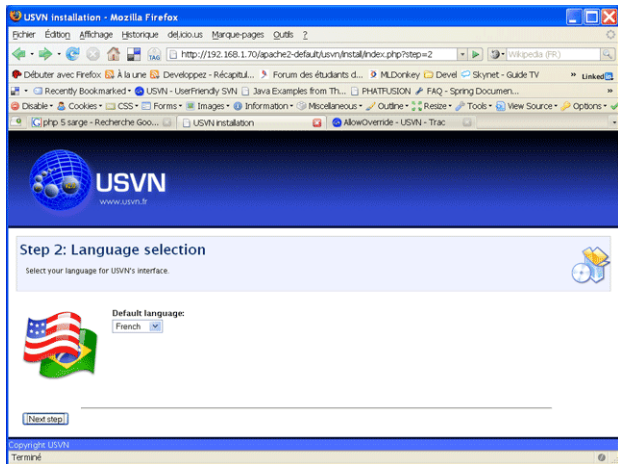
Installation - Accueil



Installation - Choix de la langue

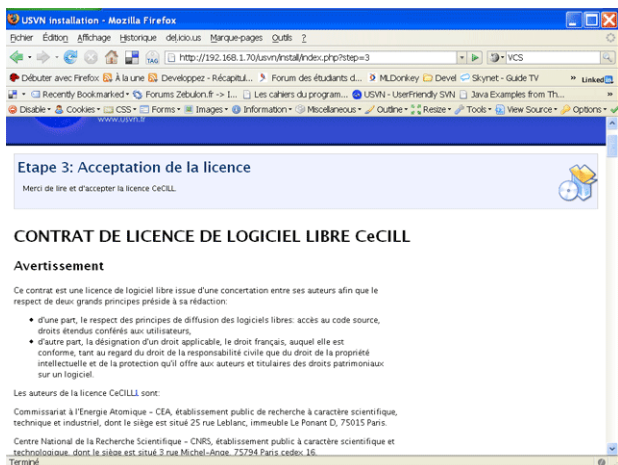
Le deuxième écran permet le choix de la langue, et à l'heure où j'écris cet article les langues suivantes sont gérées :

- Français
- Anglais
- Espagnol
- Chinois



Installation - Licence

Ensuite, la licence CeCILL est affichée.

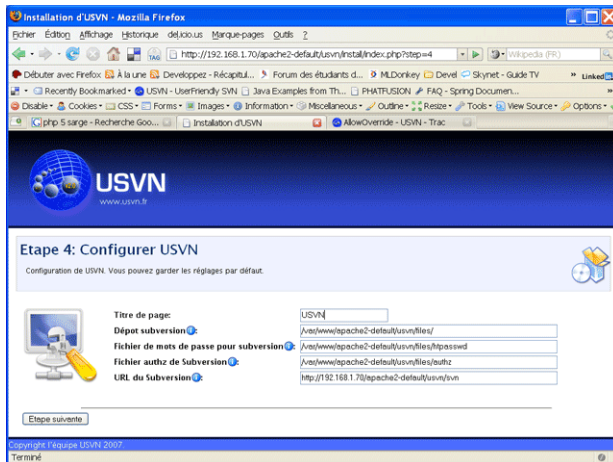


Installation - Configuration des répertoires

L'écran suivant est la configuration du nom du site, du chemin où seront stockés les dépôts ainsi que les fichiers de configuration, et l'URL d'accès à SVN.

La seule contrainte dans les chemins est qu'ils soient accessibles en lecture/écriture par Apache (utilisateur www-data sous Debian Etch).


Dans le cadre de cet article, les options par défaut sont utilisées.



Installation - Configuration de la base de données

L'écran suivant est celui du choix de la base de donnée : Mysql ou SQLite.

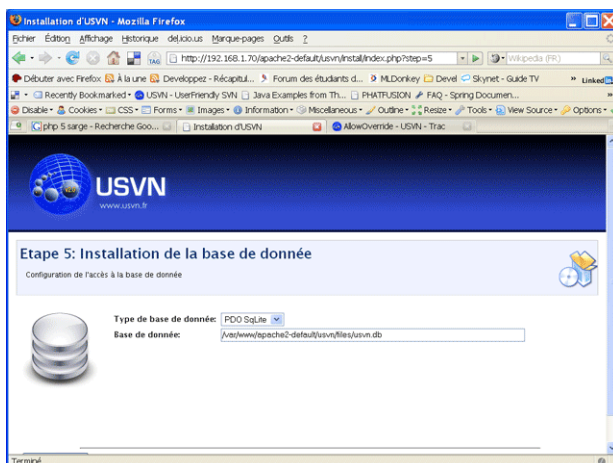
Dans cet article, partant du fait que MySQL n'est pas forcément installée, c'est SQLite qui est choisi.

 **SQLite est un moteur SQL particulier. Contrairement à des SGBD comme Mysql ou Postgres, SQLite ne fonctionne pas sur un mode client/serveur.**

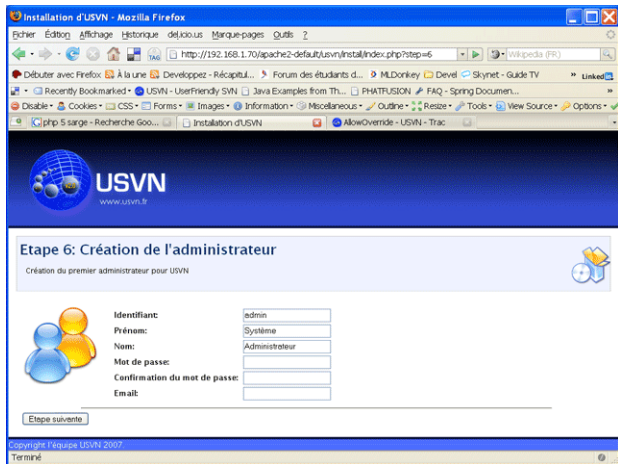
Le moteur est directement intégré dans le programme qui l'utilise, et les données sont stockées dans un fichier unique.

Il reste cependant très performant mais n'est pas prévu pour permettre de nombreux accès concurrents.

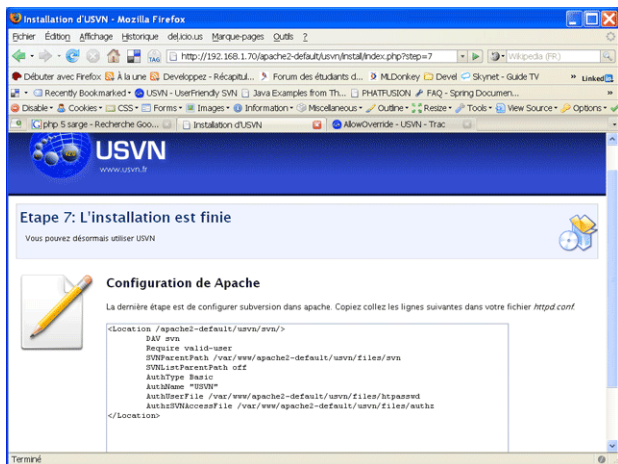
En contre partie, il ne nécessite aucune installation ni gestion des comptes utilisateurs.



Installation - Création de l'administrateur



Installation - Fin



L'installation est presque finie, il reste à mettre le texte de configuration fourni dans le fichier de configuration de Apache (/etc/apache2/sites-available/default par exemple).

La configuration d'Apache ayant été changée, il est nécessaire de recharger celle-ci :

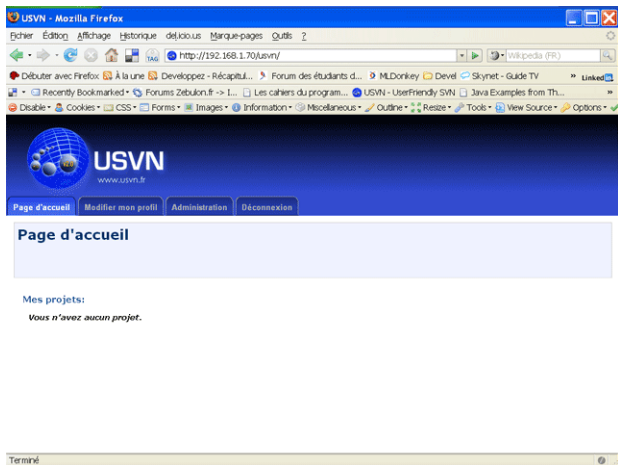
```
/etc/init.d/apache2 reload
```

Utilisation

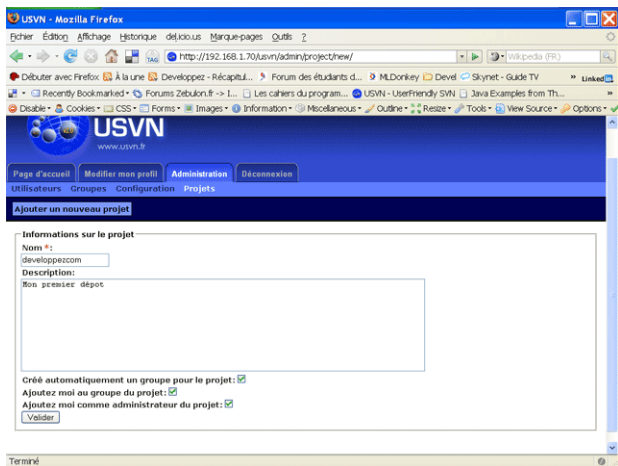
Si tout va bien, lors de l'accès au site, l'écran suivant devrait apparaître :



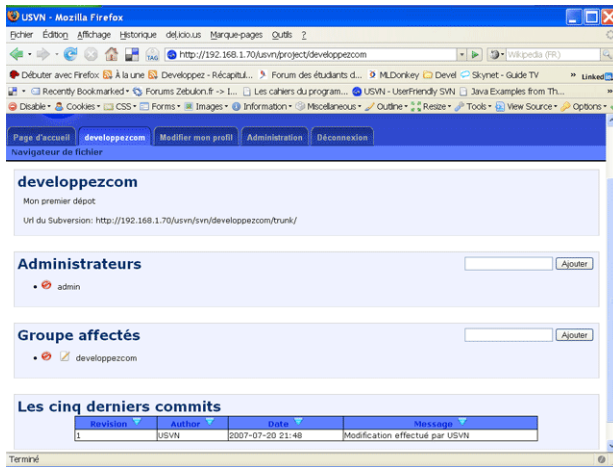
Une fois authentifié, les projets sont affichés



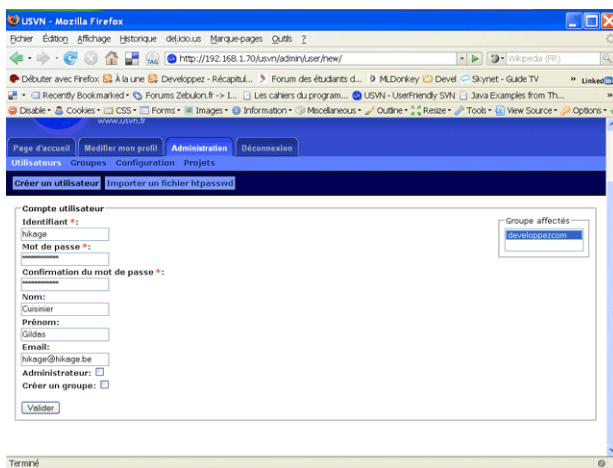
L'ajout d'un projet se fait via l'onglet **Administration** :



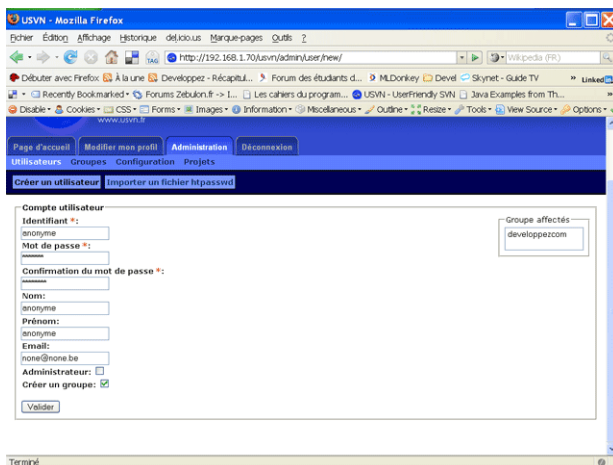
Une fois l'ajout effectué, il est possible de voir un résumé des informations le concernant.




Il est possible d'ajouter des utilisateurs via l'onglet Administration. Dans cet article deux utilisateurs sont ajoutés : hikage et anonyme.

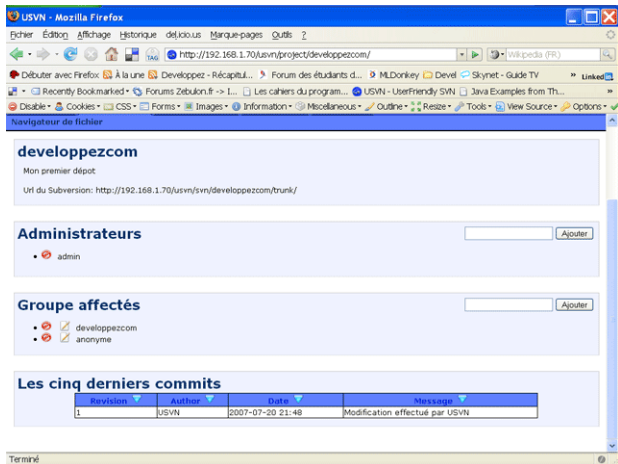


 Dans le cas de l'utilisateur hikage, celui-ci est ajouté au groupe developpezcom.



 Remarquons que durant la création de l'utilisateur anonyme, il est demandé de créer un groupe du même nom que celui-ci.

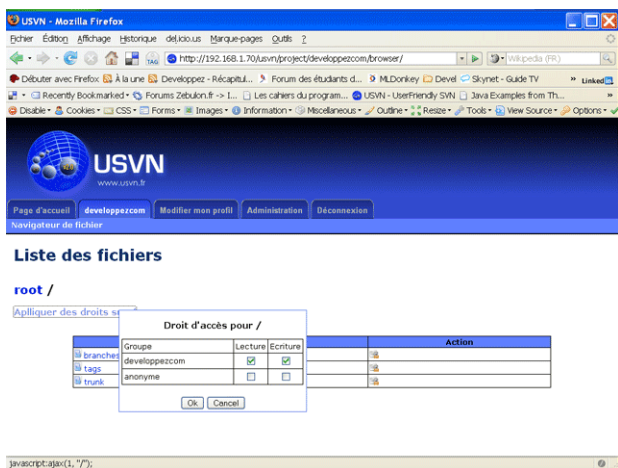
Une fois les utilisateurs ajoutés, il est possible de donner un accès au groupe anonyme pour le projet developpezcom :




Via l'écran de résumé d'un projet, il est possible de gérer les droits d'accès à celui-ci.

Pour cela, il faut entrer dans le navigateur de fichier.

Dans cet écran, le fait de cliquer sur *Appliquer des droits sur /* fait apparaître un popup permettant de spécifier les accès par groupe.



De plus cet écran permet de naviguer dans les répertoires, et de spécifier des droits individuels sur chaque fichiers/répertoires.

 Une démonstration Flash plus complète de la gestion d'un projet avec USVN est disponible [ici](#)

Il est intéressant de savoir qu'il est possible de déléguer une partie de la gestion d'un projet à un utilisateur.

Pour cela, dans l'écran de résumé du projet, il est nécessaire d'ajouter un utilisateur comme Administrateur.

Une fois qu'un utilisateur est ajouté comme administrateur il peut :

- Gérer les administrateurs d'un projet
- Gérer les groupes d'un projet
- Modifier les droits d'accès au projet

Cependant, il n'a pas le droit de créer/supprimer des utilisateurs, de créer/supprimer des groupes ou de créer/supprimer des projets, cette responsabilité reste à l'administrateur général.

IV - Conclusion

Remerciements

Je tiens à remercier Etanne d'avoir pris le temps de corriger cet article, ainsi que gege2061 et l'équipe de USVN pour l'aide précieuse qu'ils ont apportés.

Ressources

Voici quelques ressources pour Subversion :



Le site officiel de Subversion



Installation de Subversion sous Windows



F.A.Q. SCM



TortoiseSvn, un client Svn intégré à Windows



SmartSVN, un client Subversion en Java



Le site officiel de USVN

