TortoiseSVN

Un client Subversion pour Windows

Version 1.6.7

Stefan Küng
Lübbe Onken
Simon Large

TortoiseSVN: Un client Subversion pour Windows: Version 1.6.7

par Stefan Küng, Lübbe Onken, et Simon Large
traduction: Jérémy Badier (jeremy.badier@gmail.com)

Publié le 2009/12/09 10:28:44 (r17895)

Table des matières

Liste des illustrations

Liste des tableaux

Préface

- Travaillez-vous dans une équipe ?
- Vous est-il jamais arrivé de travailler sur un fichier, et quelqu'un d'autre travaillait sur le même fichier au même moment ? Avez-vous perdu vos changements de ce fichier à cause de cela ?
- Avez-vous jamais sauvegardé un fichier, et ensuite voulu annuler les changements effectués ? Avez-vous jamais souhaité voir à quoi ressemblait un fichier il y a quelques temps ?
- Avez-vous jamais trouvé un bug dans votre projet et voulu savoir quand ce bug a été introduit dans vos fichiers ?

Si vous avez répondu « oui » à l'une de ces questions, alors TortoiseSVN est fait pour vous ! Continuez seulement cette lecture pour découvrir comment TortoiseSVN peut vous aider dans votre travail. Ce n'est pas si difficile.

1. Public

Ce manuel est écrit pour les gens initiés à l'informatique qui veulent utiliser Subversion pour gérer leurs données, mais qui ne sont pas à l'aise pour utiliser le client en ligne de commande pour ce faire. Puisque TortoiseSVN est une extension du shell Windows, il est sous-entendu que l'utilisateur est familiarisé avec l'explorateur Windows et sait comment l'utiliser.

2. Guide de lecture

Cette Préface donne quelques explications au sujet du projet TortoiseSVN, de la communauté de personnes qui travaille dessus et des conditions de licence pour l'utiliser et le distribuer.

Cette Chapitre 1, *Introduction* explique ce qu'est TortoiseSVN, ce qu'il fait, ses origines et les bases pour l'installer sur votre PC.

Dans Chapitre 2, *Concepts de base du contrôle de version* nous donnons une courte introduction au système de contrôle de révision *Subversion* qui est à la base de TortoiseSVN. C'est emprunté à la documentation du projet Subversion et cela explique les différentes approches au contrôle de version et comment Subversion fonctionne.

Le chapitre Chapitre 3, *Le référentiel* explique comment mettre en place un référentiel local, ce qui est utile pour tester Subversion et TortoiseSVN en utilisant un seul ordinateur. Il explique aussi un peu l'administration d'un référentiel qui est similaire pour les référentiels hébergés sur un serveur.Il ya également une section expliquant comment mettre en place un serveur.

La section Chapitre 4, *Guide d'utilisation quotidienne* est la plus importante puisqu'elle explique toutes les fonctionnalités principales de TortoiseSVN et comment les utiliser. Elle est sous forme de tutoriel qui indique comment extraire une copie de travail, la modifier, livrer les changements effectués, etc. Il traite de sujets de plus en plus avancés.

Chapitre 5, *Le programme SubWCRev* est un programme installé avec TortoiseSVN permettant d'extraire des informations de votre copie de travail et de les écrire dans un fichier. Ce qui est utile pour ajouter des informations de compilation à vos projets.

La section Annexe B, *Comment faire pour...* répond à quelques questions courantes concernant les tâches qui ne sont pas expliquées ailleurs.

La section Annexe D, *Automatiser TortoiseSVN* montre comment les boîtes de dialogue de TortoiseSVN peuvent être appelées en ligne de commande. C'est utile pour faire des scripts interactifs.

La Annexe E, *Référence croisée de l'interface en ligne de commande* fait le lien entre les commandes de TortoiseSVN et leurs équivalents en ligne de commande du client Subversion `svn.exe`.

3. TortoiseSVN est gratuit !

TortoiseSVN est gratuit. Vous n'avez pas à payer pour l'avoir, et vous pouvez l'utiliser comme vous le souhaitez. Il est développé sous licence GP (GPL).

TortoiseSVN est un projet Open Source. Cela signifie que vous avez un accès total au code source de ce programme. Vous pouvez y accéder avec ce lien *http://tortoisesvn.tigris.org/svn/tortoisesvn/*. L'utilisateur est `guest`, et laissez le mot de passe vide. La version la plus récente (sur laquelle nous travaillons actuellement) est dans `/trunk/`, les versions publiées sont situées dans `/tags/`.

4. Communauté

TortoiseSVN et Subversion sont tout deux développés par une communauté de personnes qui travaillent sur ces projets. Elles viennent de pays différents dans le monde entier et se sont rassemblées pour créer de merveilleux programmes.

5. Remerciements

Tim Kemp
: pour avoir fondé le projet TortoiseSVN

Stefan Küng
: pour le travail difficile pour que TortoiseSVN devienne ce qu'il est maintenant

Lübbe Onken
: pour les belles icônes, le logo, la chasse aux bugs, la traduction et le soin apporté à la documentation

Simon Large
: pour aider à documenter et chasser les bugs

Le manuel de Subversion
: pour l'introduction grandiose à Subversion et son chapitre 2 que nous avons copié ici

Le projet Tigris Style
: pour certains des styles réutilisés dans cette documentation

Nos contributeurs
: pour les patches, les rapports de bug et les nouvelles idées, et pour avoir aidé les autres en répondant aux questions sur notre mailing list.

Nos donateurs
: pour les nombreuses heures de bonheur avec la musique qu'ils nous ont envoyé

6. Terminologie utilisée dans ce document

Pour rendre la lecture des documents plus facile, les noms de tous les écrans et des menus de TortoiseSVN sont marqués dans une police différente. La boîte de dialogue Journal par exemple.

Un choix de menu est indiqué par une flèche. TortoiseSVN → Voir le journal veut dire : sélectionnez *Voir le journal* à partir du menu contextuel *TortoiseSVN*.

Quand un menu contextuel local apparaît dans une des boîtes de dialogue de TortoiseSVN, on le montre comme cela : Menu contextuel → Enregistrer sous...

Les boutons de l'interface utilisateur sont indiqués comme ceci : Appuyer sur OK pour continuer.

Les actions utilisateur sont indiquées en utilisant une police en gras. **Alt+A** : appuyez sur la touche **A** en maintenant la touche **Alt** de votre clavier enfoncée. Glisser click droit : appuyez sur le bouton droit de la souris et en le maintenant enfoncé *faites glisser* les éléments vers le nouvel emplacement.

La sortie système et l'entrée au clavier sont indiquées aussi avec une police `différente`.

Important

Les notes importantes sont marquées avec une icône.

Astuce

Astuces pour vous rendre la vie plus facile.

Attention

Endroits où vous devez faire attention à ce que vous faites.

Avertissement

Quand un soin extrême doit être pris, une corruption de données ou d'autres choses désagréables peuvent arriver si ces avertissements sont ignorés.

Chapitre 1. Introduction

Le contrôle de version est l'art de gérer les changements de l'information. Cela a longtemps été un outil critique pour les programmeurs, qui passent typiquement leur temps à faire de petites modifications au logiciel et à défaire ensuite ces changements le lendemain. Imaginez une équipe de ces programmeurs travaillant concurremment - et peut-être même simultanément sur les même fichiers ! - et vous pouvez voir pourquoi un bon système est nécessaire pour *gérer le chaos potentiel*.

1.1. Qu'est-ce que TortoiseSVN ?

TortoiseSVN est un client open-source gratuit pour le système de contrôle de version *Subversion*. C'est-à-dire TortoiseSVN gère des fichiers et des répertoires à travers le temps. Les fichiers sont stockés dans un *référentiel* central. Le référentiel ressemble beaucoup à un serveur de fichiers ordinaire, sauf qu'il se rappelle chaque changement jamais fait à vos fichiers et répertoires. Cela vous permet de récupérer les versions précédentes de vos fichiers et examiner l'historique de comment et quand vos données ont changé. C'est pourquoi beaucoup de personnes pensent que Subversion et les systèmes de contrôle de version en général sont une sorte de « machine à remonter le temps ».

Quelques systèmes de contrôle de version sont aussi des systèmes de gestion de configuration logicielle (GCL). Ces systèmes sont spécifiquement conçus pour gérer des arborescences de code source et ont beaucoup de fonctionnalités spécifiques au développement de logiciel - comme la compréhension de langages de programmation en natif, ou des outils d'approvisionnement pour construire le logiciel. Subversion, cependant, n'est pas un de ces systèmes ; c'est un système général qui peut être utilisé pour gérer *n'importe quelle* collection de fichiers, y compris du code source.

1.2. L'historique de TortoiseSVN

En 2002, Tim Kemp a constaté que Subversion était un très bon système de contrôle de version, mais il lui manquait un bon client avec une interface graphique. L'idée d'un client Subversion comme une intégration du shell de Windows a été inspirée par le client semblable pour CVS nommé TortoiseCVS.

Tim a étudié le code source de TortoiseCVS et l'a utilisé comme base pour TortoiseSVN. Il a alors commencé le projet, a enregistré le domaine `tortoisesvn.org` et a mis le code source en ligne. Pendant ce temps, Stefan Küng cherchait un bon système de contrôle de version gratuit et a trouvé Subversion et le source de TortoiseSVN. Puisque TortoiseSVN n'était toujours pas prêt à l'emploi il a alors rejoint le projet et a commencé à programmer. Bientôt il a réécrit la plupart du code existant et a commencé à ajouter des commandes et des fonctionnalités, jusqu'au point où rien n'est resté du code original.

Comme Subversion est devenu plus stable, il a attiré de plus en plus d'utilisateurs qui ont aussi commencé à utiliser TortoiseSVN comme leur client Subversion. La base utilisateur a grandi rapidement (et grandit toujours chaque jour). C'est à ce moment que Lübbe Onken s'est proposé d'aider avec des icônes agréables et un logo pour TortoiseSVN. Et il s'occupe du site Web et gère la traduction.

1.3. Les fonctionnalités de TortoiseSVN

Qu'est-ce qui fait de TortoiseSVN un si bon client Subversion ? Voici une courte liste des fonctionnalités.

Intégration dans le shell

TortoiseSVN s'intègre uniformément dans le shell Windows (c'est-à-dire l'explorateur). Cela signifie que vous pouvez continuer à travailler avec les outils avec lesquels vous êtes déjà familiers. Et vous n'avez pas à changer d'application à chaque fois que vous avez besoin des fonctionnalités du contrôle de version !

Et vous n'êtes même pas obligés d'utiliser l'explorateur Windows. Les menus contextuels de TortoiseSVN marchent dans beaucoup d'autres gestionnaires de fichiers et dans la boîte de dialogue Fichier/Ouvrir qui est commune à la plupart des applications Windows standards. Vous devriez, cependant, tenir compte que TortoiseSVN est intentionnellement développé comme extension pour l'explorateur Windows. Ainsi il est possible que dans d'autres applications l'intégration ne soit pas aussi complète et le recouvrement d'icônes peut ne pas s'afficher par exemple.

Recouvrement d'icônes
: Le statut de chaque fichier et de chaque répertoire versionnés est indiqué par des petites icônes de recouvrement. De cette façon vous pouvez voir tout de suite quel est le statut de votre copie de travail.

Accès facile aux commandes de Subversion
: Toutes les commandes de Subversion sont disponibles à partir du menu contextuel de l'explorateur. TortoiseSVN y ajoute son propre sous-menu.

Puisque TortoiseSVN est un client Subversion, nous voudrions aussi vous montrer certaines des fonctionnalités de Subversion :

Répertoires versionnés
: CVS suit seulement à la trace l'histoire de fichiers individuels, mais Subversion met en oeuvre un système de fichiers « virtuel » versionné qui suit à la trace les changements sur des arborescences entières à travers le temps. Les fichiers *et* les répertoires sont versionnés. En conséquence, il y a du coté client de vraies commandes **déplacer** et **copier** qui fonctionnent sur les fichiers et les répertoires.

Livraisons atomiques
: Une livraison va sur le référentiel complètement, ou pas du tout. Cela permet aux développeurs de construire et livrer les changements comme des morceaux logiques.

Metadonnées versionnées
: Chaque fichier et chaque répertoire a un jeu invisible de « propriétés » attachées. Vous pouvez inventer et stocker n'importe quelle paire arbitraire clef/valeur que vous souhaitez. Les propriétés sont versionnées dans le temps, comme le contenu du fichier.

Choix de couches réseau
: Subversion a une notion abstraite de l'accès au référentiel, le rendant facile à mettre en oeuvre à travers de nouveaux mécanismes de réseau. Le serveur réseau « avancé » de Subversion est un module pour le serveur Web Apache, qui utilise une variante de HTTP appelée WebDAV/DeltaV. Cela donne un grand avantage à Subversion en stabilité et en interopérabilité et fournit des différentes fonctionnalités clés gratuitement : authentification, autorisation, compression de fil et navigation de référentiel, par exemple. Un processus de serveur Subversion plus petit, autonome est aussi disponible. Ce serveur utilise un protocole personnalisé qui peut être facilement tunnelé par ssh.

Gestion cohérente des données
: Subversion exprime les différences de fichier en utilisant un algorithme de différenciation binaire, qui travaille identiquement sur les fichiers textes (lisibles par l'homme) et les fichiers binaires (illisibles par l'homme). Les deux types de fichiers sont stockés également compressés dans le référentiel, et les différences sont transmises dans les deux directions à travers le réseau.

Embranchements et étiquetages efficaces
: Le coût de l'embranchement et de l'étiquetage n'a pas besoin d'être proportionnel à la taille de projet. Subversion crée des branches et des étiquettes en copiant simplement le projet, en utilisant un mécanisme semblable à un lien dur. Ainsi ces opérations prennent seulement un temps très petit, constant et un espace très petit dans le référentiel.

Hackabilité
: Subversion n'a aucun bagage historique ; il est mis en oeuvre comme une collection de bibliothèques C partagées avec des API bien définies. Cela fait que Subversion est extrêmement maintenable et utilisable par d'autres applications et d'autres langages.

1.4. Installer TortoiseSVN

1.4.1. Configuration requise

TortoiseSVN fonctionne sur Win2000 SP2, WinXP ou supérieur. Windows 98, Windows ME et Windows NT4 ne sont plus supportés depuis TortoiseSVN 1.2.0, mais vous pouvez toujours télécharger les versions précédentes si vous en avez vraiment besoin.

Si vous rencontrez des problèmes pendant ou après l'installation de TortoiseSVN, veuiller d'abord vous référer à Annexe A, *Foire aux questions (FAQ)*.

1.4.2. Installation

TortoiseSVN est livré avec un programme d'installation facile à utiliser. Double cliquez sur le fichier d'installation et suivez les indications. Le programme d'installation s'occupera du reste.

Important

Vous devez être administrateur pour installer TortoiseSVN.

1.4.3. Packs de langue

L'interface utilisateur de TortoiseSVN a été traduite dans beaucoup de langues différentes, donc vous devez être capables de télécharger un pack de langue qui réponde à vos besoins. Vous pouvez trouver les packs de langue sur notre *page du statut des traductions* [http://tortoisesvn.net/translation_status]. Et s'il n'y a encore de pack de langue disponible, pourquoi ne pas rejoindre l'équipe et soumettre votre propre traduction ;-)

Chaque pack de langue est empaqueté comme un installeur `.exe`. Exécutez juste le programme d'installation et suivez les instructions. La prochaine fois que vous redémarrez, la traduction sera disponible.

1.4.4. Vérificateur d'orthographe

TortoiseSVN inclut un vérificateur d'orthographe qui vous permet de vérifier vos commentaires de livraison. C'est particulièrement utile si le la langue du projet n'est pas votre langue maternelle. Le vérificateur d'orthographe utilise les mêmes fichiers de dictionnaire que *OpenOffice* [http://openoffice.org] et *Mozilla* [http://mozilla.org].

L'installeur ajoute automatiquement les dictionnaires d'anglais américains et britanniques. Si vous voulez d'autres langues, l'option la plus facile est d'installer simplement un des packs de langue de TortoiseSVN. Cela installera les fichiers de dictionnaire appropriés en même temps que l'interface utilisateur TortoiseSVN locale. La prochaine fois que vous redémarrez, le dictionnaire sera disponible aussi.

Ou vous pouvez installer les dictionnaires vous-même. Si vous avez OpenOffice ou Mozilla installés, vous pouvez copier ces dictionnaires, qui se trouvent dans les dossiers d'installation de ces applications. Autrement, vous devez télécharger les fichiers de dictionnaire requis depuis *http://wiki.services.openoffice.org/wiki/Dictionaries*

Une fois que vous avez les fichiers de dictionnaire, vous devez probablement les renommer pour que les noms de fichier contiennent seulement les caractères locaux. Exemple :

- `fr_FR.aff`
- `fr_FR.dic`

Ensuite copiez les simplement dans le sous-dossier `bin` du dossier d'installation de TortoiseSVN. Normalement ce sera `C:\Program Files\TortoiseSVN\bin`. Si vous ne voulez pas encombrer le sous-dossier `bin`, vous pouvez placer vos fichiers de vérification d'orthographe dans `C:\Program Files\TortoiseSVN\Languages` à la place. Si ce dossier n'est pas là, vous devez le créer d'abord. La prochaine fois vous démarrez TortoiseSVN, le vérificateur d'orthographe sera disponible.

Si vous installez plusieurs dictionnaires, TortoiseSVN utilise ces règles pour choisir lequel utiliser.

1. Vérifier le réglage `tsvn:projectlanguage`. Référez-vous à Section 4.17, « Configuration des projets » pour des informations concernant les propriétés de projet.

2. Si aucune langue de projet n'est indiquée, ou si cette langue n'est pas installée, essayer la langue correspondant aux options régionales de Windows.

3. Si les options régionales exactes de Windows ne marchent pas, essayer la langue de « Base », i.e. `de_CH` (Allemand Suisse) devient `de_DE` (Allemand).

4. Si aucune des règles ci-dessus ne marche, alors la langue par défaut est l'anglais, qui est inclus avec l'installation standard.

Chapitre 2. Concepts de base du contrôle de version

Ce chapitre est une version légèrement modifiée du même chapitre dans le manuel de Subversion. Une version en ligne du manuel de Subversion est disponible ici : *http://svnbook.red-bean.com/*.

Ce chapitre est une introduction courte, occasionnelle à Subversion. Si le contrôle de version est nouveau pour vous, ce chapitre est certainement pour vous. Nous commençons par une discussion sur les concepts généraux du contrôle de version, nous ferons notre chemin vers les idées spécifiques derrière Subversion et nous montrons quelques exemples simples de Subversion en utilisation.

Bien que les exemples dans ce chapitre montrent des gens partageant des collections de code source de programme, gardez à l'esprit que Subversion peut gérer n'importe quel sorte de collection de fichier - il n'est pas limité à l'aide de programmeurs.

2.1. Le référentiel

Subversion est un système centralisé pour partager l'information. Son coeur est un *référentiel*, qui est un dépôt central de données. Le référentiel stocke l'information sous forme d'une *arborescence de système de fichiers* - une hiérarchie typique de fichiers et de répertoires. N'importe quel nombre de *clients* se connecte au référentiel et ensuite lit ou écrit ces fichiers. En écrivant des données, un client rend l'information disponible aux autres ; en lisant des données, le client reçoit l'information des autres.

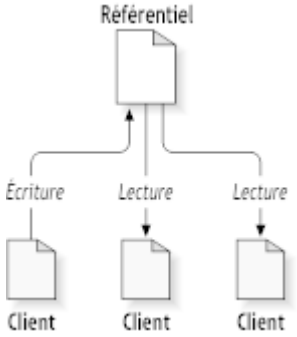


Figure 2.1. Un système Client/Serveur typique

Alors pourquoi est-ce si intéressant ? Jusqu'ici, cela ressemble à la définition d'un serveur de fichiers typique. Et en effet, le référentiel *est* une sorte de serveur de fichiers, mais il n'est pas de votre genre habituel. Ce qui rend le référentiel de Subversion spécial est qu'*il se rappelle de chaque changement* jamais écrit : chaque changement de chaque fichier, et même les changements de l'arborescence des répertoires elle-même, comme l'ajout, la suppression et le réarrangement des fichiers et des répertoires.

Quand un client lit des données du référentiel, il voit normalement seulement la dernière version de l'arborescence. Mais le client peut aussi voir les états *précédents*. Par exemple, un client peut poser des questions historiques comme, « que contenait ce répertoire mercredi dernier ? », ou « qui sont les dernières personnes à avoir changé ce fichier et quels changements ont-elles fait ? » C'est ce genre de questions qui sont au coeur de n'importe quel *système de contrôle de version* : des systèmes qui sont conçus pour enregistrer et suivre les changements des données au cours du temps.

2.2. Modèles de gestion de version

Tous les systèmes de contrôle de version doivent résoudre le même problème fondamental : comment le système permettra-t-il aux utilisateurs de partager l'information, mais les empêchera accidentellement de

se marcher sur les pieds ? Il est trop facile pour les utilisateurs d'écraser accidentellement les changements de chacun sur le référentiel.

2.2.1. Le problème du partage de fichier

Considérons ce scénario : supposons que nous avons deux collaborateurs, Harry et Sally. Ils décident chacun d'éditer le même fichier du référentiel en même temps. Si Harry sauvegarde ses changements sur le référentiel d'abord, il est possible qu'ensuite (quelques moments plus tard) Sally puisse accidentellement les écraser avec sa propre nouvelle version du fichier. Tandis que la version d'Harry du fichier ne sera pas perdue pour toujours (parce que le système se rappelle chaque changement), les changements qu'Harry a fait *ne seront pas* dans la version plus récente du fichier de Sally, parce qu'elle n'a jamais vu les changements d'Harry pour commencer. Le travail d'Harry est effectivement encore perdu - ou du moins manquant de la dernière version du fichier - et probablement par accident. C'est certainement une situation que nous voulons éviter!

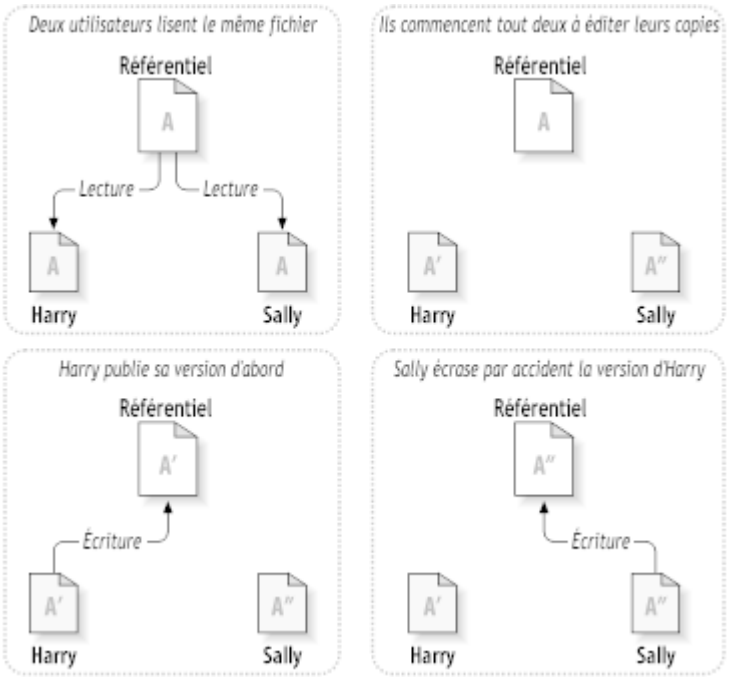


Figure 2.2. Le problème à éviter

2.2.2. La solution Verrouiller-Modifier-Déverrouiller

Beaucoup de systèmes de contrôle de version utilisent le modèle *verrouiller-modifier-déverrouiller* pour aborder ce problème, qui est une solution très simple. Dans un tel système, le référentiel ne permet qu'à une seule personne de changer un fichier à la fois. D'abord Harry doit *verrouiller* le fichier avant qu'il ne puisse commencer à y faire des changements. Le verrouillage d'un fichier s'apparente alors à l'emprunt d'un livre dans une bibliothèque ; si Harry a verrouillé un fichier, alors Sally ne peut pas le modifier. Si elle essaye de verrouiller le fichier, le référentiel refusera la requête. Tout qu'elle peut faire est lire le fichier et attendre qu'Harry finisse ses changements et relâche le verrou. Dès qu'Harry déverrouille le fichier, son tour est fini, et Sally peut alors prendre son tour en le verrouillant et en y apportant ses modifications.

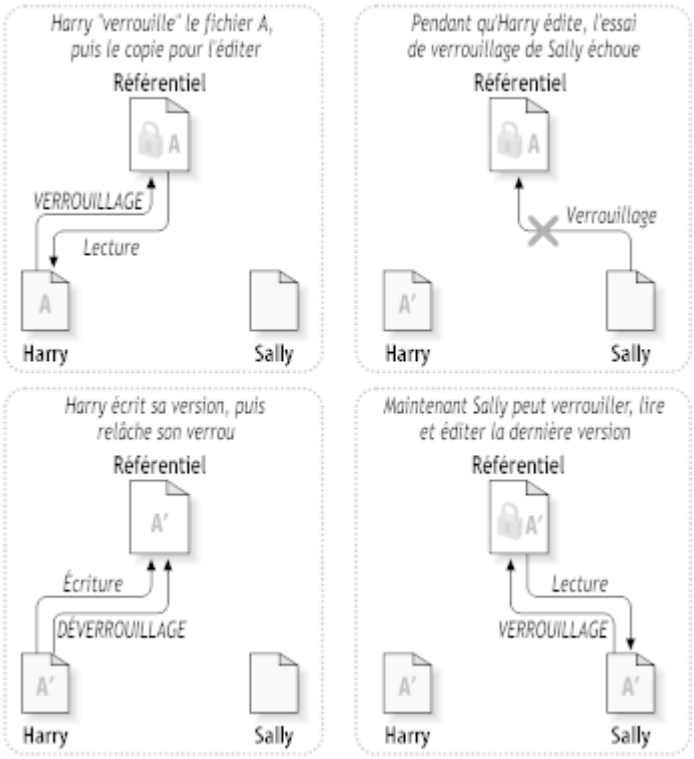


Figure 2.3. La solution Verrouiller-Modifier-Déverrouiller

Le problème avec le modèle "verrouiller-modifier-déverrouiller" est qu'il est un peu restrictif et devient souvent un barrage pour les utilisateurs :

- *Le verrouillage peut causer des problèmes administratifs.* Parfois Harry verrouillera un fichier et ensuite l'oubliera. En attendant, parce que Sally attend toujours pour éditer le fichier, ses mains sont liées. Et ensuite Harry va en vacances. Maintenant Sally doit appeler un administrateur pour relâcher le verrou d'Harry. La situation se finit en causant beaucoup de retard inutile et de temps gaspillé.

- *Le verrouillage peut causer une sérialisation inutile.* Et si Harry édite le début d'un fichier texte et Sally veut simplement éditer la fin du même fichier ? Ces changements ne se chevauchent pas du tout. Ils pourraient facilement éditer le fichier simultanément et aucun grand mal n'arriverait, à supposer que les changements aient été correctement fusionnés ensemble. Ils n'ont aucun besoin de prendre leur tour dans cette situation.

- *Le verrouillage peut créer un faux sentiment de sécurité.* Disons qu'Harry verrouille et édite le fichier A, tandis que Sally verrouille et édite le fichier B en même temps. Mais supposons qu'A et B dépendent l'un de l'autre et les changements faits à chacun sont sémantiquement incompatibles. Soudainement A et B ne marchent désormais plus ensemble. Le système de verrouillage était impuissant à empêcher le problème - encore il a fourni d'une façon ou d'une autre un sentiment de fausse sécurité. C'est facile pour Harry et Sally de s'imaginer qu'en verrouillant les fichiers, chacun commence une tâche sûre, isolée et les interdit ainsi de discuter de leurs changements incompatibles dès le début.

2.2.3. La solution Copier-Modifier-Fusionner

Subversion, CVS et les autres systèmes de contrôle de version utilisent un modèle *copier-modifier-fusionner* comme une alternative au verrouillage. Dans ce modèle, le client de chaque utilisateur lit le référentiel et crée une *copie de travail* personnelle du fichier ou du projet. Les utilisateurs travaillent alors en parallèle, modifiant leurs copies privées. Finalement, les copies privées sont fusionnées ensemble dans une version nouvelle, finale. Le système de contrôle de version aide souvent avec la fusion, mais en fin de compte un être humain est responsable pour qu'elle se produise correctement.

Voici un exemple. Disons qu'Harry et Sally ont chacun une copie de travail du même projet, extrait du référentiel. Ils travaillent en même temps et modifient localement le même fichier A. Sally sauvegarde

ses changements dans le référentiel d'abord. Quand Harry essaie de sauvegarder ses changements, le référentiel l'informe que son fichier A est *périmé*. Autrement dit, dans le référentiel, ce fichier A a été modifié depuis qu'il a été extrait. Donc Harry demande à son client de *fusionner* les nouveaux changements du fichier A du référentiel avec sa copie de travail. Il y a des chances que les changements de Sally ne se chevauchent pas avec les siens ; ainsi une fois qu'il a les deux changements intégrés, il sauvegarde sa copie de travail sur le référentiel.

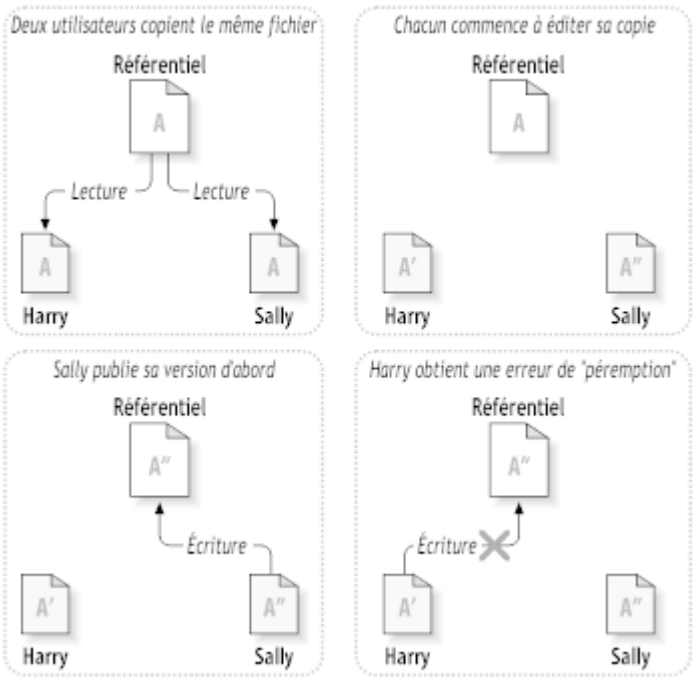


Figure 2.4. La solution Copier-Modifier-Fusionner

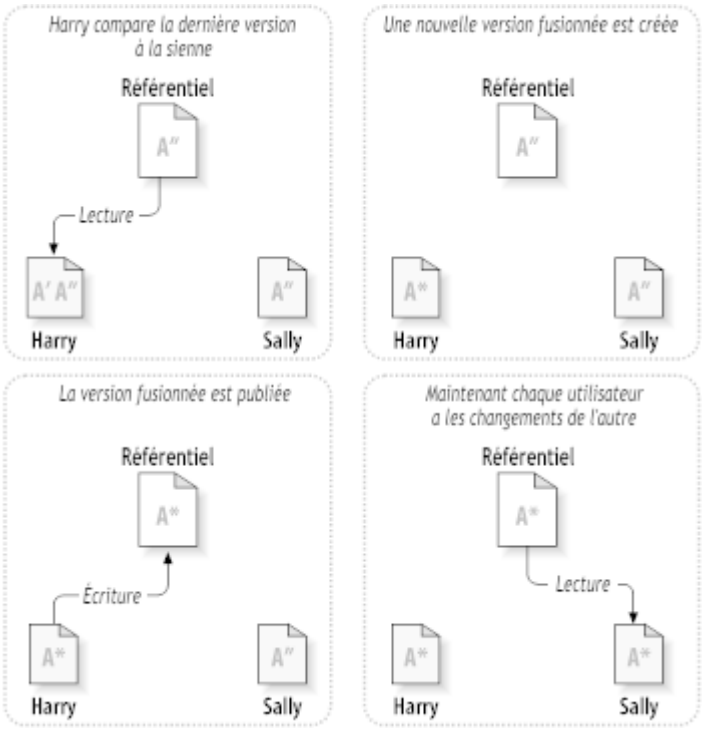


Figure 2.5. ...Suite du modèle Copier-Modifier-Fusionner

Mais que se passe-t-il si les changements de Sally *chevauchent* les changements d'Harry ? Que se passe-t-il alors ? Cette situation est appelée un *conflit* et habituellement, ce n'est pas vraiment un problème. Quand Harry demande à son client de fusionner les derniers changements du référentiel vers sa copie de travail, sa copie du fichier A est marquée d'une façon ou d'une autre comme étant dans un état de conflit : il sera capable de voir les deux jeux de changements en conflit et choisira manuellement entre eux. Notez que le logiciel ne peut pas résoudre automatiquement les conflits ; seuls les êtres humains sont capables de comprendre et de faire les choix intelligents nécessaires. Une fois qu'Harry a manuellement résolu les changements se chevauchant (peut-être en discutant du conflit avec Sally !), il peut sans risque sauvegarder le fichier fusionné sur le référentiel.

Le modèle copier-modifier-fusionner peut sembler un peu chaotique, mais en pratique, il fonctionne de manière extrêmement fluide. Les utilisateurs peuvent travailler en parallèle, n'ayant jamais à s'attendre. Quand ils travaillent sur les mêmes fichiers, il s'avère que la plupart de leurs changements simultanés ne se chevauchent pas du tout ; les conflits sont peu fréquents. Et le temps que cela prend pour résoudre des conflits est moindre que le temps perdu par un système de verrouillage.

À la fin, tout cela se réduit à un facteur critique : la communication entre les utilisateurs. Quand les utilisateurs communiquent mal, les conflits tant syntaxiques que sémantiques augmentent. Aucun système ne peut forcer les utilisateurs à communiquer parfaitement et aucun système ne peut détecter les conflits sémantiques. Ainsi il n'y a aucune raison d'être apaisé par une fausse promesse qu'un système de verrouillage empêchera d'une façon ou d'une autre des conflits ; en pratique, le verrouillage semble inhiber la productivité plus qu'autre chose.

Il y a une situation commune où le modèle verrouiller-modifier-déverrouiller s'en sort mieux et c'est quand vous avez des fichiers qui ne sont pas fusionnables. Par exemple, si votre référentiel contient quelques images graphiques et deux personnes modifient une image en même temps, il n'y a aucun moyen de fusionner ces changements. Soit Harry, soit Sally perdra ses changements.

2.2.4. Que fait Subversion ?

Subversion utilise la solution copier-modifier-fusionner par défaut et dans des nombreux cas c'est tout ce dont vous aurez jamais besoin. Cependant, à partir de la version 1.2, Subversion supporte aussi le verrouillage de fichier, si vous avez des fichiers non-fusionnables, ou si vous êtes simplement forcés à une politique de verrouillage par le management, Subversion fournira encore les fonctionnalités dont vous avez besoin.

2.3. Subversion en action

2.3.1. Copies de travail

Vous a déjà lu au sujet des copies de travail ; maintenant nous allons vous montrer comment le client Subversion les crée et les utilise.

Une copie de travail de Subversion est une arborescence de répertoires ordinaire sur votre système local, contenant une collection de fichiers. Vous pouvez éditer ces fichiers comme vous le souhaitez, et si ce sont des fichiers de code source, vous pouvez compiler votre programme de la façon habituelle. Votre copie de travail est votre propre secteur de travail privé : Subversion n'incorporera jamais les changements d'autres personnes, ni ne rendra vos propres changements disponibles aux autres, jusqu'à ce que vous lui disiez explicitement de le faire.

Après avoir fait des changements dans votre copie de travail et avoir vérifié qu'ils fonctionnent correctement, Subversion vous fournit des commandes pour *publier* vos changements et ainsi les rendre disponibles aux les autres personnes travaillant avec vous sur le projet. Si d'autres personnes publient leurs changements dans le référentiel, Subversion vous fournit des commandes pour fusionner ces changements dans votre répertoire de travail.

Une copie de travail contient aussi quelques fichiers supplémentaires, créés et entretenus par Subversion, pour l'aider à effectuer ces commandes. Particulièrement chaque répertoire dans votre copie de travail

contient un sous-répertoire nommé `.svn`, aussi connu comme le *répertoire administratif* de la copie de travail. Les fichiers de chaque répertoire administratif aident Subversion à reconnaître quels fichiers contiennent des changements non publiés et quels fichiers sont périmés par rapport au travail des autres.

Un référentiel typique de Subversion contient souvent les fichiers (ou le code source) pour plusieurs projets ; d'habitude, chaque projet est un sous-répertoire dans l'arborescence du système de fichiers du référentiel. Dans cette optique, la copie de travail d'un utilisateur correspondra d'habitude à un sous-arbre particulier du référentiel.

Par exemple, supposons que vous avez un référentiel contenant deux projets d'application.

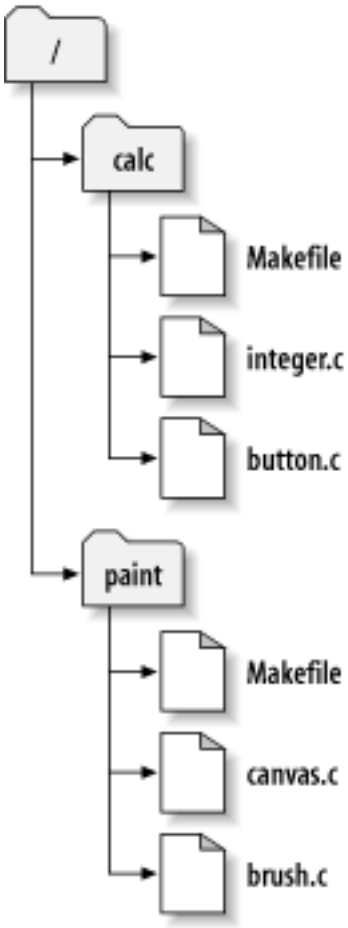


Figure 2.6. Le système de fichiers du référentiel

En d'autres termes, la racine du référentiel a deux sous-répertoires : `paint` et `calc`.

Pour obtenir une copie de travail, vous devez *extraire* une sous-arborescence du référentiel. (Le terme *extraire* peut faire penser à un quelconque verrouillage ou à une réservation des ressources, mais il n'en est rien ; il crée simplement une copie privée du projet pour vous).

Supposons que vous faites des changements sur `button.c`. Puisque le répertoire `.svn` se rappelle la date de modification et le contenu original du fichier, Subversion peut dire que vous avez modifié le fichier. Cependant, Subversion ne rend pas vos changements publics jusqu'à ce que vous le lui disiez explicitement. L'acte de publier vos changements est plus communément appelé *livrer* (ou *valider*) les changements au référentiel.

Pour publier vos changements aux autres, vous pouvez utiliser la commande de Subversion **livrer**.

Maintenant vos changements sur `button.c` ont été livrés au référentiel ; si un autre utilisateur extrait une copie de travail de `/calc`, il verra vos changements dans la dernière version du fichier.

Supposons que vous avez une collaboratrice, Sally, qui a extrait une copie de travail de `/calc` en même temps que vous. Quand vous livrez votre changement sur `button.c`, la copie de travail de Sally est laissée inchangée ; Subversion modifie seulement les copies de travail à la demande de l'utilisateur.

Pour actualiser son projet, Sally peut demander à Subversion de *mettre à jour* sa copie de travail, en utilisant la commande de Subversion **mettre à jour**. Cela incorporera vos changements dans sa copie de travail, en même temps que d'autres qui ont été livrés depuis qu'elle l'a extrait.

Notez que Sally n'a pas dû spécifier les fichiers à mettre à jour; Subversion utilise l'information dans le répertoire `.svn` et la nouvelle information dans le référentiel, pour décider quels fichiers doivent être actualisés.

2.3.2. URL de référentiel

Les référentiels de Subversion peuvent être accédés par beaucoup de méthodes différentes - sur le disque local, ou par des protocoles de réseau divers. Un emplacement de référentiel, cependant, est toujours une URL. Le schéma d'URL indique la méthode d'accès :

Schéma	Méthode d'accès
`file://`	Accès direct au référentiel sur disque local ou réseau.
`http://`	Accès via le protocole WebDAV à un serveur Apache avec Subversion.
`https://`	Même chose que `http://`, mais avec cryptage SSL.
`svn://`	Accès TCP/IP non authentifié via un protocole personnalisé à un serveur `svnserve`.
`svn+ssh://`	Accès TCP/IP authentifié, crypté via un protocole personnalisé à un serveur `svnserve`.

Tableau 2.1. URL d'accès au référentiels

Pour la plupart, les URL de Subversion utilisent la syntaxe standard, autorisant la définition des noms serveur et des numéros de port dans l'URL. La méthode d'accès `file://` est normalement utilisée pour les accès locaux, bien qu'elle puisse être utilisée avec des chemins UNC pour un hôte non local. L'URL prend donc la forme `file://nomhote/chemin/vers/referentiel`. Pour la machine locale, la partie `nomhote` de l'URL doit être soit absente, soit `localhost`. Pour cette raison, les chemins locaux apparaissent normalement avec trois slashs, `file:///chemin/vers/referentiel`.

Ainsi, les utilisateurs du système de fichier `file:` sur les plate-formes Windows devront utiliser une syntaxe « standard » non-officielle pour avoir accès aux référentiels qui sont sur la même machine, mais sur un disque différent du disque de travail. Les deux syntaxes de d'URL suivantes marcheront ; `X` est le disque sur lequel est hébergé le référentiel :

```
file:///X:/chemin/vers/référentiel
...
file:///X|/chemin/vers/référentiel
...
```

Notez qu'une URL utilise des slashs ordinaires bien que la forme native d'un chemin (non-URL) utilise des antislashs sous Windows.

Vous pouvez sans risque avoir accès à un référentiel FSFS via un partage réseau, mais vous *ne pouvez pas* avoir accès à un référentiel BDB de cette façon.

Avertissement

Ne créez pas ou n'accédez pas à un référentiel Berkeley DB sur un partage réseau. Il *ne peut pas* exister sur un système de fichiers distant. Même si vous faites mapper le disque

> réseau à une lettre de disque. Si vous essayez d'utiliser Berkeley DB sur un partage réseau, les résultats sont imprévisibles - vous pouvez voir des erreurs mystérieuses tout de suite, ou cela peut s'écouler des mois avant que vous ne découvriez que votre base de données du référentiel est subtilement corrompue.

2.3.3. Révisions

Une opération **svn commit** peut publier les changements de n'importe quel nombre de fichiers et de répertoires comme une seule transaction atomique. Dans votre copie de travail, vous pouvez changer le contenu des fichiers, créer, supprimer, renommer et copier des fichiers et des répertoires et ensuite livrer le jeu complet de changements comme une unité.

Dans le référentiel, chaque livraison est traitée comme une transaction atomique : tous les changements de la livraison ont lieu, ou aucun n'a lieu. Subversion essaye de conserver cette atomicité face aux plantages du programme, pannes système, problèmes de réseau et autres actions utilisateur.

Chaque fois que le référentiel accepte une livraison, cela crée un nouvel état de l'arborescence du système de fichiers, appelé une *révision*. À chaque révision est assignée un entier naturel unique, plus grand que le numéro de la révision précédente d'une unité. La révision initiale d'un référentiel récemment créé est numérotée zéro et ne consiste en rien d'autre qu'un répertoire racine vide.

Une façon agréable de visualiser le référentiel est une série d'arbres. Imaginez un tableau de numéros de révision, commençant à 0, s'étirant de gauche à droite. Chaque numéro de révision a un arbre du système de fichiers s'accrochant au-dessous de lui et chaque arbre est un « instantané » de la façon à laquelle le référentiel ressemble après chaque livraison.

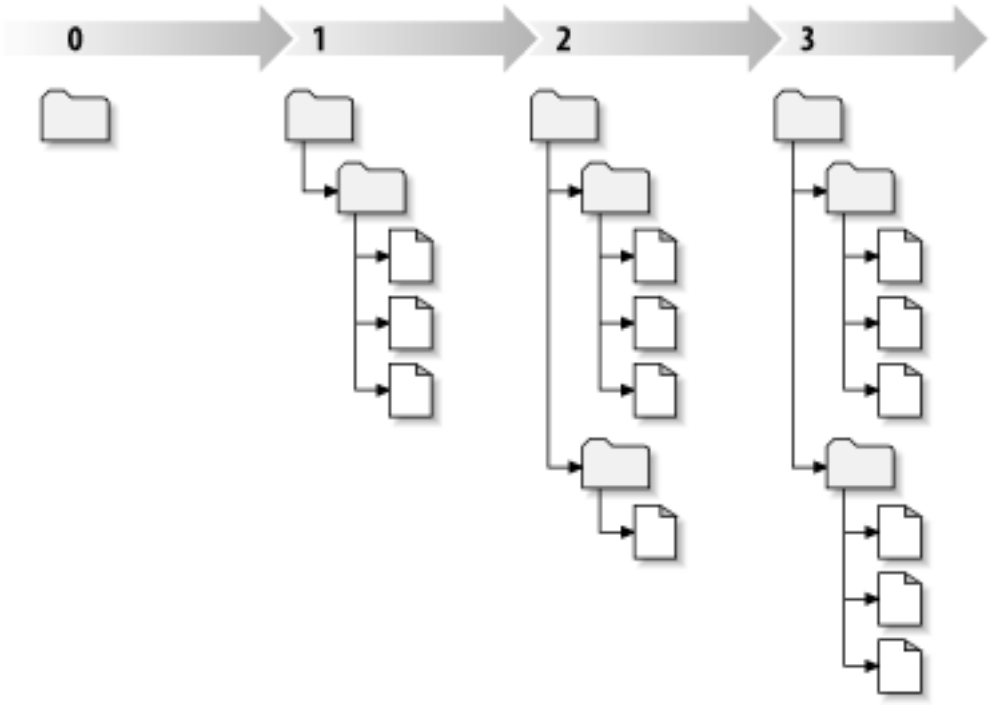


Figure 2.7. Le référentiel

> **Numéros de révision globaux**
>
> À la différence de beaucoup d'autres systèmes de contrôle de version, les numéros de révision de Subversion s'appliquent à *toute l'arborescence*, et non à des fichiers individuels. Chaque numéro de révision sélectionne un arbre entier, un état particulier du référentiel après un certain changement livré. Une autre façon d'y penser est que cette révision N représente l'état du système de fichiers du

> référentiel après que la Nième livraison. Quand un utilisateur de Subversion parle de ``la révision 5 de `foo.c`", il veut dire vraiment ``foo.c comme il apparaît à la révision 5." Remarquez qu'en général, les révisions N et M d'un fichier ne diffèrent *pas* nécessairement !

Il est important de noter que les copies de travail ne correspondent pas toujours à une seule révision dans le référentiel ; elles peuvent contenir des fichiers de plusieurs révisions différentes. Par exemple, supposez que vous extrayiez une copie de travail d'un référentiel dont la révision la plus récente est 4 :

```
calc/Makefile:4
     integer.c:4
     button.c:4
```

À l'heure actuelle, ce répertoire de travail correspond exactement à la révision 4 dans le référentiel. Cependant, supposez que vous faites un changement sur `button.c` et que vous livrez ce changement. En admettant qu'aucune autre livraison n'ait eu lieue, votre livraison créera la révision 5 du référentiel, et votre copie de travail ressemblera maintenant à cela :

```
calc/Makefile:4
     integer.c:4
     button.c:5
```

Supposons que, à ce point, Sally livre une modification sur `integer.c`, créant la révision 6. Si vous utilisez **svn update** pour actualiser votre copie de travail, elle ressemblera alors à ceci :

```
calc/Makefile:6
     integer.c:6
     button.c:6
```

Les changements de Sally sur `integer.c` apparaîtront dans votre copie de travail et votre changement seront toujours présent dans `button.c`. Dans cet exemple, le texte `Makefile` est identique dans des révisions 4, 5 et 6, mais Subversion marquera votre copie de travail de `Makefile` avec la révision 6 pour indiquer qu'il est toujours d'actualité. Ainsi, après avoir fait une mise à jour propre au sommet de votre copie de travail, elle correspondra généralement exactement à une révision dans le référentiel.

2.3.4. Comment les copies de travail suivent le référentiel

Pour chaque fichier dans un répertoire de travail, Subversion enregistre deux informations essentielles dans le secteur administratif `.svn/` :

- sur quelle révision votre fichier de travail est basé (cela s'appelle la *révision de travail* du fichier), et
- un enregistrement d'horodatage quand la copie locale a été mise à jour en dernier par le référentiel.

Avec ces informations, en parlant au référentiel, Subversion peut dire dans lequel des quatre états suivants se trouve un fichier de travail :

Inchangé et courant
: Le fichier est inchangé dans le répertoire de travail et aucun changement sur ce fichier n'a été livré au référentiel depuis sa révision de travail. Une **livraison** du fichier ne fera rien et une **mise à jour** du fichier ne fera rien.

Changé localement et courant
: Le fichier a été changé dans le répertoire de travail et aucun changement sur ce fichier n'a été livré au référentiel depuis sa révision de base. Il y a des changements locaux qui n'ont pas été livrés au référentiel, ainsi une **livraison** du fichier réussira à publier vos changements et une **mise à jour** du fichier ne fera rien.

Inchangé et périmé
: Le fichier n'a pas été changé dans le répertoire de travail, mais il a été changé dans le référentiel. Le fichier devrait être mis à jour éventuellement, pour l'actualiser avec la révision publique. Une **livraison** du fichier ne fera rien et une **mise à jour** du fichier intègrera les derniers changements dans votre copie de travail.

Changé localement et périmé
: Le fichier a été modifié dans le répertoire de travail, et dans le référentiel. Une **livraison** du fichier échouera avec une erreur *périmé*. Le fichier devrait d'abord être mis à jour ; la commande **mise à jour** essayera de fusionner les deux versions. Si Subversion réussit pas à faire correctement la fusion, il laisse l'utilisateur résoudre le conflit.

2.4. Résumé

Nous avons couvert un certain nombre de concepts fondamentaux de Subversion dans ce chapitre :

- Nous avons présenté les notions du référentiel central, la copie de travail du client et le tableau d'arbres de révision de référentiel.
- Nous avons vu quelques exemples simples comment deux collaborateurs peuvent utiliser Subversion pour publier et recevoir des changements l'un de l'autre, en utilisant le modèle "copier-modifier-fusionner".
- Nous avons un peu parlé de la façon dont Subversion suit à la trace et gère l'information dans une copie de travail.

Chapitre 3. Le référentiel

Peu importe le protocole que vous utilisez pour avoir accès à vos référentiels, vous devez toujours créer au moins un référentiel. Cela peut être fait soit avec le client de ligne de commande de Subversion soit avec TortoiseSVN.

Si vous n'avez pas encore créé de référentiel Subversion, il est temps de le faire maintenant.

3.1. Création de référentiel

Vous pouvez créer un référentiel avec le backend FSFS ou avec le format de base de données Berkeley (BDB) plus ancien. Le format de FSFS est plus rapide et plus facile à gérer, il fonctionne maintenant sur les partages réseau et sous Windows 98 sans problème. Le format BDB est considéré comme plus stable car il est utilisé depuis plus longtemps ; cependant, FSFS est maintenant utilisé depuis plusieurs années, donc cet argument est dépassé. Lisez *Choosing a Data Store* [http://svnbook.red-bean.com/en/1.5/svn.reposadmin.planning.html#svn.reposadmin.basics.backends] dans le manuel de Subversion pour plus d'informations.

3.1.1. Créer un référentiel avec le client de ligne de commande

1. Créez un dossier vide avec le nom SVN (par exemple `D:\SVN\`), qui est utilisé comme racine pour tous vos référentiels.
2. Créez un autre dossier `MonNouveauRéférentiel` dans `D:\SVN\`.
3. Ouvrez l'invite de commande (ou la boîte DOS), allez vers `D:\SVN\` et tapez

   ```
   svnadmin create --fs-type bdb MonNouveauRéférentiel
   ```

   ou

   ```
   svnadmin create --fs-type fsfs MonNouveauRéférentiel
   ```

Maintenant vous avez un nouveau référentiel situé à `D:\SVN\MonNouveauRéférentiel`.

3.1.2. Créer le référentiel avec TortoiseSVN

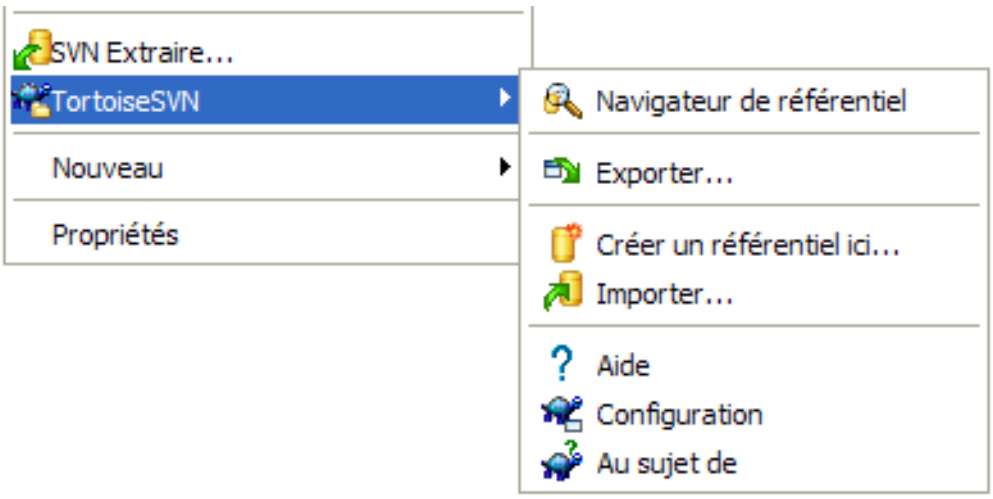


Figure 3.1. Le menu TortoiseSVN pour les dossiers non versionnés

1. Ouvrez l'explorateur Windows
2. Créez un nouveau dossier et nommez-le par exemple `SVNRéférentiel`
3. Faites un clic droit sur le dossier nouvellement créé et sélectionnez TortoiseSVN → Créer un référentiel ici....

   Un référentiel est alors créé à l'intérieur du nouveau dossier. *N'éditez pas ces fichiers vous-même !!!* Si vous avez des erreurs assurez vous que le dossier est vide et qu'il n'est pas protégé en écriture.

Astuce

TortoiseSVN ne permet plus de créer des référentiels BDB, cependant vous pouvez toujours le faire en ligne de commande. Les référentiels FSFS sont généralement plus aisés à maintenir et, pour des raisons de compatibilité entre les différentes versions de BDB, nous permet de maintenir plus aisément TortoiseSVN.

Pour des problèmes de compatibilité, les accès aux référentiels BDB de type `file://` ne seront plus supportés dans les versions futures de TortoiseSVN, cependant l'accès à ce type de référentiel via un serveur en utilisant les protocoles `svn://`, `http://` ou `https://` sera toujours supporté. Pour cette raison, pour les référentiels destinés à être utilisés via le protocole `file://` doivent être créés en FSFS.

Bien sûr, nous recommendons également de ne pas utiliser d'accès de type `file://`, hormis à des fins de test. Utiliser un serveur est plus sécurisé et plus fiable pour toute utilisation mutualisée.

3.1.3. Accès local au référentiel

Pour accéder à votre référentiel local vous avez besoin du chemin vers ce dossier. Souvenez-vous juste que Subversion s'attend à des chemins de référentiels dans la forme `file:///C:/SVNRéférentiel/`. Notez l'utilisation de slashs.

Pour avoir accès à un référentiel placé sur une partage réseau vous pouvez soit utiliser le mappage de disque, ou vous pouvez utiliser le chemin UNC. Pour les chemins UNC, la forme est `file://NomDuServeur/chemin/vers/référentiel/`. Notez qu'il y a seulement 2 slashs au début ici.

Avant SVN 1.2, les chemins UNC devaient être donnés dans la forme plus obscure `file:///\NomDuServeur/chemin/vers/référentiel` Cette forme est toujours supportée, mais n'est pas recommandée.

Avertissement

Ne créez pas ou n'accédez pas à un référentiel Berkeley DB sur un partage réseau. Il *ne peut pas* exister sur un système de fichiers distant. Même si vous mappez le disque réseau sur une lettre de disque. Si vous essayez d'utiliser Berkeley DB sur un partage réseau, les résultats seront imprévisibles - vous pouvez voir des erreurs mystérieuses tout de suite, ou il peut s'écouler des mois avant que vous ne découvriez que votre base de données du référentiel est corrompue.

3.1.4. Accéder à un référentiel situé dans un partage réseau

Au moins en théorie, il est possible d'utiliser un référentiel FSFS sur un partage réseau en utilisant le protocole `file://`, ce n'est certainement *pas* recommendé. En fait, nous le déconseillons *fortement*, et ne supportons pas un tel usage.

Premièrement vous donnez le droit d'accès en écriture dans le référentiel à tous les utilisateurs, de manière à ce qu'aucun ne puisse supprimer ou rendre complètement inutilisable de quelque manière que ce soit le référentiel.

Ensuite, tous les protocoles d'échange de fichiers ne supportent pas le verouillage dont Subversion a besoin, donc votre référentiel peut être corrompu. Ca ne devrait pas arriver tout de suite, mais un jour deux utilisateurs vont essayer d'accéder au référentiel en même temps.

Troisièmement, les droits des fichiers doivent être fixés exactement de cette manière. Vous pouvez vous en tirer sur un partage Windows, mais SAMBA est particulièrement compliqué.

Les accès de type `file://` sont réservés à des fins de test et de debug ou pour une utilisation mono-utilisateur. Lorsque vous voulez partager votre référentiel vous devez *vraiment* mettre en place un vrai serveur, et ce n'est pas aussi compliqué qu'il n'y parait. Lisez Section 3.5, « Accéder au référentiel » pour guider votre choix.

3.1.5. Disposition du référentiel

Avant que vous n'importiez vos données dans le référentiel vous devriez d'abord penser à la façon dont vous voulez organiser vos données. Si vous utilisez une des dispositions recommandées, ce sera beaucoup plus facile plus tard.

Il y a quelques standards, des manières recommandées d'organiser un référentiel. La plupart des personnes créent un répertoire `trunk` contenant la « version principale » de développement, un répertoire `branches` qui contient les copies de travail paralèlles et un répertoire `tags` qui contient les versions stables. Si un référentiel ne contient qu'un seul projet, ces répertoires sont créés à la base du référentiel :

```
/trunk
/branches
/tags
```

Si un référentiel contient de multiples projets, les gens indexent souvent leur disposition par branche :

```
/trunk/paint
/trunk/calc
/branches/paint
/branches/calc
/tags/paint
/tags/calc
```

...ou par projet:

```
/paint/trunk
/paint/branches
/paint/tags
/calc/trunk
/calc/branches
/calc/tags
```

L'indexation par projet a un sens si les projets ne sont pas étroitement liés et si chacun est extrait individuellement. Pour des projets liés où vous pouvez vouloir extraire tous les projets en une fois, ou où les projets sont tous liés ensemble dans un unique package de distribution, il est souvent mieux d'indexer par branche. De cette façon vous avez seulement un tronc à extraire et les relations entre les sous-projets sont plus facilement visibles.

Si vous adoptez une approche de niveau supérieur `/trunk /tags /branches`, il va sans dire que vous devez copier le tronc en entier pour chaque branche et chaque étiquette et d'une certaine façon, cette structure offre le plus de flexibilité.

Pour les projets sans rapport vous pouvez préférer utiliser des référentiels séparés. Quand vous livrez des changements, c'est le numéro de révision du référentiel entier qui change, pas le numéro de révision du projet. Avoir 2 projets sans rapport qui partagent un référentiel peut signifier de grands écarts dans les numéros de révision. Les projets Subversion et TortoiseSVN apparaissent à la même adresse hôte, mais sont des référentiels complètement séparés permettant le développement indépendant et aucune confusion sur les numéros de génération.

Bien sûr, vous êtes libre d'ignorer ces dispositions communes. Vous pouvez créer n'importe quel sorte de variation, ce qui marche le mieux pour vous ou votre équipe. Rappelez-vous que quoi que vous choisissez, ce n'est pas un engagement permanent. Vous pouvez réorganiser votre référentiel à tout moment. Parce que les branches et les étiquettes sont des répertoires ordinaires, TortoiseSVN peut les déplacer ou les renommer comme vous le souhaitez.

Passer d'une disposition à une autre ne revient qu'à exécuter une série de mouvements côté serveur ; si vous n'aimez pas la façon dont les choses sont organisées dans le référentiel, jonglez juste avec les répertoires.

Si vous n'avez pas encore créé la structure de base de votre référentiel, vous devriez le faire maintenant. Il y a deux manière de le faire. Si vous souhaitez simplement une structure `/trunk /tags /branches`, vous pouvez utiliser l'explorateur de référentiel pour créer les trois répertoires (dans trois livraisons différentes). Si vous souhaitez créer une arborescence plus complexe, il sera plus simple de le faire directement sur le disque puis de l'importer dans une seule livraison, comme suit:

1. créez un nouveau répertoire vide sur votre disque dur
2. créez votre structure de dossier de niveau supérieur désirée à l'intérieur de ce dossier - n'y mettez pas encore de fichiers !
3. importez cette structure dans le référentiel via un clic droit sur le dossier et choisissez TortoiseSVN → Importer... Cela importera votre dossier temporaire dans la racine du référentiel pour créer la disposition de base du référentiel.

Notez que le nom du dossier que vous importez n'apparaît pas dans le référentiel, seulement son contenu. Par exemple, créez la structure de dossier suivante :

```
C:\Temp\Nouveau\trunk
C:\Temp\Nouveau\branches
C:\Temp\Nouveau\tags
```

Importez `C:\Temp\Nouveau` dans la racine du référentiel, qui ressemblera alors à cela :

```
/trunk
/branches
/tags
```

3.2. Sauvegarde de référentiel

Quel que soit le type de référentiel que vous utilisez, il est extrêmement important que vous maintenez des sauvegardes régulières et que vous vérifiez la sauvegarde. Si le serveur tombe, vous pouvez être capable d'avoir accès à une version récente de vos fichiers, mais sans le référentiel tout votre historique est perdu pour toujours.

Le moyen le plus simple (mais non recommandé) est juste de copier le dossier du référentiel sur le médium de secours. Cependant, vous devez être absolument sûr qu'aucun processus n'ait accès aux données. Dans ce contexte, accès veut dire *pas* d'accès du tout. Un référentiel BDB est écrit même quand l'opération semble seulement exiger la lecture, comme l'obtention du statut. Si votre référentiel est accédé pendant la copie, (navigateur Internet laissé ouvert, WebSVN, etc) la sauvegarde sera sans valeur.

La méthode recommandée est d'exécuter

```
svnadmin hotcopy chemin/vers/référentiel chemin/vers/sauvegarde --clean-logs
```

pour créer une copie de votre référentiel d'une manière sûre. Sauvegardez alors la copie. L'option `--clean-logs` n'est pas exigée, mais supprime les fichiers de journal superflus quand vous sauvegardez un référentiel BDB, qui peut économiser un certain espace.

The `svnadmin` tool is installed automatically when you install the Subversion command line client. If you are installing the command line tools on a Windows PC, the best way is to download the Windows installer version. It is compressed more efficiently than the `.zip` version, so the download is smaller, and it takes care of setting the paths for you. You can download the latest version of the Subversion command line client from *http://subversion.tigris.org/getting.html*.

3.3. Scripts de hook côté serveur

Un script hook est un programme déclenché par un certain événement du référentiel, comme la création d'une nouvelle révision ou la modification d'une propriété non versionnée. Chaque hook fournit suffisamment d'informations pour dire à quel événement il se rapporte, sur quelle(s) cible(s) il doit fonctionner et le nom utilisateur de la personne qui a déclenché l'événement. Selon le résultat du hook ou le statut retourné, le programme hook peut continuer l'action, l'arrêter, ou la suspendre d'une certaine façon. Veuillez vous référez au chapitre sur *Scripts de Hook* [http://svnbook.red-bean.com/en/1.5/svn.reposadmin.create.html#svn.reposadmin.create.hooks] dans le manuel de Subversion pour plus de détails sur la façon dont les hooks sont mis en place.

Ces scripts de hook sont exécutés par le serveur qui héberge le référentiel. TortoiseSVN permet également de configurer des scripts de hook à exécuter côté client en réponse à certains événements. Voir Section 4.30.8, « Scripts hook côté client » pour plus d'information.

Des exemples de scripts hook peuvent être trouvés dans le répertoire `hooks` du référentiel. Ces exemples de scripts sont appropriées pour des serveurs Unix/Linux, mais doivent être modifiés si votre serveur est basé sur Windows. Le hook peut être un fichier batch ou un exécutable. L'exemple ci-dessous montre un fichier batch qui pourrait être utilisé pour mettre en oeuvre un hook de pre-revprop-change.

```
rem Autorise seulement les changements sur les commentaires.
if "%4" == "svn:log" exit 0
echo Property '%4' ne peut pas être modifié >&2
exit 1
```

Notez que tout ce qui est envoyé sur stdout est abandonné. Si vous voulez qu'un message apparaisse dans la boîte de dialogue du rejet de la livraison vous devez l'envoyer sur stderr. Dans un fichier batch, c'est réalisé en utilisant `>&2`

3.4. Liens d'extraction

Si vous voulez donner l'accès à votre référentiel Subversion à d'autres utilisateurs, vous pouvez vouloir mettre un lien vers celui-ci sur votre site Web. Une façon de rendre ceci plus accessible est d'inclure un *lien d'extraction* pour les autres utilisateurs de TortoiseSVN.

Quand vous installez TortoiseSVN, il enregistre un nouveau protocole `tsvn:`. Quand un utilisateur de TSVN clique sur un lien de ce type, la boîte de dialogue d'extraction s'ouvrira automatiquement avec l'URL du référentiel déjà remplie.

Pour inclure un tel lien dans votre page html, vous devez ajouter du code de ce type :

```
<a href="tsvn:http://nom.de.domaine.du.project.org/svn/trunk">
</a>
```

Bien sûr ce sera encore mieux si vous avez inclus une image appropriée. Vous pouvez utiliser le *logo de Tortoise SVN* [http://tortoisesvn.tigris.org/images/TortoiseCheckout.png] ou vous pouvez vous servir de votre propre image.

```
<a href="tsvn:http://nom.de.domaine.du.projet.org/svn/trunk">
<img src=TortoiseCheckout.png></a>
```

Vous pouvez aussi faire pointer le raccourci vers une révision spécifique, par exemple

```
<a href="tsvn:http://nom.de.domaine.du.projet.org/svn/trunk?100">
</a>
```

3.5. Accéder au référentiel

Pour utiliser TortoiseSVN (ou un autre client Subversion), vous avez besoin d'un hébergement pour vos référentiels. Vous pouvez soit les stocker localement et y accéder en utilisant le protocole `file://`, soit les placer sur un serveur et y accéder avec les protocoles `http://` ou `svn://`. Ces deux protocoles peuvent aussi être cryptés. Vous utiliserez alors `https://`, `svn+ssh://`, ou encore `svn://` avec SASL.

Si votre serveur est chez un hégergeur publique comme *Google Code* [http://code.google.com/hosting/] ou qu'il a déjà été configuré entièrement par quelqu'un d'autre. Allez à la page Chapitre 4, *Guide d'utilisation quotidienne*.

Si vous n'avez pas de serveur, si vous travaillez seul et/ou si vous souhaitez juste tester Subversion et TortoiseSVN, alors les référentiels locaux sont probablement la meilleure solution. Créez juste un référentiel sur votre ordinateur comme décrit dans Chapitre 3, *Le référentiel*. Vous pouvez sauter le reste de ce chapitre et aller directement au Chapitre 4, *Guide d'utilisation quotidienne* pour savoir comment commencer à l'utiliser.

Mettre en place un référentiel destiné à être utilisé par plusieurs utilisateurs sur un partage réseau n'est pas conseillé. Lisez Section 3.1.4, « Accéder à un référentiel situé dans un partage réseau » pour savoir pourquoi nous pensons que ce n'est pas une bonne idée. Mettre en place un serveur n'est pas aussi compliqué qu'il n'y parait, sera beaucoup plus sûr et probablement plus rapide.

Les sections suivantes sont des guides pas à pas d'installation de ce type de serveur sur une machine Windows.Bien sûr vous pouvez aussi l'installer sur une machine Linux, mais c'est en dehors du contexte de ce guide. Des informations plus détaillées sur les options du serveur Subversion et sur la manière de faire le bon choix d'architecture sont disponibles dans le livre Subversion, dans la section *Configuration du Serveur* [http://svnbook.red-bean.com/en/1.5/svn.serverconfig.html].

3.6. Serveur basé sur Svnserve

3.6.1. Introduction

Subversion est livré avec Svnserve - un serveur stand-alone léger qui utilise une variante du protocole TCP/IP pour se connecter. C'est la solution idéale pour de petites installations, ou si un bon gros Apache ne peut être utilisé.

Dans la plupart des cas svnserve est plus facile à installer et fonctionne plus rapidement que le serveur utilisant Apache, cela dit il y manque quelques fonctionnalités. D'autant plus maintenant qu'est inclu le support SASL.

3.6.2. Installer svnserve

1. Get the latest version of Subversion from *http://subversion.tigris.org/getting.html*. Alternatively get a pre-packaged installer from CollabNet at *http://www.collab.net/downloads/subversion*. This installer will setup svnserve as a Windows service, and also includes some of the tools you need if you are going to use SASL for security.

2. Si vous déjà une version de Subversion d'installée, et que svnserve est lancé, vous devrez l'arrêter avant de continuer.

3. Exécutez le programme d'installation de Subversion. Si vous l'exécutez sur votre serveur (ce qui est recommendé) vous pouvez passer l'étape 4.

4. Ouvrez l'explorateur Windows, allez au répertoire d'installation de Subversion (habituellement `C:\Program Files\Subversion`) et dans le répertoire `bin`, trouvez les fichiers `svnserve.exe`, `intl3_svn.dll`, `libapr.dll`, `libapriconv.dll`, `libapriutil.dll`, `libdb*.dll`, `libeay32.dll` et `ssleay32.dll` - copiez ces fichiers, ou copiez simplement tout le contenu du répertoire `bin`, dans un répertoire sur votre serveur, par exemple `c:\svnserve`

3.6.3. Exécuter svnserve

Maintenant que svnserve est installé, vous avez besoin qu'il tourne sur votre serveur. L'approche la plus simple est d'exécuter ce qui suit à partir d'un interpréteur de commandes DOS ou de créer un raccourci Windows :

```
svnserve.exe --daemon
```

svnserve démarrera maintenant en attendant des requêtes entrantes sur le port 3690. Le commutateur --daemon indique à svnserve de fonctionner comme un processus de démon, donc il existera toujours jusqu'à ce qu'il soit arrêté manuellement.

Si vous n'avez pas encore créé de référentiel, suivez les instructions données avec le paramétrage du serveur Apache Section 3.7.4, « Configuration ».

Pour tester que svnserve fonctionne, utilisez TortoiseSVN → Explorateur de référentiel pour voir un référentiel.

En supposant que votre référentiel est placé dans `c:\repos\TestRepo` et votre serveur est appelé `localhost`, entrer :

```
svn://localhost/repos/TestRepo
```

quand l'explorateur de référentiel le demande.

Vous pouvez aussi augmenter la sécurité et gagner du temps en entrant l'URL avec svnserve en utilisant le commutateur --root pour indiquer l'emplacement racine et limiter l'accès à un répertoire spécifique sur le serveur :

```
svnserve.exe --daemon --root disque:\chemin\vers\racine\référentiel
```

En utilisant le test précédent comme guide, svnserve fonctionnerait maintenant comme :

```
svnserve.exe --daemon --root c:\référentiels
```

Et dans TortoiseSVN notre URL de l'explorateur de référentiel est maintenant raccourcie en :

```
svn://localhost/TestRéférentiel
```

Notez que le commutateur --root est aussi nécessaire si votre référentiel est placé sur une partition ou un disque différent de l'emplacement de svnserve sur votre serveur.

Svnserve s'occupera de plusieurs référentiels. Indiquez juste leur emplacement dans le répertoire racine, et accédez y en utilisant des chemins relatifs à la racine.

Avertissement

Ne créez pas ou n'accédez pas à un référentiel Berkeley DB sur un partage réseau. Il *ne peut pas* exister sur un système de fichiers distant. Même si vous faites mapper le disque réseau à une lettre de disque. Si vous essayez d'utiliser Berkeley DB sur un partage réseau, les résultats sont imprévisibles - vous pouvez voir des erreurs mystérieuses tout de suite, ou cela peut s'écouler des mois avant que vous ne découvriez que votre base de données du référentiel est subtilement corrompue.

3.6.3.1. Exécuter svnserve en tant que service

Exécuter svnserve par un utilisateur n'est généralement pas la meilleure solution. Cela revient à avoir toujours un utilisateur connecté sur votre serveur et vous rappeler de le relancer après un redémarrage. Une meilleure manière consiste à lancer svnserve en tant que service Windows. À partir de Subversion 1.4, svnserve peut être installé comme un service windows natif.

Pour installer svnserve comme service Windows, exécutez la commande suivante. Le service sera alors créé et lancé à chaque démarrage de Windows.

```
sc create svnserve binpath= "c:\svnserve\svnserve.exe --service
    --root c:\referentiel" displayname= "Subversion" depend= tcpip
    start= auto
```

Si un des chemin d'accès contient des espaces, vous devrez encapsuler ceux ci avec des double cotes, comme suit :

```
sc create svnserve binpath= "
    \"C:\Program Files\Subversion\bin\svnserve.exe\"
    --service --root c:\referentiel" displayname= "Subversion"
    depend= tcpip start= auto
```

Vous pouvez également ajouter une description après avoir créé le service. Elle sera visible depuis la fenêtre de gestion des services de Windows.

```
sc description svnserve "Serveur Subversion (svnserve)"
```

Notez la ligne de commande relativement atypique utilisée par `sc`. Dans la paire `clé= valeur` il ne doit pas y avoir d'espace entre la clé et le = mais il doit y en avoir un avant la valeur.

Astuce

Microsoft recommande maintenant que les services soient exécutés soit par le compte Service Local soit par le compte Service Réseau. Référez-vous *The Services and Service Accounts Security Planning Guide* [http://www.microsoft.com/technet/security/topics/serversecurity/serviceaccount/default.mspx]. Pour créer le service sous le compte Service Local, ajoutez ce qui suit à l'exemple ci-dessus.

```
obj= "NT AUTHORITY\LocalService"
```

Notez que vous devriez donner au compte Service Local les droits appropriés tant pour Subversion et vos référentiels que pour toute application utilisée par les scripts hook. Le groupe intégré pour cela est appelé "SERVICE LOCAL".

A partir du moment où vous avez installé le service, vous devez aller dans la fenêtre de gestion des services pour le démarrer (juste cette fois ci, il se lancera ensuite automatiquement après le redémarrage du serveur).

Pour plus d'informations, consultez le document*Windows Service Support for Svnserve* [http://svn.collab.net/repos/svn/trunk/notes/windows-service.txt].

Si vous aviez une ancienne version de svnserve utilisant le wrapper `SVNService`, et que vous voulez maintenant utiliser le support natif, vous devez désincrire le wrapper comme service (n'oubliez pas d'arrêter d'abord le service !). Utilisez simplement la commande

```
svnservice -remove
```

pour retirer l'entrée dans la base de registre.

3.6.4. Authentification de base avec svnserve

L'installation par défaut de svnserve fournit un accès anonyme en lecture seule. Cela signifie que vous pouvez utiliser une URL du type `svn://` pour extraire, mettre à jour, ou utiliser l'explorateur de référentiel dans TortoiseSVN pour voir le référentiel, mais vous ne pourrez faire aucune livraison.

Pour permettre l'accès en écriture à un référentiel, vous devez éditer le fichier `conf/svnserve.conf` dans votre répertoire de référentiel. Ce fichier contrôle la configuration du démon svnserve et contient aussi la documentation utile.

Vous pouvez activer l'accès en écriture anonyme en mettant simplement :

```
[general]
anon-access = write
```

Cependant, vous ne saurez pas qui a fait des changements à un référentiel, puisque la propriété `svn:author` sera vide. Vous serez aussi incapables de contrôler qui fait des changements à un référentiel. C'est un paramétrage quelque peu risqué !

Une façon de surmonter cela est de créer une base de données de mot de passe :

```
[general]
anon-access = none
auth-access = write
password-db = fichier_utilisateur
```

Où `fichier_utilisateur` est un fichier qui existe dans le même répertoire que `svnserve.conf`. Ce fichier peut vivre ailleurs dans votre système de fichiers (utile pour quand vous avez plusieurs référentiels qui exigent les mêmes droits d'accès) et peuvent être référencés en utilisant un chemin absolu, ou un chemin relatif au répertoire `conf`. Si vous incluez un chemin, il doit être écrit de `/la/manière/unix`. L'utilisation de \ ou des lettres de disque ne marchera pas. Le `fichier_utilisateur` devrait avoir une structure de :

```
[users]
nom_utilisateur = mot_de_passe
...
```

Cet exemple interdira tout accès aux utilisateurs non authentifiés (anonymes) et donnera l'accès en lecture-écriture aux utilisateurs inscrits dans `fichier_utilisateur`.

Astuce

Si vous gérez plusieurs référentiels avec la même base de données de mots de passe, l'utilisation d'un groupe d'identification facilitera la tâche des utilisateurs, dans la mesure où TortoiseSVN peut garder vos accréditations dans la mémoire cache vous n'avez à les saisir qu'une seule fois. Plus d'informations sont disponibles dans le manuel de Subversion dans les sections *Create a 'users' file and realm* [http://svnbook.red-bean.com/en/1.5/svn.serverconfig.svnserve.html#svn.serverconfig.svnserve.auth.users] and *Client Credentials Caching* [http://svnbook.red-bean.com/en/1.5/svn.serverconfig.netmodel.html#svn.serverconfig.netmodel.credcache]

3.6.5. Sécuriser le serveur avec SASL

3.6.5.1. Qu'est-ce que SASL ?

La couche d'authentification et de sécurisation Cyrus est open source et développée par l'université Carnegie Mellon. Elle étoffe les protocoles réseau de capacités d'authentification et de cryptage, et tout comme Subversion à partir de sa version 1.5, svnserve et TortoiseSVN savent utiliser de cette librairie.

Pour plus de détails sur les options disponibles, consultez la section *Using svnserve with SASL* [http://svnbook.red-bean.com/en/1.5/svn.serverconfig.svnserve.html#svn.serverconfig.svnserve.sasl] du manuel de référence de Subversion. Si vous recherchez un moyen simple de mettre en place un système d'authentification et de cryptage sur un serveur Windows, de manière à pouvoir accéder à votre référentiel depuis le gros méchant Internet, continuez à lire.

3.6.5.2. Authentification SASL

Pour activer le mécanisme spéficique à SASL sur le serverr, vous aurez besoin de faire trois choses. Premièrement, créer une section `[sasl]` dans le fichier `svnserve.conf` de votre référentiel, avec ce couple clé-valeur:

```
use-sasl = true
```

Ensuite, créez un fichier nommé `svn.conf` au bon endroit - typiquement dans le réportoire d'installation de Subversion.

Troisièmement, créer deux clefs de registre pour indiquer à SASL où trouver les éléments. Créez une clé nommée `[HKEY_LOCAL_MACHINE\SOFTWARE\Carnegie Mellon\Project Cyrus \SASL Library]` et insérez y deux valeurs de type chaine de caractère: `SearchPath` ayant comme valeur le chemin contenant le greffon `sasl*.dll` (qui est normalement le répertoire d'installation de Subversion), et `ConfFile` ayant comme valeur le chemin contenant le fichier `svn.conf`. Si vous avez utilisé le programme d'installation de CollabNet, ces clés auront déjà été créées pour vous.

Ajoutez les lignes suivantes au fichier `svn.conf` :

```
pwcheck_method: auxprop
auxprop_plugin: sasldb
mech_list: DIGEST-MD5
sasldb_path: C:\TortoiseSVN\sasldb
```

La dernière ligne contient le chemin vers la base de données d'authentification, qui est un fichier nommé `sasldb`. Elle peut aller n'importe où, mais un choix adapté est le répertoire parent du référentiel. Assurez vous que le service svnserve à le droit de lire ce fichier.

Si svnserve est démarré, redémarrez le pour qu'il puisse prendre en compte les nouvelles valeurs de configuration.

A présent que tout est mis en place, il ne vous reste plus qu'à créer des utilisateurs ainsi que leur mot de passe. Pour ce faire, utilisez l'utilitaire `saslpasswd2`. Si vous avez utilisé le programme d'installation de CollabNet, cet utilitaire sera dans le répertoire d'installation. Utilisez une commande du style :

```
saslpasswd2 -c -f C:\TortoiseSVN\sasldb -u realm utilisateur
```

L'option `-f` donne le chemin vers la base de données, `realm` doit avoir la même valeur que celle définie dans le fichier `svnserve.conf` de votre référentiel, et utilisateur est exactement ce à quoi vous vous attendez. La valeur realm ne doit pas contenir d'espace.

Vous pouvez voir la liste des utilisateurs enregistrés dans la base de données en utilisant le programme `sasldblistusers2`.

3.6.5.3. Cryptage SASL

Pour activer ou désactiver des niveaux de cryptage, vous pouvez renseigner les deux valeurs suivantes dans le fichier `svnserve.conf` de votre référentiel :

```
[sasl]
use-sasl = true
min-encryption = 128
max-encryption = 256
```

Les options `min-encryption` et `max-encryption` controlent le niveau de cryptage requis par le serveur. Pour désactiver complètement ce cryptage, mettez ces deux valeurs à 0. Pour activer un checksum simple (i.e., protéger et garantir l'intégrité des données non cryptées), mettez les deux options à 1. Si vous voulez que le cryptage soit possible (mais pas nécessaire), mettez le minimum à 0, et le maximum à une valeur de quelques bits. Pour obliger sans condition le cryptage, les deux options à des valeurs supérieures à 1. Dans notre exemple précédent, on oblige les clients à faire un cryptage situé entre 128 bits et 256 bits.

3.6.6. Authentification avec svn+ssh

Une autre façon d'authentifier les utilisateurs avec un serveur basé sur svnserve est d'utiliser un shell sécurisé (SSH) pour tunneler les requêtes. Cette méthode n'est pas aussi facile à configurer que SASL, mais peut être utile dans certains cas.

Avec cette approche, svnserve n'est pas exécuté comme un processus démon, le shell sécurisé démarre svnserve pour vous plutôt, l'exécutant comme l'utilisateur SSH authentifié. Pour activer cela, vous avez besoin d'un démon shell sécurisé sur votre serveur.

Une méthode simple pour mettre en place votre serveur est donnée là : Annexe G, *Sécuriser Svnserve grâce à SSH*. Vous pouvez trouver d'autres informations concernant SSH dans les FAQ en tapant « SSH » comme champ de recherche.

Plus d'informations sur svnserve dans *Version Control with Subversion* [http://svnbook.red-bean.com].

3.6.7. Autorisation basée sur le chemin avec svnserve

En démarrant avec Subversion 1.3, svnserve supporte le même arrangement d'autorisation à base de chemin `mod_authz_svn` qui est disponible avec le serveur Apache. Vous devez éditer le fichier `conf/svnserve.conf` dans votre répertoire de référentiel et ajouter une ligne se référant à votre fichier d'autorisation.

```
[general]
authz-db = authz
```

Ici, `authz` est un fichier que vous créez pour définir les autorisations d'accès. Vous pouvez utiliser un fichier séparé pour chaque référentiel, ou vous pouvez utiliser le même fichier pour plusieurs référentiels. Lisez Section 3.7.6, « Autorisation basée sur le chemin » pour une description du format du fichier.

3.7. Serveur basé sur Apache

3.7.1. Introduction

La plus flexible de toutes les installations serveur possibles pour Subversion est basée sur Apache. Bien qu'un peu plus compliquée à mettre en place, elle offre des avantages que d'autres serveurs n'ont pas :

WebDAV
: .Le serveur Subversion basé sur Apache utilise le protocole WebDAV qui est aussi supporté par beaucoup d'autres programmes. Vous pourriez par exemple monter un tel référentiel comme un « Répertoire Web » dans l'explorateur Windows et y accéder ensuite comme n'importe quel autre dossier.

Parcourir le référentiel
: Vous pouvez diriger votre navigateur à l'URL de votre référentiel et naviguer dans son contenu sans avoir de client Subversion installé. Cela donne accès à vos données à un cercle beaucoup plus large d'utilisateurs.

Authentification
: Vous pouvez utiliser n'importe quel mécanisme d'identification que supporte Apache, y compris SSPI et LDAP.

Sécurité
: Puisqu'Apache est très stable et sécurisé, vous obtenez automatiquement la même sécurité pour votre référentiel. Cela inclut le cryptage SSL.

3.7.2. Installer Apache

La première chose dont vous avez besoin avant d'installer Apache est d'un ordinateur avec Windows2000, WinXP+SP1, Windows 2003, Vista ou Windows Server 2008.

Avertissement

Veuiller noter que Windows XP sans le service pack 1 entrainera des problèmes de données réseau et pourrait donc corrompre votre référentiel !

1. Téléchargez la dernière version du serveur web Apache à partir de *http://httpd.apache.org/download.cgi*. Assurez-vous que vous téléchargez la version 2.2.x - la version 1.3.xx ne marchera pas !

   Le programme d'installation d'Apache est disponible en cliquant sur le lien `autres fichiers`, allez ensuite dans le répertoire `binaries/win32`. Vous pouvez utiliser le fichier msi `apache-2.2.x-win32-x86-openssl-0.9.x.msi` (celui incluant le support OpenSSL).

2. Une fois que vous avez l'installeur Apache2, vous pouvez double-cliquer dessus et il vous guidera dans le processus d'installation. Assurez-vous que vous entrez l'URL du serveur correctement (si vous n'avez pas de nom de DNS pour votre serveur entrez seulement l'adresse IP). Je recommande d'installer Apache *pour Tous les Utilisateurs, sur le Port 80, en tant que Service* . Note : si vous avez déjà IIS ou un autre programme lancé qui écoute sur le port 80 l'installation pourrait échouer. Si cela arrive, allez au répertoire de programmes, `\Apache Group\Apache2\conf` et repérer le fichier `httpd.conf`.

Éditez ce fichier pour que `Listen 80` soit changé pour un port libre, par exemple `Listen 81`. Redémarrez alors l'installation - elle devrait cette fois se terminer sans problème.

3. Maintenant testez si le serveur web Apache fonctionne correctement en allant avec votre navigateur internet à `http://localhost/` - un site web préconfiguré devrait s'afficher.

Attention

Si vous décidez d'installer Apache en tant que service, soyez averti que par défaut il fonctionnera en tant que compte du système local. Une pratique plus sécurisée pour vous serait de créer un compte séparé pour exécuter Apache.

Assurez-vous que le compte sur le serveur sur lequel Apache fonctionne possède une entrée explicite dans la liste de contrôle d'accès du répertoire du référentiel (clic-droit sur le répertoire | propriétés | sécurité), avec contrôle total. Autrement, les utilisateurs ne seront pas capables de livrer leurs changements.

Même si Apache fonctionne en tant que système local, vous avez toujours besoin d'une telle entrée (qui sera le compte SYSTEM dans ce cas).

Si cette permission n'est pas activé dans la configuration de vote serveur Apache, vos utilisateurs auront des messages d'erreur « Accès refusé », qui s'afficheront dans le journal des erreurs Apache comme erreur 500.

3.7.3. Installer Subversion

1. Téléchargez la dernière version des fichiers binaires Win32 de Subversion pour Apache. Soyez sûr de prendre la version correspondant à la version d'Apache que vous avez installé, dans le cas contraire vous aurez un message imcompréhensible quand vous redémarrerez. Si votre version d'Apache est la 2.2.x allez à cette adresse : *http://subversion.tigris.org/servlets/ProjectDocumentList?folderID=8100*.

2. Exécutez l'installeur de Subversion et suivez les instructions. Si l'installeur de Subversion a reconnu que vous avez installé Apache, donc vous êtes presque au bout. S'il n'a pas pu trouver un serveur Apache alors vous devez faire quelques étapes complémentaires.

3.

   En utilisant l'explorateur Windows, allez au répertoire d'installation de Subversion (d'habitude `c:\program files\Subversion`) et trouvez les fichiers `/httpd/mod_dav_svn.so` et `mod_authz_svn.so`. Copiez ces fichiers au répertoire de modules Apache (d'habitude `c:\program files\apache group\apache2\modules`).

4. Copiez les fichiers `/bin/libdb*.dll` et `/bin/intl3_svn.dll` du répertoire d'installation de Subversion au répertoire bin d'Apache.

5. Éditez le fichier de configuration d'Apache (d'habitude `C:\Program Files\Apache Group\Apache2\conf\httpd.conf`) avec un éditeur de texte comme le Bloc-notes et faites les changements suivants :

   Décommentez (supprimez la marque '#') les lignes suivantes :

   ```
   #LoadModule dav_fs_module modules/mod_dav_fs.so
   #LoadModule dav_module modules/mod_dav.so
   ```

   Ajoutez les deux lignes suivantes à la fin de la section `LoadModule`.

   ```
   LoadModule dav_svn_module modules/mod_dav_svn.so
   ```

```
LoadModule authz_svn_module modules/mod_authz_svn.so
```

3.7.4. Configuration

Vous avez maintenant mis en place Apache et Subversion, mais Apache ne sait pas encore comment gérer les clients Subversion comme TortoiseSVN. Pour qu'Apache sache quelle URL sera utilisée pour les référentiels Subversion vous devez éditer le fichier de configuration d'Apache (d'habitude placé dans `c:\program files\apache group\apache2\conf\httpd.conf`) avec votre éditeur de texte préféré (par exemple le Bloc-notes) :

1. À la fin du fichier de configuration, ajoutez les lignes suivantes :

   ```
   <Location /svn>
     DAV svn
     SVNListParentPath on
     SVNParentPath D:\SVN
     #SVNIndexXSLT "/svnindex.xsl"
     AuthType Basic
     AuthName "Subversion repositories"
     AuthUserFile passwd
     #AuthzSVNAccessFile svnaccessfile
     Require valid-user
   </Location>
   ```

   Elle correspondent à une configuration d'Apache pour prendre en compte des référentiels Subversion situés physiquement dans le répertoire `D:\SVN`. Les référentiels sont disponibles pour le monde extérieur à partir de l'URL : `http://MonServeur/svn/` . L'accès est limité aux utilisateurs/mots de passe listés dans le fichier `passwd`.

2. Pour créer le fichier `passwd`, ouvrez l'invite de commande (la boîte DOS) une nouvelle fois, allez dans le dossier `apache2` (habituellement `c:\program files\apache group\apache2`) et créez le fichier en tapant

   ```
   bin\htpasswd -c passwd <nom de l'utilisateur>
   ```

   Cela créera un fichier avec le nom `passwd` qui est utilisé pour l'authentification. Des utilisateurs supplémentaires peuvent être ajoutés avec

   ```
   bin\htpasswd passwd <nom de l'utilisateur>
   ```

3. Redémarrer le service Apache une nouvelle fois.

4. Allez avec votre navigateur sur `http://MonServeur/svn/MonNouveauRéférentiel` (où `MonNouveauRéférentiel` est le nom du référentiel Subversion que vous avez créé auparavant). Si tout s'est bien passé, un nom d'utilisateur et un mot de passe vous seront demandés, puis vous pouvez voir le contenu de votre référentiel.

Une courte explication sur ce que vous venez juste de saisir :

Réglage	Explication
<Location /svn>	veut dire que les référentiels Subversion sont disponibles à partir de l'URL `http://MonServeur/svn/`
DAV svn	indique à Apache quel module sera responsable pour servir cette URL - dans ce cas le module de Subversion.

Réglage	Explication
SVNListParentPath on	Pour la version 1.3 de Subversion et supérieure, cette directive permet de lister tous les référentiels disponibles sous `SVNParentPath`.
SVNParentPath D:\SVN	indique à Subversion de chercher des référentiel sous `D:\SVN`
SVNIndexXSLT "/svnindex.xsl"	Utilisé pour parcourir Internet avec un plus joli navigateur.
AuthType Basic	active l'authentification de base, c'est-à-dire Nom de l'utilisateur/mot de passe
AuthName "Subversion repositories"	est utilisé comme information chaque fois qu'un dialogue d'identification apparaît pour dire à l'utilisateur pour quoi l'authentification est demandée.
AuthUserFile passwd	spécifie quel fichier de mots de passe est utilisé pour l'authentification
AuthzSVNAccessFile	Emplacement du fichier d'accès pour les chemins à l'intérieur d'un référentiel Subversion
Require valid-user	spécifie que seuls ls utilisateurs qui ont entré un nom d'utilisateur/mot de passe correct peuvent accéder à l'URL

Tableau 3.1. Réglages du `httpd.conf` d'Apache

Mais ce n'est qu'un exemple. Il y a beaucoup, beaucoup plus de possibilités que vous pouvez faire avec le serveur web Apache.

- Si vous voulez que votre référentiel ait l'accès en lecture pour tout le monde mais l'accès en écriture seulement pour des utilisateurs spécifiques vous pouvez changer la ligne

  ```
  Require valid-user
  ```

  en

  ```
  <LimitExcept GET PROPFIND OPTIONS REPORT>
  Require valid-user
  </LimitExcept>
  ```

- Utiliser un fichier `passwd` limite et autorise l'accès à tous vos référentiels comme une unité. Si vous voulez plus de contrôle sur les utilisateurs par dossier à l'intérieur d'un référentiel vous pouvez décommenter la ligne

  ```
  #AuthzSVNAccessFile svnaccessfile
  ```

  et créer un fichier d'accès de Subversion. Apache s'assurera que seuls les utilisateurs valides sont capables d'avoir accès votre emplacement `/svn` et passera alors le nom d'utilisateur au module `AuthzSVNAccessFile` de Subversion pour qu'il puisse mettre en application un accès plus granulaire basé sur les règles inscrites dans le fichier d'accès de Subversion. Notez que les chemins sont spécifiés comme `référentiel:chemin` ou simplement `chemin`. Si vous ne spécifiez pas de référentiel particulier, cette règle d'accès s'appliquera à tous les référentiels sous `SVNParentPath`. Le format du fichier de stratégie d'autorisation utilisé par `mod_authz_svn` est décrit dans Section 3.7.6, « Autorisation basée sur le chemin »

- Pour parcourir le référentiel avec un navigateur Internet plus agréable, décommentez la ligne

```
#SVNIndexXSLT "/svnindex.xsl"
```

et mettez les fichiers `svnindex.xsl`, `svnindex.css` et `menucheckout.ico` dans le dossier racine (la plupart du temps `C:/Program Files/Apache Group/Apache2/htdocs`). Le nom de ce répertoire est fixé par la directive `DocumentRoot` dans votre fichier de configuration Apache.

Vous pouvez récupérer ces trois fichiers depuis notre référentiel : *http://tortoisesvn.tigris.org/svn/tortoisesvn/trunk/contrib/other/svnindex*. (Section 3, « TortoiseSVN est gratuit ! » explique comment accéder au référentiel des sources de TortoiseSVN).

Le fichier XSL du référentiel TortoiseSVN a une fonctionnalité pratique : Si vous parcourez le référentiel grâce à un navigateur Internet, chaque répertoire aura une icone sur sa droite. Si vous cliquez sur cette icone, la fenêtre de livraison de TortoiseSVN sera ouverte depuis cette URL.

3.7.5. Plusieurs référentiels

Si vous avez utilisé la directive `SVNParentPath` alors vous n'avez pas à changer le fichier de configuration d'Apache chaque fois que vous ajoutez un nouveau référentiel Subversion. Créez simplement le nouveau référentiel au même emplacement que le premier référentiel et vous avez fini ! Dans ma société, j'ai un accès direct à ce dossier spécifique sur le serveur via SMB (accès Windows normal aux fichiers). Donc je crée juste un nouveau dossier là, j'exécute la commande de TortoiseSVN TortoiseSVN → Créer un référentiel ici... et un nouveau projet a un foyer...

Si vous utilisez Subversion 1.3 ou supérieur, vous pouvez utiliser la directive `SVNListParentPath on` pour permettre à Apache de produire une liste de tous les projets disponibles si vous allez avec votre navigateur sur le chemin parent plutôt qu'à un référentiel spécifique.

3.7.6. Autorisation basée sur le chemin

Le module `mod_authz_svn` permet un contrôle précis des autorisations d'accès basé sur les noms des utilisateurs et les chemins de référentiel. C'est possible avec le serveur Apache et, à partir de Subversion 1.3, c'est aussi disponible avec svnserve.

Un fichier d'exemple ressemblerait à ceci :

```
[groups]
admin = john, kate
devteam1 = john, rachel, sally
devteam2 = kate, peter, mark
docs = bob, jane, mike
training = zak
# Accès par défaut à TOUS les référentiels
# Tout le monde peut lire, les admins peuvent écrire, Dan German est exclus.
[/]
* = r
@admin = rw
dangerman =
# Permet aux développeurs un accès total aux référentiels de leurs projets
[proj1:/]
@devteam1 = rw
[proj2:/]
@devteam2 = rw
[bigproj:/]
@devteam1 = rw
@devteam2 = rw
trevor = rw
# Donne aux personnes en charge de la documentation un accès en écrite sur tous l
```

```
[/trunk/doc]
@docs = rw
# Donne aux débutants un accès en écriture uniquement au référentiel d'entraîneme
[TrainingRepos:/]
@training = rw
```

Notez que la vérification de chaque chemin peut être une opération couteuse, particulièrement dans le cas du journal de révision. Le serveur vérifie chaque chemin changé dans chaque révision et le vérifie pour la lisibilité, qui peut être consommatrice de temps sur les révisions qui affectent de grands nombres de fichiers.

L'authentification et l'autorisation sont des processus séparés. Si un utilisateur veut obtenir l'accès à un chemin de référentiel, il doit satisfaire *tant* les pré-requis habituels d'authentification que les pré-requis d'autorisation du fichier d'accès.

3.7.7. Authentification avec un domaine Windows

Comme vous l'avez peut-être remarqué vous devez faire une entrée nom d'utilisateur/mot de passe dans le fichier `passwd` pour chaque utilisateur séparément. Et si (pour des raisons de sécurité) vous voulez que vos utilisateurs changent périodiquement leurs mots de passe vous devez faire le changement manuellement.

Mais il y a une solution pour ce problème - du moins si vous avez accès au référentiel de l'intérieur d'un réseau local avec un contrôleur de domaine Windows : `mod_auth_sspi` !

Le module SSPI original a été offert par Syneapps en incluant le code source. Mais son développement a été arrêté. Mais ne désespérez pas, la communauté l'a récupéré et l'a amélioré. Il a un nouveau foyer sur *SourceForge* [http://sourceforge.net/projects/mod-auth-sspi/].

- Téléchargez le module qui correspond à votre version d'Apache, copiez alors le fichier `mod_auth_sspi.so` dans le dossier contenant les modules Apache.
- Éditez le fichier de configuration d'Apache : ajoutez la ligne

  ```
  LoadModule sspi_auth_module modules/mod_auth_sspi.so
  ```

  à la section `LoadModule`. Assurez-vous d'insérer cette ligne *avant* la ligne

  ```
  LoadModule auth_module modules/mod_auth.so
  ```

- Pour faire que les emplacements de Subversion utilisent ce type d'authentification, vous devez changer la ligne

  ```
  AuthType Basic
  ```

  en

  ```
  AuthType SSPI
  ```

  vous devez ajouter aussi

  ```
  SSPIAuth On
  SSPIAuthoritative On
  SSPIDomain <domaincontroller>
  SSPIOmitDomain on
  ```

```
SSPIUsernameCase lower
SSPIPerRequestAuth on
SSPIOfferBasic On
```

dans le bloc `<Location /svn>`. Si vous n'avez pas de contrôleur de domaine, laissez le nom du contrôle de domaine comme `<domaincontroller>`.

Notez que si vous vous authentifiez en utilisant SSPI, alors vous n'avez plus besoin de la ligne `AuthUserFile` pour définir un fichier de mot de passe. Apache authentifie votre nom d'utilisateur et votre mot de passe avec votre domaine Windows à la place. Vous devrez mettre à jour la liste d'utilisateurs dans votre `svnaccessfile` pour faire aussi référence au `DOMAINE\nom d'utilisateur`.

Important

L'authentification SSPI n'est disponible que pour les connexions sécurisées de type SSL (https). Si vous vous connectez à votre serveur en utilisant une connexion http simple, cela ne fonctionnera pas.

Pour activer SSL sur votre serveur, voir le chapitre : Section 3.7.9, « Sécuriser le serveur avec SSL »

Astuce

Les fichiers `AuthzSVNAccessFile` de Subversion sont sensibles à la casse en ce qui concerne les noms d'utilisateur ("JUtilisateur" diffère "jutilisateur").

Dans le monde de Microsoft, les domaines Windows et les mots de passe ne sont pas sensibles à la casse. Cependant, certains administrateurs réseau aiment créer des comptes utilisateur en CamelCase (par exemple "JUtilisateur").

Cette différence peut vous mordre quand vous utilisez l'authentification SSPI lorsque le domaine Windows et les noms d'utilisateur sont passés à Subversion comme lorsque l'utilisateur les tape à l'invite. Internet Explorer passe souvent le nom de l'utilisateur à Apache en utilisant automatiquement la casse avec laquelle le compte a été créé.

Le résultat final est que vous pouvez avoir besoin d'au moins deux entrées dans votre `AuthzSVNAccessFile` pour chaque utilisateur - une entrée minuscule et une entrée dans la même casse que celle qu'Internet Explorer passe à Apache. Vous devrez aussi apprendre à vos utilisateurs à aussi taper leurs accréditations en utilisant les minuscules quand ils accèdent aux référentiels via TortoiseSVN.

Les journaux d'Erreurs et d'Accès d'Apache sont vos meilleurs amis dans le déchiffrement de problèmes comme ceux-ci puisqu'ils vous aideront à déterminer la chaîne du nom d'utilisateur passée par le module `AuthzSVNAccessFile` de Subversion. Vous pouvez avoir besoin d'expérimenter avec le format exact de la chaîne de l'utilisateur dans le `svnaccessfile` (par exemple `DOMAINE\utilisateur` contre `DOMAINE//utilisateur`) pour que tout fonctionne.

3.7.8. Plusieurs sources d'authentification

Il est aussi possible d'avoir plus d'une source d'authentification pour votre référentiel Subversion. Pour ce faire, vous devez rendre chaque type d'authentification non définitive, pour qu'Apache vérifie plusieurs sources pour la correspondance nom utilisateur/mot de passe.

Un scénario fréquent consiste à utiliser l'authentification par le domaine Windows et par fichier `passwd`, pour que vous puissiez fournir un accès SVN aux utilisateurs qui ne possèdent pas de login dans le domaine Windows.

- Pour activer l'authentification via le domaine Windows et par un ficher d'authentification `passwd`, ajoutez les lignes suivantes dans le bloc `<Location>` de votre fichier de configuration Apache :

```
AuthBasicAuthoritative Off
SSPIAuthoritative Off
```

Voici un exemple de configuration complète d'Apache pour une authentification combinant le domaine Windows et le fichier d'authentification `passwd` :

```
<Location /svn>
  DAV svn
  SVNListParentPath on
  SVNParentPath D:\SVN

  AuthName "Subversion repositories"
  AuthzSVNAccessFile svnaccessfile.txt

# NT Domain Logins.
  AuthType SSPI
  SSPIAuth On
  SSPIAuthoritative Off
  SSPIDomain <domaincontroller>
  SSPIOfferBasic On

# Htpasswd Logins.
  AuthType Basic
  AuthBasicAuthoritative Off
  AuthUserFile passwd

  Require valid-user
</Location>
```

3.7.9. Sécuriser le serveur avec SSL

Même si Apache 2.2.x permet le support OpenSSL, ce n'est pas activé par défatu. Vous devez l'activer manuellement.

1. Dans le fichier de configuraton apache, décommentez les lignes:

   ```
   #LoadModule ssl_module modules/mod_ssl.so
   ```

   et en bas

   ```
   #Include conf/extra/httpd-ssl.conf
   ```

   changez alors la ligne (sur une seule ligne)

   ```
   SSLMutex "file:C:/Program Files/Apache Software Foundation/\
   Apache2.2/logs/ssl_mutex"
   ```

   en

   ```
   SSLMutex default
   ```

2. Ensuite vous devez créer un certificat SSL. Pour faire cela ouvrez une invite de commande (la boîte DOS) et allez au dossier d'Apache (par exemple `C:\program files\apache group \apache2`) et tapez la commande suivante :

   ```
   bin\openssl req -config conf\openssl.cnf -new -out mon-serveur.csr
   ```

   On vous demandera une phrase de mot de passe. N'utilisez pas s'il vous plaît de mots simples, mais des phrases entières, par exemple une partie d'une poésie. Plus la phrase est longue, mieux c'est. Vous devez aussi entrer l'URL de votre serveur. Toutes les autres questions sont facultatives mais nous vous recommandons de remplir celles-ci aussi.

   Normalement le fichier `privkey.pem` est créé automatiquement, mais si ce n'est pas le cas vous devez exécuter cette commande pour le générer:

   ```
   bin\openssl genrsa -out conf\privkey.pem 2048
   ```

   Ensuite, tapez les commandes

   ```
   bin\openssl rsa -in conf\privkey.pem -out conf\server.key
   ```

   et (sur une seule ligne)

   ```
   bin\openssl req -new -key conf\server.key -out conf\server.csr \
   -config conf\openssl.cnf
   ```

   puis (sur une seule ligne)

   ```
   bin\openssl x509 -in conf\server.csr -out conf\server.cert
                    -req -signkey conf\server.key -days 4000
   ```

   Cela créera un certificat qui expirera dans 4000 jours. Et enfin écrivez (sur une seule ligne) :

   ```
   bin\openssl x509 -in conf\server.cert -out conf\server.der.crt
                    -outform DER
   ```

   Ces commandes ont créé quelques fichiers dans le répertoire `conf` de Apache (`server.der.crt`, `server.csr`, `server.key`, `.rnd`, `privkey.pem`, `server.cert`).

3. Redémarrez le service Apache.

4. Allez avec votre navigateur à `https://servername/svn/project` ...

SSL et Internet Explorer

Si vous sécurisez votre serveur avec SSL et utilisez l'authentification avec un domaine Windows vous vous apercevrez que la navigation dans le référentiel avec Internet Explorer ne marche plus. Ne vous inquiétez pas - c'est seulement qu'Internet Explorer n'est pas capable d'authentifier. Les autres navigateurs n'ont pas ce problème et TortoiseSVN et les autres clients Subversion sont toujours capable d'authentifier.

Si vous voulez toujours utiliser IE pour parcourir le référentiel vous pouvez soit :

- définissez une directive `<Location /path>` dans le fichier de configuration d'Apache et ajoutez `SSPIBasicPreferred On`. Cela permettra à IE d'authentifier à nouveau,

mais les autres navigateurs et Subversion ne seront pas capables d'authentifier pour cet emplacement.

- Offrir de naviguer avec une authentification non cryptée (sans SSL) aussi. Étrangement IE n'a pas de problèmes avec l'authentification si la connexion n'est pas sécurisée avec SSL.
- Dans l'installation "standard" de ssl, il y a souvent la déclaration suivante dans l'hôte SSL virtuel d'Apache :

```
SetEnvIf User-Agent ".*MSIE.*" \
             nokeepalive ssl-unclean-shutdown \
             downgrade-1.0 force-response-1.0
```

Il y a (avait ?) de bonnes raisons pour cette configuration, regardez *http://www.modssl.org/docs/2.8/ssl_faq.html#ToC49* Mais si vous voulez l'authentification NTML vous devez utiliser `keepalive`. Si vous décommentez entièrement `SetEnvIf` vous devriez être capables d'authentifier IE avec l'authentification windows avec SSL et Apache sur Win32 avec `mod_auth_sspi` inclus.

Forcer l'accès SSL

Une fois que vous avez mis en place SSL pour rendre votre référentiel plus sécurisé, vous pourriez vouloir désactiver l'accès normal non-SSL (http) et permettre seulement l'accès https. Pour ce faire, vous devez ajouter une autre directive au bloc `<Location>` de Subversion : `SSLRequireSSL`.

Voici un exemple de bloc `<Location>` :

```
<Location /svn>
  DAV svn
  SVNParentPath D:\SVN
  SSLRequireSSL
  AuthType Basic
  AuthName "Subversion repositories"
  AuthUserFile passwd
  #AuthzSVNAccessFile svnaccessfile
  Require valid-user
</Location>
```

3.7.10. Utiliser des certificats avec des hôtes SSL virtuels

Envoyé à la liste de diffusion TortoiseSVN par Nigel Green. Merci !

Avec certaines configurations de serveur, vous pouvez avoir à mettre en place un seul serveur avec 2 hôtes SSL virtuels: Le premier pour les accès internet publics, ne nécessitant aucun certificat côté client. Et un second, sécurisé avec certificat côté client, où tourne le serveur Subversion.

Dans le fichier de configuration de Apache, ajouter une directive `SSLVerifyClient Optional` à la section *per-server* (i.e. en dehors de balises `VirtualHost` et `Directory`) force Apache à demander un certificat au client au moment d'établir la connexion SSL. A cause d'un bug dans le module `mod_ssl` il est crucial de demander le certificat à ce moment car sinon il ne fonctionnera plus en cas de tentative de re-connexion SSL.

La solution est d'ajouter la directive suivante dans le répertoire du virtual host que vous voulez réserver à Subversion:

```
SSLRequire %{SSL_CLIENT_VERIFY} eq "SUCCESS"
```

Cette directive autorise l'accès au répertoire seulement si le certificat du client a été reçu et validé.

Pour résumer; les lignes utiles du fichier de configuration Apache sont:

```
SSLVerifyClient Optional

### Configuration du virtual host public
### (ne nécessitant pas de certificat)

<VirtualHost 127.0.0.1:443>
  <Directory "chemin du repertoire racine public">
  </Directory>
</VirtualHost>

###  Configuration du virtual host pour SUBVERSION
### (nécessitant une certificat)
<VirtualHost 127.0.0.1:443>
  <Directory "chemin du repertoire racine de subversion">
    SSLRequire %{SSL_CLIENT_VERIFY} eq "SUCCESS"
  </Directory>

  <Location /svn>
    DAV svn
    SVNParentPath /chemin du referentiel
  </Location>
</VirtualHost>
```

Chapitre 4. Guide d'utilisation quotidienne

Ce document décrit l'utilisation quotidienne du client TortoiseSVN. Ce n'est *pas* une introduction aux systèmes de contrôle de version, *ni* une introduction à Subversion (SVN). C'est plutôt un endroit où vous pouvez regarder quand vous savez approximativement ce que vous voulez faire, mais vous ne vous rappelez pas tout à fait comment le faire.

If you need an introduction to version control with Subversion, then we recommend you read the fantastic book: *Version Control with Subversion* [http://svnbook.red-bean.com/].Si vous avez besoin d'une introduction au contrôle de version avec Subversion, alors nous vous recommandons la lecture du livre fantastique : *Version Control with Subversion* [http://svnbook.red-bean.com/].

Ce document est aussi un travail en cours, de même que TortoiseSVN et Subversion. Si vous trouvez des erreurs, veuillez les annoncer sur la mailing list pour que nous puissions mettre à jour la documentation. Certaines copies d'écran dans le Guide d'Utilisation Quotidienne (GUQ) pourraient ne pas refléter l'état actuel du logiciel. Veuillez nous pardonner. Nous travaillons sur TortoiseSVN pendant notre temps libre.

Pour se servir au mieux du guide d'utilisation quotidienne :

- Vous devez avoir déjà installé TortoiseSVN.
- Vous devriez être familier avec les systèmes de contrôle de version.
- Vous devez connaître les bases de Subversion.
- Vous devriez avoir mis en place un serveur et/ou avoir accès à un référentiel Subversion.

4.1. Pour commencer

4.1.1. Recouvrement d'icônes

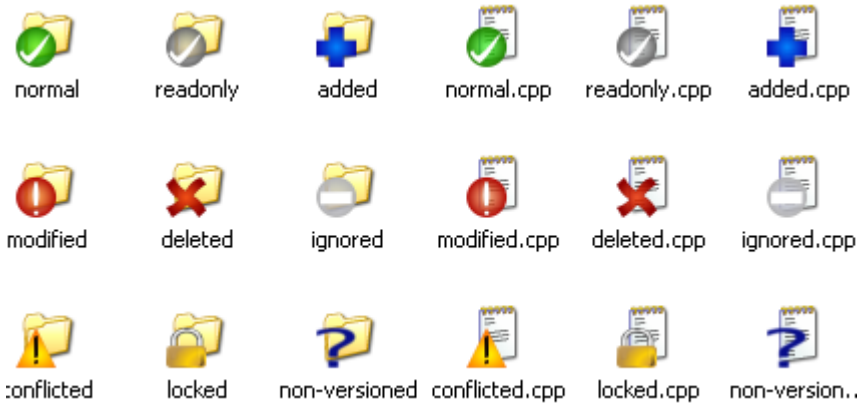


Figure 4.1. L'Explorateur montrant le recouvrement d'icônes

Une des fonctionnalités les plus visibles de TortoiseSVN sont les recouvrements d'icônes qui apparaissent sur les fichiers de votre copie de travail. Celles-ci vous montrent en un clin d'oeil quels fichiers ont été modifiés. Référez-vous à Section 4.7.1, « Recouvrement d'icônes » pour découvrir que représentent les différentes icônes.

4.1.2. Menus contextuels

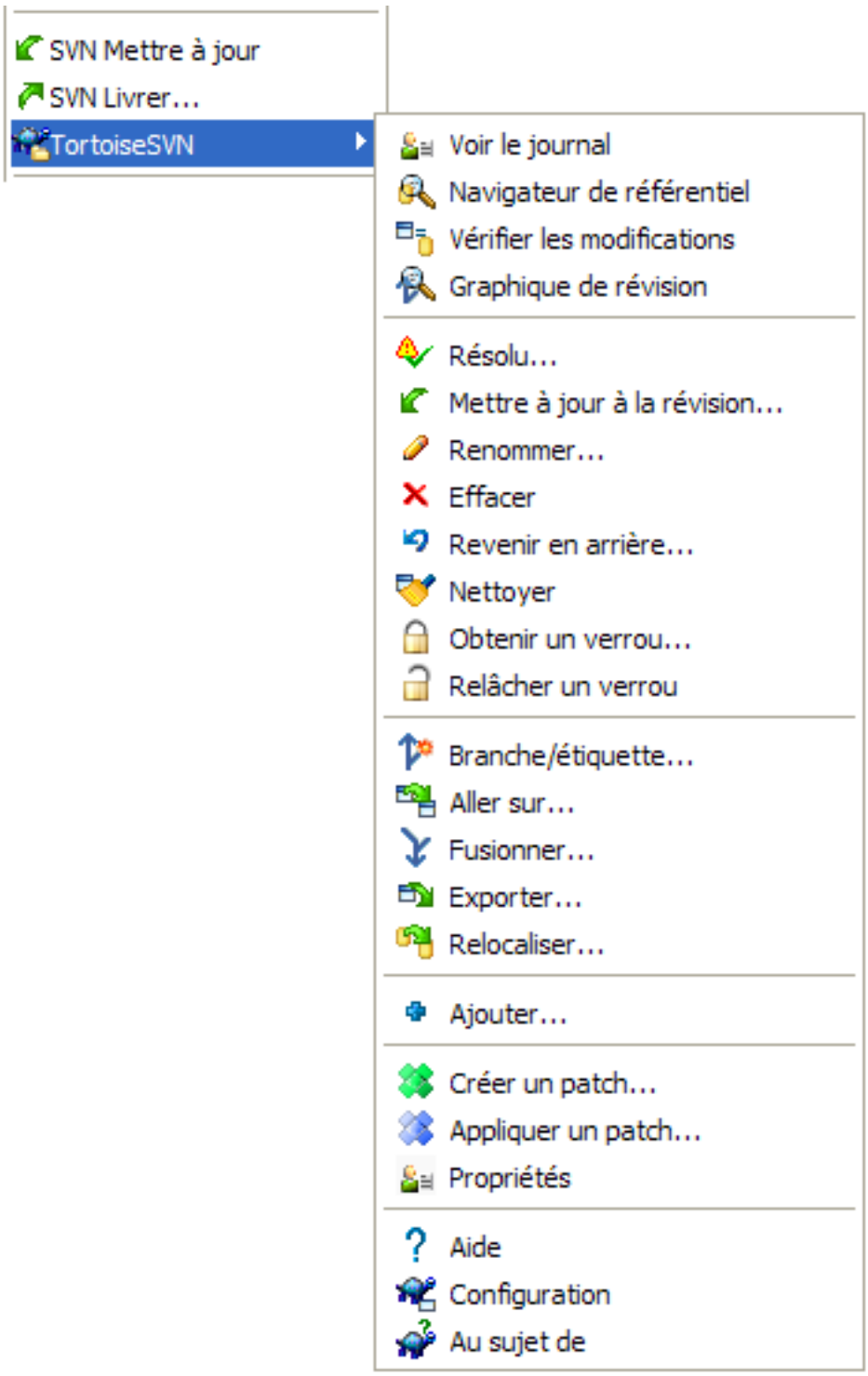


Figure 4.2. Menu contextuel pour un répertoire sous contrôle de version

Toutes les commandes de TortoiseSVN sont invoquées à partir du menu contextuel de l'explorateur Windows. La plupart sont directement visibles quand vous faites un clic droit sur un fichier ou sur un dossier. Les commandes disponibles dépendent si le fichier ou le dossier ou son dossier parent sont sous contrôle de version ou non. Vous pouvez aussi voir le menu TortoiseSVN comme un partie du menu fichier de l'Explorateur.

Astuce

Quelques commandes qui sont très rarement utilisées ne sont accessibles que dans l'extension du menu contextuel. Pour afficher cette extension, appuyer sur **Shift** au moment où vous faites le click-droit.

Dans certains cas, vous pouvez voir plusieurs entrées TortoiseSVN. Ce n'est pas un bug !

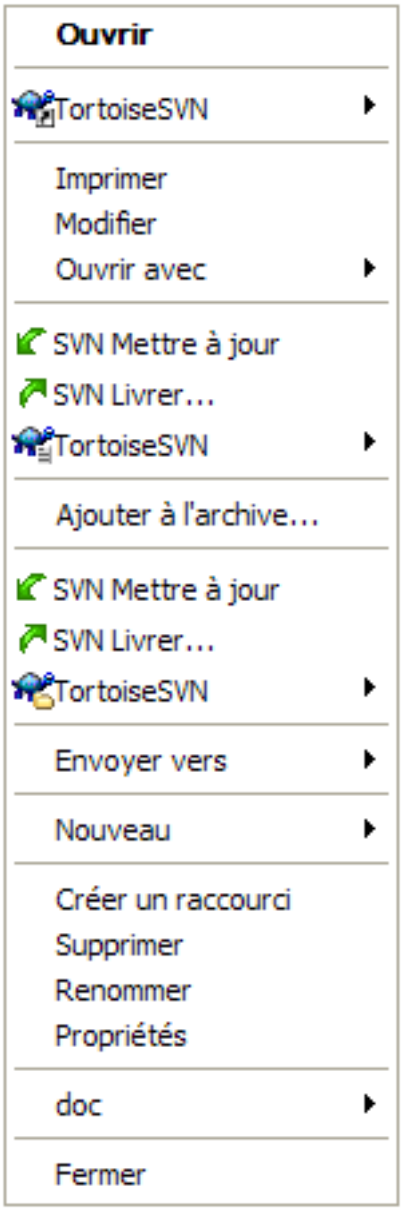


Figure 4.3. Menu fichier de l'Explorateur pour un raccourci dans un répertoire non versionné

Cet exemple est pour un raccourci non versionné dans un dossier versionné et dans le menu de fichier de l'Explorateur il y a *trois* entrées pour TortoiseSVN. L'une est pour le dossier, une autre pour le raccourci lui-même et la troisième pour l'objet sur lequel pointe le raccourci. Pour vous aider à les distinguer entre elles, les icônes ont un indicateur dans le coin inférieur droit pour montrer si l'entrée de menu est pour un fichier, un dossier, un raccourci ou pour des plusieurs éléments sélectionnés.

Si vous êtes un utilisateur de Windows 2000 vous verrez le menu contextuel sans icone. Nous savons que cela fonctionnait dans les versions précédentes, mais Microsoft has changed the way its icon handlers work for Vista, requiring us to use a different display method which unfortunately does not work on Windows 2000.

4.1.3. Glisser-déposer

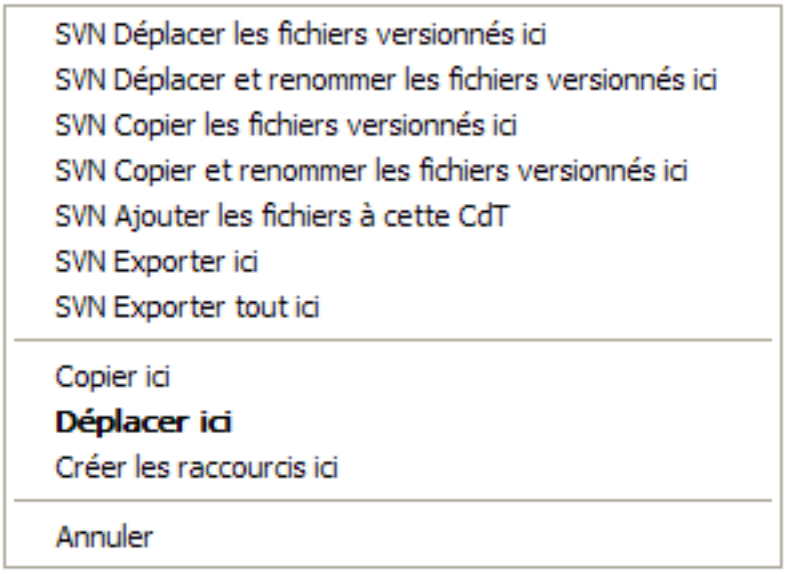


Figure 4.4. Menu pour un répertoire sous contrôle de version lors d'un glisser-déplacer avec le bouton droit

D'autres commandes sont disponibles par glisser-déposer, quand vous glissez-déposez avec le bouton droit des fichiers ou des dossiers vers un nouvel emplacement à l'intérieur des copies de travail ou quand vous glissez-déposez avec le bouton droit un fichier non versionné ou un dossier dans un répertoire qui est sous contrôle de version.

4.1.4. Raccourcis communs

Quelques opérations communes ont des raccourcis Windows bien connus, mais qui n'apparaissent ni sur les boutons ni dans les menus. Si vous ne savez pas comment faire quelque chose d'évident, comme le rafraîchissement d'une vue, regardez ici.

F1
: L'aide, bien sûr.

F5
: Rafraîchir la vue courante. C'est peut-être la commande d'une touche la plus utile. Par exemple... Dans l'Explorateur cela rafraîchira le recouvrement des icônes sur votre copie de travail. Dans la boîte de dialogue de livraison il re-parcourra la copie de travail pour voir ce qui peut avoir besoin d'être livré. Dans la boîte de dialogue du journal il entrera à nouveau en contact avec le référentiel pour vérifier s'il y a des changements plus récents.

Ctrl-A
: Sélectionner tout. Cela peut être utilisé si vous obtenez un message d'erreur et que vous voulez copier-coller dans un email. Utilisez Ctrl-A pour choisir le message d'erreur et ensuite...

Ctrl-C
: ... Copier le texte choisi.

4.1.5. Authentification

Si le référentiel auquel vous essayez d'avoir accès est protégé par un mot de passe, une boîte de dialogue d'identification s'affichera.

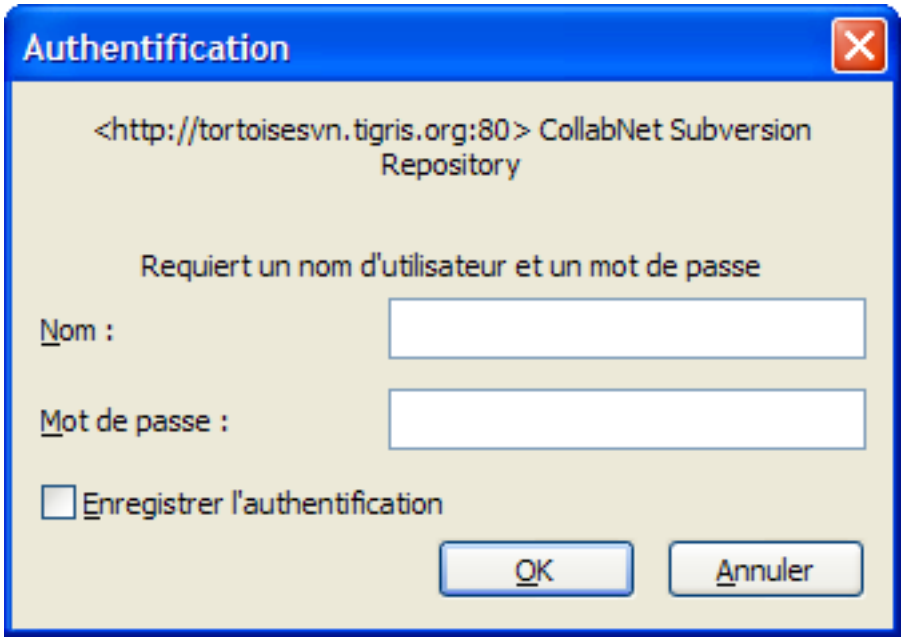


Figure 4.5. Boîte de dialogue d'authentification

Entrez votre nom d'utilisateur et votre mot de passe. La case à cocher fera que TortoiseSVN stockera les accréditations dans le répertoire de Subversion par défaut : `%APPDATA%\Subversion\auth` dans trois sous-répertoires :

- `svn.simple` contains credentials for basic authentication (username/password).
- `svn.ssl.server` contient des certificats serveur SSL.
- `svn.username` contient les accréditations pour l'authentification avec le nom d'utilisateur uniquement (aucun mot de passe nécessaire).

Si vous voulez effacer le cache d'authentification pour tous les serveurs, vous pouvez le faire à partir de la page Données Sauvegardées de la boîte de dialogue de configuration de TortoiseSVN. Ce bouton effacera toutes les données d'authentification gardées en cache dans les répertoire `auth` de Subversion, aussi bien que les authentifications stockées par des versions précédemment installées de TortoiseSVN. Référez-vous à Section 4.30.6, « Configuration des données sauvegardées ».

Certains préfèrent que leurs données d'identification soient supprimées à l'ouverture ou à la fermeture de Windows. Le moyen de le faire est d'utiliser un script de fermeture pour supprimer le répertoire `%APPDATA%\Subversion\auth`, i.e.

```
@echo off
rmdir /s /q "%APPDATA%\Subversion\auth"
```

Vous pouvez trouver le mode d'emploi pour installer ce type de script à cette adresse *windows-help-central.com* [http://www.windows-help-central.com/windows-shutdown-script.html].

Pour plus d'informations sur la façon de mettre en place l'authentification et le contrôle d'accès pour votre serveur, référez-vous à Section 3.5, « Accéder au référentiel »

4.1.6. Maximiser les fenêtres

Beaucoup de fenêtres de TortoiseSVN ont nombre d'informations à afficher, mais il est souvent plus utile de maximiser leur largeur ou leur hauteur plutôt que de les afficher en plein écran. Pour ce faire, il y a des raccourcis dans le bouton Maximiser. Utilisez le bouton du milieu de la souris pour agrandir la hauteur, et le bouton droit de la souris pour élargir.

4.2. Importer des données dans un référentiel

4.2.1. Importer

If you are importing into an existing repository which already contains some projects, then the repository structure will already have been decided. If are importing data into a new repository then it is worth taking the time to think about how it will be organised. Read Section 3.1.5, « Disposition du référentiel » for further advice.

Cette section décrit la commande Importer, qui a été faite pour importer une arborescence de répertoires dans le référentiel en une action. Bien que cette commande fonctionne, elle a quelques défauts:

- Il n'y a aucun moyen de sélectionner des éléments à inclure, à part en utilisant le filtre global des fichiers à ignorer.
- Le répertoire importé ne devient pas une copie de travail. Vous devez faire une extraction pour en avoir une directement du serveur.
- Il est aisé de se tromper de répertoire à importer dans le référentiel.

Pour ces raisons nous recommandons d'utiliser la méthode en deux étapes décrite dans : Section 4.2.2, « Importer en place » au lieu de cette commande . Mais puisque vous êtes là, c'est le fonctionnement de la commande Importer...

Avant que vous n'importiez votre projet dans un référentiel, vous devriez :

1. Supprimez tous les fichiers qui ne sont pas nécessaires pour générer le projet (fichiers temporaires, fichiers produits par un compilateur par exemple *.obj, binaires compilés...)
2. Organisez les fichiers en dossiers et sous-dossiers. Bien qu'il soit possible de renommer/déplacer les fichiers plus tard, il est fortement recommandé de fixer la structure de votre projet juste avant l'importation !

Choisissez maintenant le dossier de niveau supérieur de votre structure de répertoire de projet dans l'explorateur Windows et faite un clic droit pour ouvrir le menu contextuel. Choisissez la commande TortoiseSVN → Importer... qui fait s'afficher une boîte de dialogue :

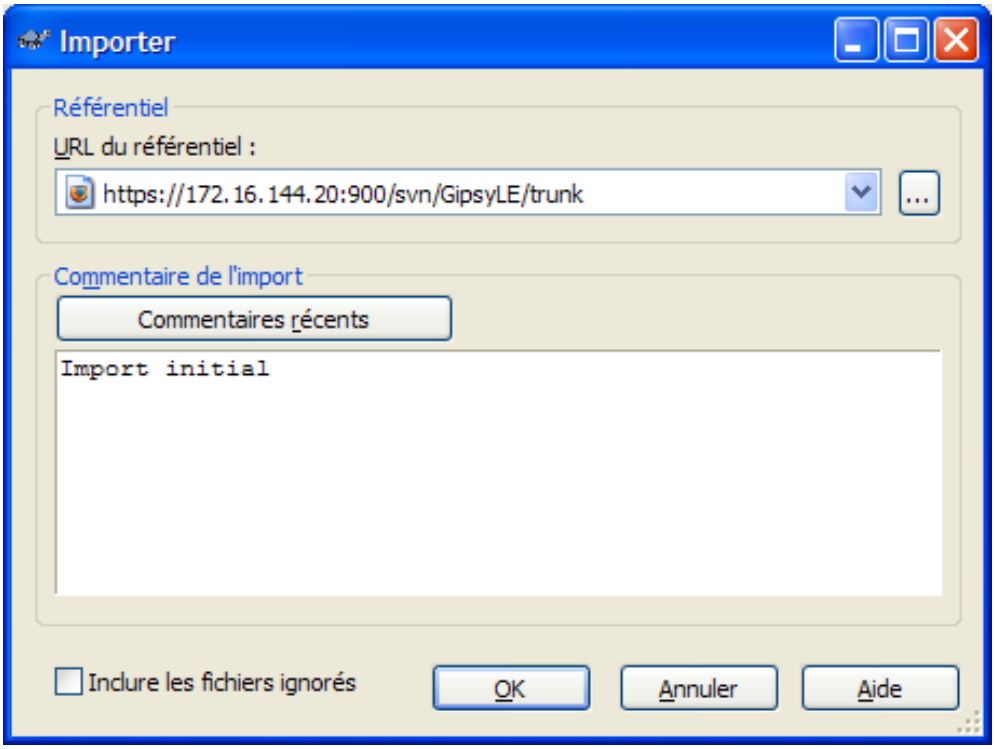


Figure 4.6. Le boîte de dialogue Importer

Dans la boite de dialogue, vous devez entrer l'URL du référentiel dans lequel vous souhaitez importer le projet. Il est important de comprendre que le répertoire local que vous importez n'apparaitra pas dans le référentiel, mais juste son contenu.Par exemple, si vous avez la structure suivante :

```
C:\Projects\Widget\source
C:\Projects\Widget\doc
C:\Projects\Widget\images
```

et que vous importez `C:\Projects\Widget` dans `http://monnomdedomaine.com/svn/trunk` vous serez alors surpris de voir que vos sous répertoires iront directement dans `trunk` au lieu d'être dans un répertoire `Widget`. Vous devez spécifier le sous répertoire comme faisant partie de l'URL, `http://monnomdedomaine.com/svn/trunk/Widget-X`. Notez que la commande importer créera automatiquement les sous répertoires dans le référentiel s'ils n'existent pas.

Le message d'importation est utilisé comme un commentaire.

Par défaut, les fichiers et les dossiers qui correspondent au filtre d'exclusion ne sont *pas* importés. Pour ignorer ce comportement, vous pouvez utiliser la case à cocher Inclure les fichiers ignorés. Référez-vous à Section 4.30.1, « Configuration générale » pour plus d'informations sur la configuration d'un filtre d'exclusion.

Dès que vous appuyez sur OK TortoiseSVN importe dans le référentiel l'arborescence complète des répertoires, fichiers compris. Le projet est maintenant sous contrôle de version dans le référentiel. Veuillez noter que le dossier que vous avez importé n'est *PAS* sous contrôle de version ! Pour obtenir une *copie de travail* sous contrôle de version, vous devez faire une extraction de la version que vous venez d'importer. Ou vous renseigner sur la manière d'importer un répertoire déjà en place.

4.2.2. Importer en place

Si vous disposez déjà d'un référentiel, et que vous souhaitez y ajouter une nouvelle arborescence, suivez les étapes suivantes :

1. Utiliser l'explorateur de référentiel pour créer un nouveau projet directement dans le référentiel.

2. Extrayez le nouveau répertoire parent du répertoire que vous voulez importer. Un message va s'afficher, indiquant que le répertoire n'est pas vide. Vous disposez maintenant d'un répertoire parent versionné, dont le contenu n'est pas sous contrôle de version.

3. Sélectionnez TortoiseSVN → Ajouter... sur le répertoire sous contrôle de version pour ajouter un ou des éléments. Vous pouvez ajouter ou supprimer des fichiers, activer les propriétés `svn:ignore` sur les répertoires et faire toutes les modifications dont vous avez besoin.

4. Livrez le répertoire de plus haut niveau, et vous aurez une nouvelle arborescence versionnée, et une copie de travail locale, créée depuis votre répertoire existant.

4.2.3. Fichiers spéciaux

Parfois vous avez besoin d'avoir un fichier sous contrôle de version qui contient des données spécifiques aux utilisateurs. Cela signifie que vous avez un fichier que chaque développeur/utilisateur doit modifier pour convenir à son paramétrage local. Mais versionner un tel fichier est difficile car chaque utilisateur livrerait son propre fichier à chaque fois.

Dans de tels cas nous suggérons d'utiliser des fichiers *templates*. Vous créez un fichier qui contient toutes les données dont vos développeurs auront besoin, ajoutez ce fichier au contrôle de version et laissez les développeurs extraire ce fichier. Alors, chaque développeur doit *faire une copie* de ce fichier et la renommer. Après cela, modifier la copie n'est plus un problème.

Comme exemple, vous pouvez regarder le script de construction de TortoiseSVN. Il appelle un fichier nommé `TortoiseVars.bat` qui n'existe pas dans le référentiel, seul le fichier `TortoiseVars.tmpl` y est. Ce denier sert de modèle afin que chaque développeur créée son propre `TortoiseVars.bat`. À l'intérieur de ce fichier, nous avons ajouté des commentaires pour que les utilisateurs voient quelles lignes ils doivent éditer et changer selon leur paramétrage local pour qu'il fonctionne.

Afin de ne pas déranger les utilisateurs, nous avons aussi ajouté le fichier `TortoiseVars.bat` à la liste des ignorés de son dossier parent, c'est-à-dire que nous avons mis la propriété Subversion `svn:ignore` pour inclure ce nom de fichier. De cette façon il n'apparaîtra pas comme non versionné à chaque livraison.

4.3. Extraire une copie de travail

Pour obtenir une copie de travail vous devez faire une *extraction* à partir d'un référentiel.

Choisissez un répertoire dans l'explorateur Windows où vous voulez placer votre copie de travail. Faites un clic droit pour afficher le menu contextuel et choisissez la commande TortoiseSVN → Extraire..., qui affiche la boîte de dialogue suivante :

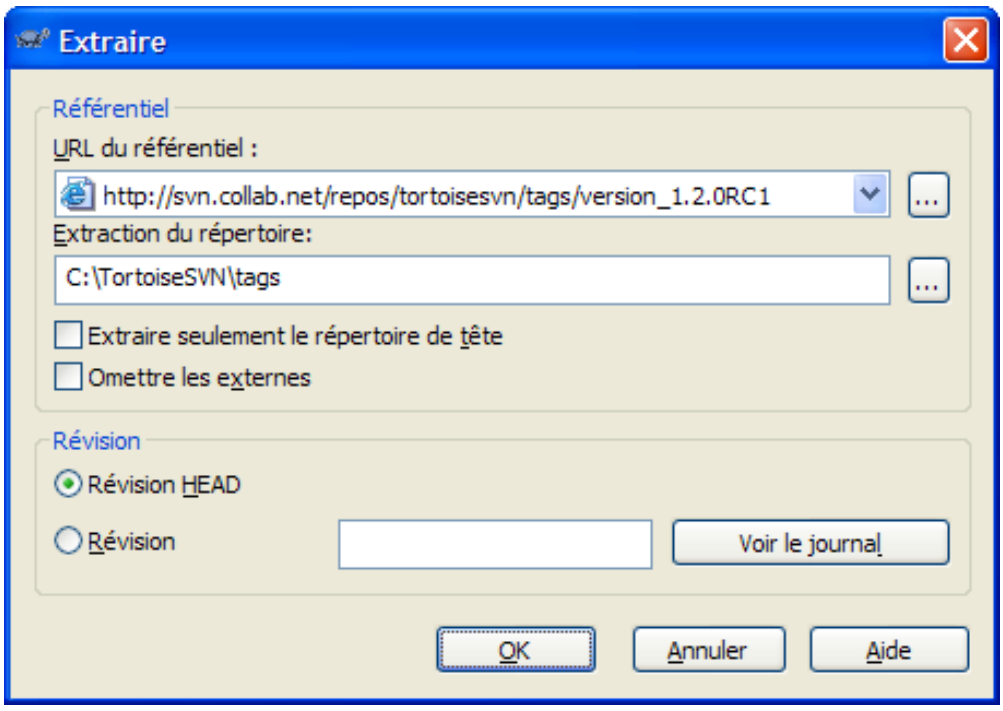


Figure 4.7. La boîte de dialogue Extraire

Si vous entrez un nom de dossier qui n'existe pas, alors il sera créé.

4.3.1. Profondeur d'extraction

Vous pouvez choisir la *profondeur* que vous voulez extraire, ce qui vous permet de spécifier la profondeur de récursion dans les sous répertoires. Si vous voulez juste une partie d'une grosse arborescence , Vous pouvez n'extraire que le répertoire de plus haut niveau, et mettre à jour les sous répertoires petit à petit.

Totalement récursive
: Extraire l'arborescence complète, incluant récursivement tous les sous répertoires.

Descendants directs, y compris les répertoires
: Extrait le répertoire spécificé, tous les fichiers et sous répertoires compris, mais ne remplit pas les sous répertoires.

Juste les fichiers
: Extrait le répertoire spécifie, y compris les fichiers mais n'extrait aucune sous répertoire.

Uniquement cet élément.
: Extraire juste le répertoire. Ne pas le remplir avec les répertoires et fichiers enfants.

Copie de travail
: Se souvenir de la profondeur dans la version de travail. Cette option n'est pas utilisée dans la fenêtre d'extraction, mais c'est la valeur qui sera utilisée par éfaut pour toutes les autres fenêtres ayant cette option.

Exclure
: Utilisé pour réduire la profondeur de la copie de travail déjà rempli. Cette option n'est disponible que dans la fenêtre Mettre à jour à la révision.

Si vous n'extrayez qu'une portion de l'arboresence (en décochant la case `récursivement`), vous pouvez lister d'autres sous répertoires via l'explorateur de référentiel (Section 4.24, « l'explorateur de référentiel ») ou la fenêtre de vérification des modifications (Section 4.7.3, « Statut local et distant »).

Dans l'explorateur de référentiel, faites un click droit sur le répertoire extrait, puis utilisez TortoiseSVN → Explorateur de référentiel pour afficher l'explorateur de référentiel. Trouvez le sous répertoire que

vous souhaitez ajouter à votre version de travail, et utilisez Menu contextuel → Mettre à jour à la révision.... Ce menu ne sera visible que si les éléments sélectionnés n'existent pas dans votre copie de travail, et que l'élément parent y est.

Dans la fenêtre de vérification des modifications, cliquez d'abord sur le bouton Vérifier le référentiel. La fenêtre affichera tous les fichiers et répertoires du référentiel mais que vous n'avez pas exporté comme étant `ajouté à distance`. Faites un Click droit sur le(s) répertoire(s) que vous voudriez ajouter à votre CdT, puis utilisez Menu contextuel → Mise à jour.

Cette fonctionnalité est très utile lorsque vous ne voulez récupérer qu'une partie d'une grosse arborescence, tout en gardant le côté pratique d'une mise à jour d'un seul élément. Supposez que vous avez une grosse arborescence avec des sous répertoires nommés `Projet01` à `Projet99`, et que vous ne vouliez récupérer que `Projet03`, `Projet25` et `Projet76/SousProj`. Suivez ces étapes :

1. Extrait le répertoire parent avec la profondeur « Juste cet élément » Vous avez maintenant un répertoire de haut niveau vide.

2. Sélectionnez le nouveau répertoire et utilisez la commande TortoiseSVN → Explorateur de référentiel pour voir le contenu du référentiel.

3. Faites un click droit sur `Project03` et Menu contextuel → Mettre à jour à la révision.... Laissez les paramètres par défaut et cliquez sur OK. Ce répertoire est a présent complet.

   Répéter le même processus pour `Project25`..

4. Allez dans `Project76/SubProj` et faites de même. Cette fois ci, notez que le répertoire `Project76` est vide excepté `SubProj`, lequel est complètement rempli. Subversion a créé pour vous les répertoires intermédiaires sans les remplir.

Modifier la profondeur de la copie de travail

Une fois que vous avez extrait une copie de travail à une certaine profondeur vous pouvez modifier cette profondeur à postériori en utilisant Menu contextuel → Mettre à jour l'élément à la révision....

Utilisant un serveur ancien

Les serveurs de version inférieure à 1.5 ne supportent pas la notion de profondeur de la copie de travail, ils n'interprêtent donc pas efficacement les requêtes l'utilisant. Les commande fonctionneront, mais un vieux serveur retournera toutes les données, laissant le client filtrer ce qui n'est pas requis, ce qui augmentera le traffic sur le réseau. Si possible, vous devriez mettre à jour votre serveur.

Si le projet contient des références à des projets externes que vous ne voulez *pas* extraire en même temps, utilisez la case à cocher Omettre les externes .

Important

If Omit externals is checked, or if you wish to increase the depth value, you will have to perform updates to your working copy using TortoiseSVN → Update to Revision... instead of TortoiseSVN → Update. The standard update will include all externals and keep the existing depth.

Il est recommandé de n'extraire que la partie `trunk` de l'arborescence de répertoire, ou un de ses sous répertoires. Si vous spécifiez le chemin parent de l'arborescence de répertoire dans l'URL alors vous

pourriez vous retrouver avec un disque dur rempli puisque vous obtiendrez une copie de l'arborescence du référentiel en entier incluant chaque branche et chaque étiquette de votre projet !

Exporter

Parfois vous pouvez avoir envie de créer une copie locale sans aucun de ces répertoires `.svn`, pour créer un paquet zippé de vos sources par exemple. Lisez Section 4.26, « Exporter une copie de travail Subversion » pour savoir comment le faire.

4.4. Livrer vos changements au référentiel

Envoyer les changements que vous avez faits dans votre copie de travail est connue comme *livrer* les changements. Mais avant de livrer vous devez vous assurer que votre copie de travail est à jour. Vous pouvez soit utiliser directement TortoiseSVN → Mettre à jour. Ou vous pouvez utiliser TortoiseSVN → Vérifier les modifications d'abord, pour voir quels fichiers ont été changés localement ou sur le serveur.

4.4.1. La boîte de dialogue Livrer

Si votre copie de travail est à jour et s'il n'y a aucun conflit, vous êtes prêts à livrer vos changements. Choisissez n'importe quel fichier et/ou dossier que vous voulez livrer, puis TortoiseSVN → Livrer....

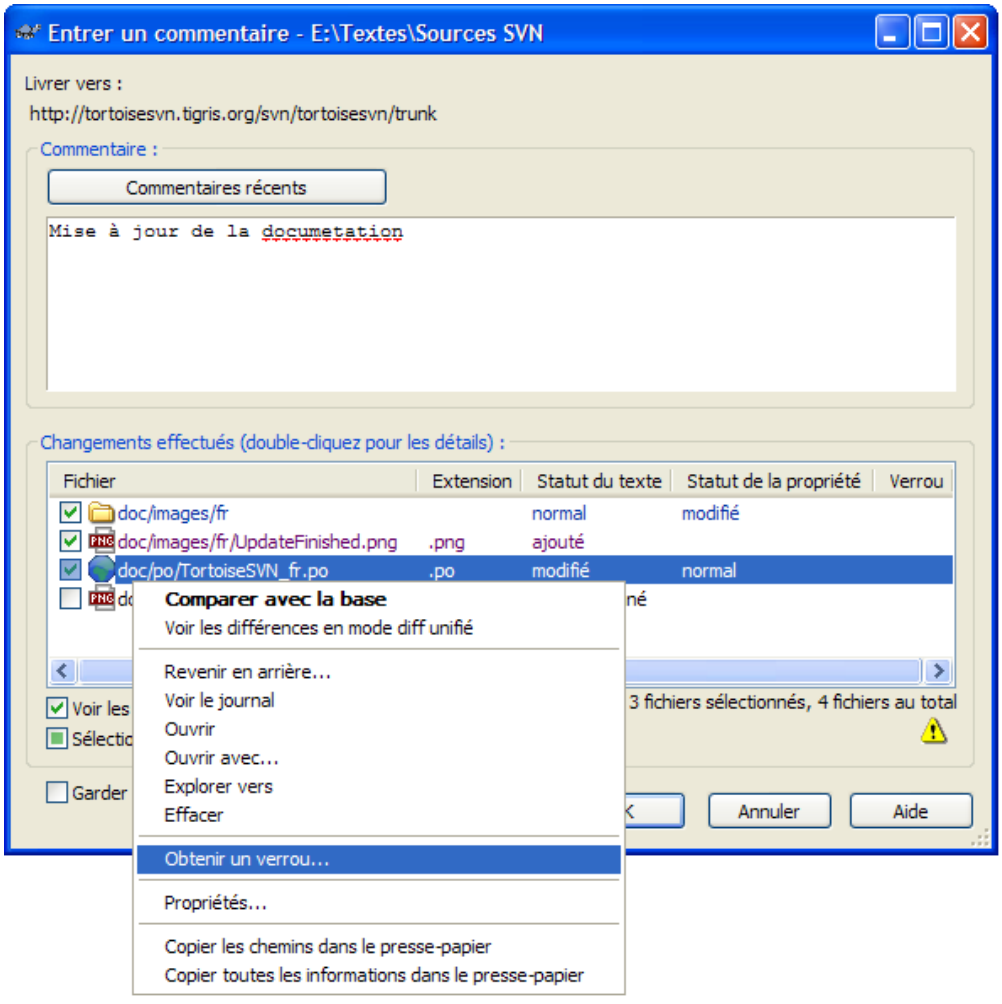


Figure 4.8. La boîte de dialogue Livrer

La boîte de dialogue Livrer vous montrera chaque fichier changé, y compris les fichiers ajoutés, supprimés et non versionnés. Si vous ne voulez pas qu'un fichier modifié soit livré, décochez juste ce fichier. Si vous voulez inclure un fichier non versionné, cochez juste ce fichier pour l'ajouter à la livraison.

Les éléments qui ont été commutés vers un chemin de référentiel différent sont aussi indiqués en utilisant un marqueur `(s)`. Vous pouvez avoir commuté quelque chose en travaillant sur une branche et avoir oublié de rebasculer sur le tronc. C'est votre signal d'alarme !

Livrer des fichiers ou des dossiers ?

Quand vous livrez des fichiers, la boîte de dialogue Livrer montre seulement les fichiers que vous avez choisis. Quand vous livrez un dossier, la boîte de dialogue Livrer choisira les fichiers modifiés automatiquement. Si vous oubliez un nouveau fichier que vous avez créé, livrer le dossier le trouvera de toute façon. La livraison d'un dossier ne veut *pas* dire que chaque fichier est marqué comme changé ; il rendre juste votre vie plus facile en faisant plus de travail pour vous.

Si vous avez modifié des fichiers qui ont été inclus à partir d'un référentiel différent en utilisant `svn:externals`, ces changements ne peuvent pas être inclus dans la même livraison atomique. Un symbole d'avertissement au-dessous de la liste de fichier vous indique si cela se produit et l'info-bulle explique que ces fichiers externes doivent être livrés séparément.

Beaucoup de fichiers non versionnés dans la boîte de dialogue Livrer

Si vous pensez que la boîte de dialogue Livrer de TSVN vous montre trop de fichiers non versionnés (générés par la compilation ou sauvegarde de votre éditeur par exemple), il y a plusieurs solutions. Vous pouvez :

- ajouter le fichier (ou une extension de caractère de remplacement) à la liste de fichiers pour exclure sur la page de configuration. Cela affectera toutes vos copies de travail.
- ajouter le fichier à la liste `svn:ignore` en utilisant TortoiseSVN → Ajouter à la liste des ignorés Cela affectera seulement le répertoire sur lequel vous mettez la propriété `svn:ignore`. En utilisant la boîte de dialogue de propriétés SVN, vous pouvez changer la propriété `svn:ignore` pour un répertoire.

Lire Section 4.13, « Ignorer des fichiers et des répertoires » pour plus d'informations.

Double-cliquer sur un fichier modifié dans la boîte de dialogue Livrer lancera l'outil externe de différenciation pour afficher vos changements. Le menu contextuel vous donnera plus d'options, comme affiché sur la capture d'écran. Vous pouvez aussi glissez les fichiers vers une autre application comme un éditeur de texte ou un IDE à partir d'ici.

Vous pouvez sélectionner ou désélectionner des éléments en cliquant sur la case à coser située à leur gauche. Pour les répertoires, vous pouvez utiliser **Shift**-sélectionner pour le faire récursivement.

Les colonnes affichées dans le panneau du bas sont personnalisables. Si vous faites un click droit sur n'importe quel en-tête de colonne vous verrez un menu contextuel permettant de choisir quelles colonnes afficher. Vous pouvez aussi changer la largeur des colonnes en faisant glisser le bord des entêtes. Ces personnalisations sont gardées, donc vous verrez les mêmes en-tête la prochaine fois.

Par défaut, lorsque que vous livrez une version, tous les verrous que vous gardez sur des fichiers sont automatiquement retirés à la fin de la livraison. Si vous souhaitez conserver ces verrous, cochez la case Garder les verrous. L'état par défaut de cette case, est récupéré dans le fichier de configuration de Subversion, à la clé `no_unlock`. Lisez Section 4.30.1, « Configuration générale » pour plus d'informations sur l'édition du fichier de configuration de Subversion.

Glisser-déposer

Vous pouvez glisser des fichiers dans la boîte de dialogue Livrer d'ailleurs, tant que les copies de travail sont extraites du même référentiel. Par exemple, vous pouvez avoir une copie de travail énorme avec plusieurs fenêtres d'explorateur ouvertes pour regarder les dossiers éloignés de la hiérarchie. Si vous voulez éviter de livrer le dossier de niveau supérieur (avec un long parcours du dossier pour vérifier les changements) vous pouvez ouvrir la boîte de dialogue de livraison pour un répertoire et y glisser des éléments des autres fenêtres pour les inclure dans la même livraison atomique.

Vous pouvez faire glisser les fichiers non versionnés de votre copie de travail dans la fenêtre de livraison, ils seront alors automatiquement ajoutés au référentiel.

Réparation des renommages externes

Parfois, les fichiers sont renommés en dehors de Subversion, et son donc affichés comme fichiers manquants et non versionnés. Pour éviter de perdre l'historique vous devez indiquer à Subversion la connexion. Sélectionnez le nom manquant (nom précédent) et le nouveau et utilisez Menu Contextuel → Réparer le Déplacement pour associer les deux fichiers au renommage.

4.4.2. Listes de changements

La fenêtre de livraison supporte la fonctionnalité liste des modifications de Subversion et s'en sert pour grouper les fichiers. Pour en savoir plus sur cette fonctionnalité vois Section 4.8, « Listes de changements ».

4.4.3. Exclude des éléments de la livraison

Parfois des fichiers sont fréquemment modifiés mais vous ne voulez pas vraiment en faire la livraision. Cela peut montrer une faille dans votre environnement de travail - Pourquoi ces fichiers sont-ils versionnés ? Ne devriez vous pas utiliser des fichiers templates ? Cependant, c'est parfois inévitable. Une cause classique est que votre fichier de projet contient la date et l'heure de la dernière compilation. Le fichier de projet doit être versionné dans la mesure où il contient toutes les propriétés de compilation, mais pas nécessairement livré vu que seul ce timestamp a changé.

Pour améliorer le traitement de cas bizarres, nous avons créé une liste de modification appelée `ignorer à la livraison`. Tout fichier ajouté dans cette liste sera automatiquement décoché dans la fenêtre de livraison. Vous pouvez cependant toujours livrer les modifications en cochant manuellement les fichiers à livrer dans la fenêtre de livraison.

4.4.4. Commentaires de livraison

Assurez-vous d'entrer un commentaire qui décrit les changements que vous livrez. Cela vous aidera à voir ce qui est arrivé et quand, lorsque vous naviguerez dans les commentaires du projet à une date ultérieure. Le message peut être aussi long ou aussi court que vous le souhaitez ; beaucoup de projets ont des directives pour ce qui devrait être inclus, la langue à utiliser et parfois même un format strict.

Vous pouvez appliquer un formatage simple à vos commentaires en utilisant une convention similaire à celle utilisée dans les emails. Pour appliquer un style au `text`, utilisez `*texte*` pour le gras, `_texte_` pour souligner, et `^texte^` pour l'italique.

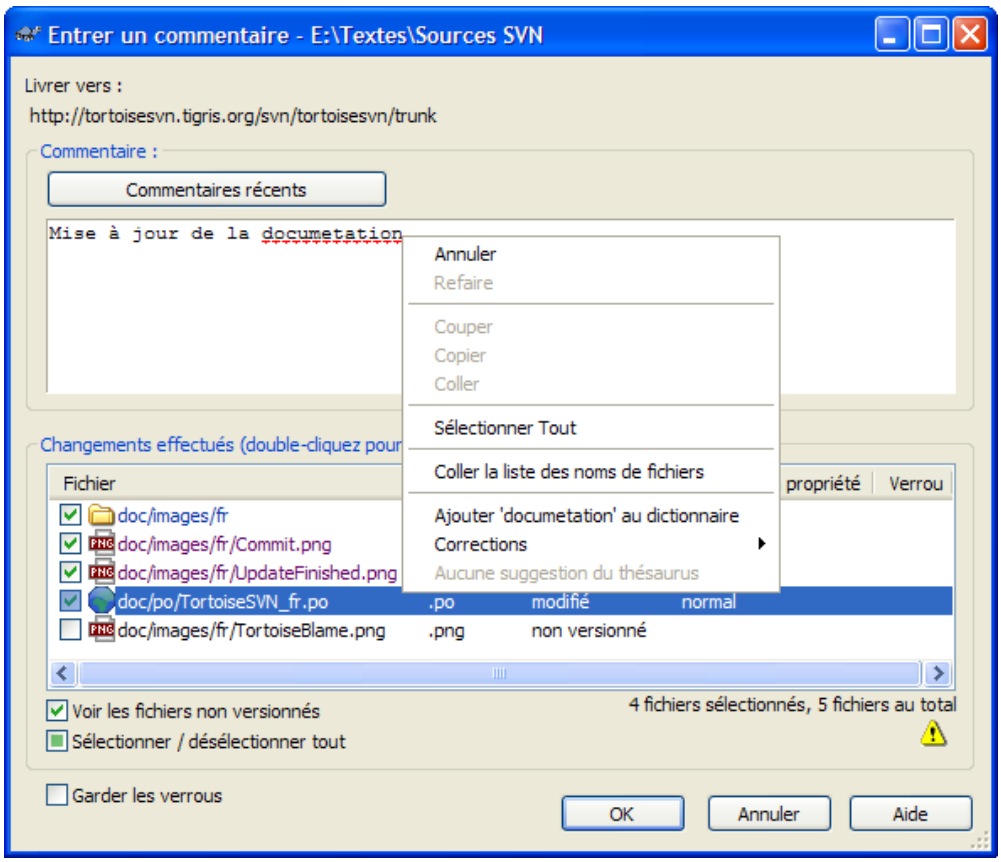


Figure 4.9. Le vérificateur d'orthographe de la boîte de dialogue Livrer

TortoiseSVN inclut un vérificateur d'orthographe pour vous aider à obtenir des commentaires corrects. Il mettra en évidence les mots mal orthographiés. Utilisez le menu contextuel pour avoir accès aux corrections suggérées. Bien sûr, il ne connaît pas *tous* les termes techniques comme vous, donc parfois les mots correctement orthographiés s'afficheront en tant qu'erreurs. Mais ne vous inquiétez pas. Vous pouvez simplement les ajouter à votre dictionnaire personnel en utilisant le menu contextuel.

La fenêtre de commentaire inclut aussi une fonctionnalité d'auto-achèvement des noms de fichier et des fonctions. Elle utilise des expressions régulières pour extraire les noms des fonctions et des classes à partir des fichiers (texte) que vous livrez, ainsi que des noms de fichier eux-mêmes. Si vous tapez un mot qui correspond à quoi que ce soit dans la liste (après avoir tapé 3 caractères ou avoir appuyé sur la touche **Ctrl+Space**), une liste déroulant apparaît vous permettant de compléter le nom. Les expressions régulières fournies avec TortoiseSVN sont contenues dans le répertoire `bin` de l'installation de TortoiseSVN. Vous pouvez aussi définir vos propres regexes et les stocker dans `%APPDATA%\TortoiseSVN\autolist.txt`. Bien sûr cette liste privée ne sera pas écrasée quand vous mettrez à jour votre installation de TortoiseSVN. Si vous n'êtes pas à l'aise avec l'utilisation des expressions régulières, jetez un coup d'oeil à l'introduction *http://en.wikipedia.org/wiki/Regular_expression* et à la documentation en ligne et au tutoriel : *http://en.wikipedia.org/wiki/Regular_expression*.

Vous pouvez réutiliser les messages de livraison des livraisons précédentes. Cliquez juste sur Messages récents pour en voir la liste. Le nombre de messages à garder en mémoire peut être modifié dans la fenêtre de propriété de TortoiseSVN.

Vous pouvez supprimer tous les messages de livraison depuis la page Données Sauvegardées de la fenêtre de Configuration de TortoiseSVN, vous pouvez également supprimer les messages précisément depuis la boite de dialogue Messages Récents en utillisant la toucje **Suppr**.

Si vous souhaitez inclure les chemins mis à jour dans votre message, vous pouvez utiliser la commande Menu Contextuel → Coller la liste des noms de fichier dans le champs texte.

Une autre façon d'insérer des chemins dans les messages informatifs est simplement de glisser/déposer les fichiers depuis la liste des fichiers dans le champs d'édition du message.

Propriétés de dossier spéciales

Il y a plusieurs propriétés de dossier spéciales qui peuvent être utilisées pour aider donner plus de contrôle sur le formatage des commentaires de livraison et la langue utilisée par le module de vérificateur d'orthographe. Lisez Section 4.17, « Configuration des projets » pour plus d'informations.

Intégration aux outils de Bug tracking

Si vous avez activé le système de bugtracking, vous pouvez indiquer un ou plusieurs bugs dans le champ ID Bug / No Incident:. Les ID doivent être séparés par des virgules. Autrement, si vous utilisez un système de bugtracking utilisant les expressions régullières, ajoutez simplement les références des bugs dans le commentaire. Pour en savoir plus : Section 4.28, « Intégration avec des systèmes de bug tracking / traqueurs d'incidents ».

4.4.5. Progression de la Livraison

Après avoir appuyé sur OK, une boîte de dialogue apparaît affichant la progression de la livraison.

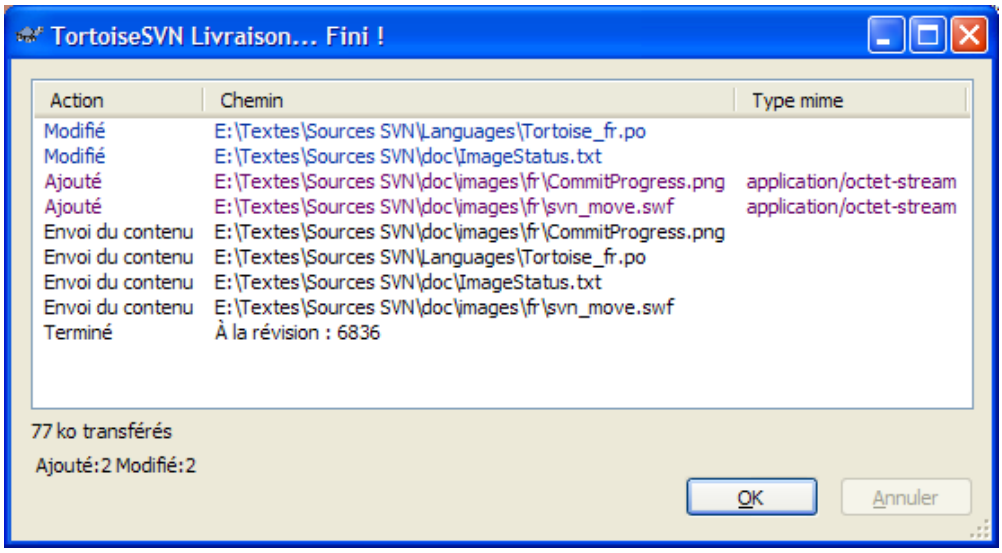


Figure 4.10. La boîte de dialogue de progression montrant une livraison en cours

La boîte de dialogue de progression utilise un code couleur pour mettre en évidence les diverses actions de la livraison

Bleu
: Livraison d'une modification.

Pourpre
: Livraison d'un nouvel ajout.

Rouge foncé
: Livraison d'une suppression ou d'un remplacement.

Noir
: Tous les autres éléments.

C'est la combinaison de couleur par défaut, mais vous pouvez personnaliser ces couleurs en utilisant la boîte de dialogue de configuration. Lisez Section 4.30.1.4, « Configuration des couleurs de TortoiseSVN » pour plus d'informations.

4.5. Mettre à jour votre copie de travail avec les changements des autres

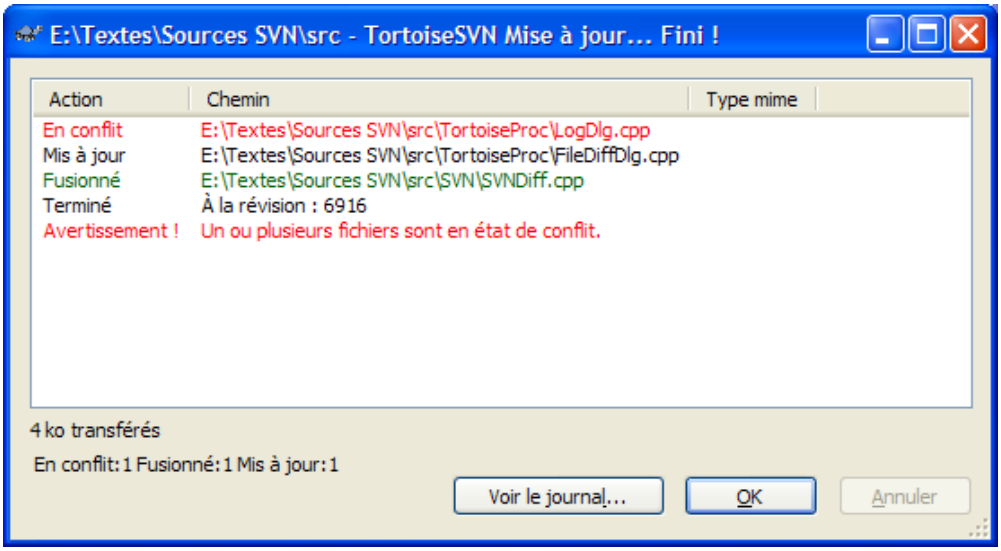


Figure 4.11. la boîte de dialogue de progression montrant une mise à jour terminée

Periodically, you should ensure that changes done by others get incorporated in your local working copy. The process of getting changes from the server to your local copy is known as *updating*. Updating may be done on single files, a set of selected files, or recursively on entire directory hierarchies. To update, select the files and/or directories you want, right click and select TortoiseSVN → Update in the explorer context menu. A window will pop up displaying the progress of the update as it runs. Changes done by others will be merged into your files, keeping any changes you may have done to the same files. The repository is *not* affected by an update.

La boîte de dialogue de progression utilise un code couleur pour mettre en évidence les diverses actions de la mise à jour

Pourpre
: Nouvel élément ajouté à votre CdT.

Rouge foncé
: Élément redondant supprimé de votre CdT, ou élément manquant remplacé dans votre CdT.

Vert
: Changements du référentiel fusionnés avec vos changements locaux avec succès.

Rouge clair
: Changements du référentiel fusionnés avec des changements locaux, aboutissant à des conflits que vous devez résoudre.

Noir
: Élément inchangé dans votre CdT mis à jour par une version plus récente du référentiel.

C'est la combinaison de couleur par défaut, mais vous pouvez personnaliser ces couleurs en utilisant la boîte de dialogue de configuration. Lisez Section 4.30.1.4, « Configuration des couleurs de TortoiseSVN » pour plus d'informations.

Si vous avez des *conflits* pendant une mise à jour (cela peut arriver si d'autres ont changé les mêmes lignes dans le même fichier que vous et que ces changements ne correspondent pas) alors la boîte de dialogue montre ces conflits en rouge. Vous pouvez double-cliquer sur ces lignes afin démarrer l'outil externe de fusion pour résoudre les conflits.

Quand la mise à jour est terminée, la boîte de dialogue de progression affiche sous la liste des fichiers un résumé du nombre d'éléments mis à jour, ajoutés, supprimés, en conflit, etc. Ce résumé d'information peut être copié dans le presse-papiers en utilisant **Ctrl+C**.

The standard Update command has no options and just updates your working copy to the HEAD revision of the repository, which is the most common use case. If you want more control over the update process, you should use TortoiseSVN → Update to Revision... instead. This allows you to update your working copy to a specific revision, not only to the most recent one. Suppose your working copy is at revision 100, but you want it to reflect the state which it had in revision 50 - then simply update to revision 50. In the same dialog you can also choose the *depth* at which to update the current folder. The terms used are described in Section 4.3.1, « Profondeur d'extraction ». The default depth is Working copy, which preserves the existing depth setting. You can also choose whether to ignore any external projects in the update (i.e. projects referenced using `svn:externals`).

Attention

If you update a file or folder to a specific revision, you should not make changes to those files. You will get « out of date » error messages when you try to commit them! If you want to undo changes to a file and start afresh from an earlier revision, you can rollback to a previous revision from the revision log dialog. Take a look at Section B.4, « Annuler des révisions dans le référentiel » for further instructions, and alternative methods.

Update to Revision can occasionally be useful to see what your project looked like at some earlier point in its history. But in general, updating individual files to an earlier revision is not a good idea as it leaves your working copy in an inconsistent state. If the file you are updating has changed name, you may even find that the file just disappears from your working copy because no file of that name existed in the earlier revision. You should also note that the item will show a normal green overlay, so it is indistinguishable from files which are up-to-date.

Si vous voulez simplement une copie locale d'une version ancienne d'un fichier, il est mieux d'utiliser la commande Menu Contextuel → Sauver la révision dans... depuis la fenêtre de commentaire de ce fichier.

Plusieurs fichiers / répertoires

Si vous choisissez plusieurs fichiers et plusieurs dossiers dans l'explorateur et sélectionnez ensuite Mettre à jour, tous ces fichiers/dossiers sont mis à jour un par un. TortoiseSVN s'assure que tous les fichiers/dossiers qui sont du même référentiel sont mis à jour à la même révision exacte ! Même si une autre livraison se produit entre ces mises à jour.

Le fichier local existe déjà

Parfois quand vous essayez de mettre à jour, la mise à jour échoue avec un message disant qu'il y a déjà un fichier local du même nom. Cela arrive typiquement quand Subversion essaye d'extraire un fichier nouvellement versionné et constate qu'un fichier non versionné du même nom existe déjà dans votre dossier de travail. Subversion n'écrasera jamais un fichier non versionné - il pourrait contenir quelque chose sur lequel vous travaillez, qui a par hasard le même nom de fichier qu'un autre développeur a utilisé pour son fichier nouvellement livré.

> Si vous obtenez ce message d'erreur, la solution est simplement de renommer le fichier local non versionné. Une fois la mise à jour terminée, vous pouvez vérifier si le fichier renommé est toujours nécessaire.
>
> Si vous continuez à obtenir des messages d'erreur, utilisez TortoiseSVN → Vérifier les modifications pour lister tous les fichiers à problème à la place. De cette façon vous pouvez tous les traiter d'un coup.

4.6. Résoudre des conflits

De temps en temps , vous aurez un *conflit* au moment de mettre à jour/fusionner vos fichiers avec le référentiel ou lorsque vous migrerez votre copie de travail vers une autre URL. Il y a deux sortes de conflits:

Conflit de fichier
: Un conflit sur un fichier arrive si deux (ou plus) développeurs changent les mêmes lignes d'un fichier.

Conflits dans l'arborescence
: Un conflit dans l'arborescence arrive quand un développeur déplace/renomme/supprime un fichier ou un dossier, qu'un autre développeur a aussi déplacé/renommé/supprimé voire juste modifié.

4.6.1. Conflit de fichiers

Un conflit de fichier arrive quand 2 développeurs our plus modifient un même fichier. Dans la mesure où Subversion ne sait rien de votre projet, il laisse la résolution des conflits à la charge des développeurs. Dès qu'un conflit est détecté, vous devriez ouvrir le fichier en question, et chercher les lignes commençant par <<<<<<<. La zone en conflit est marquée de cette manière:

```
<<<<<<< nom_de_fichier
 vos modifications
=======
 code fusionné depuis le référentiel
>>>>>>> révision
```

De plus, pour chaque conflit Subversion place trois fichiers dans votre répertoire:

nom_du_fichier.ext.mine
: C'est votre fichier comme il a existé dans votre copie de travail avant que vous mettiez à jour votre copie de travail - c'est-à-dire sans marqueurs de conflit. Ce fichier a vos derniers changements et rien d'autre.

nom_du_fichier.ext.rOLDREV
: C'est le fichier qui était la révision de BASE avant que vous mettiez à jour votre copie de travail. C'est-à-dire le fichier que vous avez extrait avant de faire votre dernière édition.

nom_du_fichier.ext.rNEWREV
: C'est le fichier que votre client de Subversion venait de recevoir du serveur quand vous avez mis à jour votre copie de travail. Ce fichier correspond à la révision de tête du référentiel.

Vous pouvez soit lancer un outil externe de fusion / un éditeur de conflit avec TortoiseSVN → Éditer les conflits soit utiliser un autre éditeur pour résoudre le conflit manuellement. Vous devrez décider à quoi devrait ressembler le code, faites les changements nécessaires et sauvegardez le fichier.

Exécutez ensuite la commande TortoiseSVN → Résolu... et livrez vos modifications au référentiel. Veuillez noter que la commande de résolution ne résout pas vraiment le conflit. Il supprime juste les

fichiers `nom_du_fichier.ext.mine` et `nom_du_fichier.ext.r*`, pour vous permettre de livrer vos changements.

Si vous avez des conflits avec des fichiers binaires, Subversion n'essaye pas de fusionner les fichiers lui-même. Le fichier local reste inchangé (exactement comme la dernière fois que vous l'avez modifié) et vous avez des fichiers `nom_du_fichier.ext.r*`. Si vous voulez renoncer à vos changements et garder la version du référentiel, utilisez simplement la commande Revenir en arrière. Si vous voulez garder votre version et écraser la version du référentiel, utilisez la commande Résolu..., livrez ensuite votre version.

Vous pouvez utiliser la commande Résolu... pour plusieurs fichiers si vous faites un clic droit sur le dossier parent et sélectionnez TortoiseSVN → Résolu... Cela affichera une boîte de dialogue listant tous les fichiers en conflit dans ce dossier et vous pouvez choisir lesquels marquer comme résolus.

4.6.2. Conflits dans l'arborescence

Un conflit dans l'arborescence est causé quand un développeur déplace/renomme/supprime un fichier ou un répertoire, qu'un autre développeur a également déplacé/renommé/supprimé ou juste modifé. Plusieurs situations peuvent générer un conflit d'arborescence, et chacune a une solution différente.

Dans Subversion, lorsque qu'un fichier est supprimé, il l'est aussi localement, de ce fait, même s'il génère un conflit d'arborescence, aucune action de résolution de conflit de ne peut être effectuée grâce au menu contextuel du click droit. Utilisez la fenêtre Vérifier les Modifications à la place pour pouvoir accéder aux options Edition des conflits.

TortoiseSVN peut aider à trouver les endroits où fusionner les modifications, mais cela ne résoudra pas tous les conflits.Souvenez vous qu'après une mise à jour la base de travail contiendra la révision des éléments tels qu'ils étaient au moment de la mise à jour. Si vous annulez une modification après avoir fait une mise à jour l'élément reviendra à la version du référentiel, pas à l'état dans lequel était l'élément avant que vous fassiez vos modifications.

4.6.2.1. Suppression locale, édition de la mise à jour

1. Le développeur A modifie `Foo.c` et en fait la livraison
2. Le développeur B a renommé simultanément `Foo.c` en `Bar.c` dans sa copie de travail, ou simplement supprimé `Foo.c` ou son répertoire parent.

Une mise à jour de la copie de travail du développeur B mène à un conflit dans l'arborescence:

- `Foo.c` a été supprimé de la copie de travail, mais est marqué comme étant en conflit d'arborescence.
- Si le conflit vient d'un renommage plus que d'une suppression alors `Bar.c` est marqué comme ajouté, mais ne contient pas les modification du développeur A.

Le développeur B doit maintenant choisir s'il veut conserver les modifications du développeur A. Dans le cas d'un renommage de fichier, il peut fusionner les modifications de `Foo.c` dans le fichier renommé `Bar.c`. Pour des suppressions de fichier ou de répertoire il peut choisir de garder l'élément avec les modifications du développeur A et annuler la suppression. Ou, en ne faisant que marquer le conflit comme étant résolu, il annule les modifications du développeur A.

La fenêtre d'édition de conflits offre la possibilité de fusionner les changements si elle peut trouver le fichier original de celui renommé en `Bar.c`. Dépendant d'où la mise à jour à été invoqué, il n'est pas toujours possible de trouver le fichier source.

4.6.2.2. Modification locale, suppression dans la mise à jour

1. Le développeur A renomme `Foo.c` en `Bar.c` et envoie les modifications au dépôt.
2. Le développeur B modifie `Foo.c` dans sa copie de travail.

Ou dans le cas d'un déplacement de répertoire ...

1. Le développeur A déplace le réperetoire parent `FooFolder` vers `BarFolder` et soumet celui-ci au dépôt.

2. Le développeur B modifie `Foo.c` dans sa copie de travail.

Une mise à jour de la copie de travail du développeur B mène à un conflit dans l'arborescence. Pour un simple conflit de fichier :

- `Bar.c` est ajouté dans la copie de travail avec le statut 'normal'.

- `Foo.c` est marqué comme ayant été ajouté (avec historique) et étant en conflit dans l'arborescence.

Pour un conflit de répertoire:

- `Bar.c` est ajouté dans la copie de travail avec le statut 'normal'.

- `Foo.c` est marqué comme ayant été ajouté (avec historique) et étant en conflit dans l'arborescence.

  `Foo.c` est marqué comme modifié.

Le développeur B doit à présent décider de garder la réorganisation du développeur A et de fusionner ses changement dans le fichier correspondant dans la nouvelle architecture, ou simplement annuler les modifications du développeur A et garder le fichier local.

Pour fusionner ses modifications locales avec la réorganisation, le développeur B doit d'abord trouver quel est le nouveau nom du fichier`Foo.c` dans le référentiel. Cela peut être fait en utilisant la fenêtre de commentaires. A présent les modifications doivent être réintégrées à la main dans le mesure où il n'y a aucun outil existant pour simplifier ce processus. Dès que les modifications ont été répercutées, le chemin conflictuel est redondant et peut donc être supprimé. Dans ce cas utilisez le bouton Supprimer dans la fenêtre de résolution de conflits pour faire le nettoyage et marquer le conflit comme résolu.

Si le développeur B décide que les modifications du développeur A étaient mauvaises alors elle doit choisir le bouton Garder dans la fenêtre de résolution des conflits. Cette action a pour effet de marquer le conflit du fichier/répertoire comme résolu, mais les modifications du développeur A doivent être retirées à la main. De nouveau la fenêtre de commentaires aide à trouver ce qui a été déplacé.

4.6.2.3. Suppression locale, suppression dans la mise à jour

1. Le développeur A renomme `Foo.c` en `Bar.c` et envoie les modifications au dépôt

2. Le développeur B renomme `Foo.c` en `Bix.c`

Une mise à jour de la copie de travail du développeur B mène à un conflit dans l'arborescence:

- `Bix.c` est marqué comme ajouté avec l'historique.

- `Bar.c` est ajouté dans la copie de travail avec le statut 'normal'.

- `Foo.c` est marqué comme étant supprimé et étant en conflit dans l'arborescence.

Pour résoudre ce conflit, le développeur B doit trouver à quel fichier renommé/déplacé correspond la version conflictueuse de `Foo.c` dans le référentiel. Ce peut être fait en utilisant la fenêtre de log.

Le développeur B doit alors décider quel nouveau nom de `Foo.c`garder - celui du développeur A ou celui issu du renommage qu'il a lui même effectué.

Dès que le développeur B a résolu manuellement le conflit, l'arborescence des conflits doit être marquée comme résolue grâce au bouton approprié dans la fenêtre d'édition des conflits.

4.6.2.4. Copie locale incomplète, modification dans la mise à jour

1. Le développeur A travaillant sur le tronc modifie `Foo.c` et en fait la livraison au dépôt.

2. Le développeur B travaillant sur une branche renomme `Foo.c` en `Bar.c` et en fait la livraison au dépôt

Une fusion des modifications du tronc du développeur A dans la branche de copie de travail du développeur B conduit à un conflit dans l'arborescence:

- `Bar.c` est déjà dans la copie de travail avec le statut 'normal'.
- `Foo.c` est marqué comme manquant dans l'arborescence des conflits.

Pour résoudre ce conflit, les développeurs B doit marquer le fichier comme résolu dans la fenêtre de résolution de conflits, lequel sera retiré de la liste des conflits. Le développeur B doit alors décider s'il faut copier le fichier manquant `Foo.c` depuis le référentiel, ou fusionner `Foo.c` avec le fichier renommé `Bar.c` ou simplement s'il faut ignorer les changements en marquant le conflit comme étant résolu et ne rien faire d'autre.

Notez que si vous copiez le fichier manquant depuis le référentiel et que vous le marquez comme résolu, votre copie sera de nouveau retirée. Vous devez résoudre d'abord le conflit.

4.6.2.5. Modification locale, suppression dans la fusion

1. Le développeur A travaillant sur le tronc renomme `Foo.c` en `Bar.c` et l'envoie au dépôt
2. Le développeur B travaillant sur une branche modifie le fichier `Foo.c` et l'envoie au dépôt.

Il y a des cas équivalents pour les déplacements de répertoire, mais ce n'est pas encore détecté dans Subversion 1.6 ...

1. Le développeur A travaillant sur le tronc déplace le dossier parent `FooFolder` vers `BarFolder` et l'envoie au dépôt.
2. Le développeur B travaillant sur une branche modifie `Foo.c` dans sa copie de travail.

Une fusion des modifications du tronc du développeur A dans la branche de copie de travail du développeur B conduit à un conflit dans l'arborescence:

- `Bar.c` est marqué comme étant ajouté.
- `Foo.c` est marqué comme ayant été modifié et étant en conflit dans l'arborescence.

Le développeur B doit à présent décider de garder la réorganisation du développeur A et de fusionner ses changement dans le fichier correspondant dans la nouvelle architecture, ou simplement annuler les modifications du développeur A et garder le fichier local.

Pour fusionner ses modifications locales avec les autre remaniements, le développeur B doit trouver à quel fichier renommé/déplacé correspond la version conflictueuse de `Foo.c` dans le dépôt. Cela peut être fait en utilisant la fenêtre de commentaires du fichier source. L'éditeur de conflit ne montre que les commentaires relatifs à la copie de travail puisqu'il ne sait pas quel chemin a été utilisé lors de la fusion, donc vous devrez le découvrir vous même. Les modifications doivent alors être fusionnées à la main puisqu'il n'existe actuellement aucun moyen d'automatiser ou de simplifier ce processus. Une fois que les changement ont été effectués, le chemin conflictueux est superflu et peut être supprimé. Dans de cas, utilisez le bouton Supprimer dans la fenêtre d'édition de conflits pour marquer le conflit comme étant résolu.

Si le développeur B décide que les modifications du développeur A étaient mauvaises elle doit alors choisir le bouton Garder dans la fenêtre de résolution de conflit. Cette action marque le conflit du fichier/répertoire comme étant résolu, mais le développeur A doit reporter ses modifications à la main. La fenêtre de commentaires du fichier source de la fusion aide à détecter ce qui a été déplacé.

4.6.2.6. Suppression locale, suppression dans la fusion

1. Le développeur A travaillant sur le tronc renomme `Foo.c` en `Bar.c` et l'envoie au dépôt

2. Le développeur B travaillant sur une branche renomme `Foo.c` en `Bix.c` et envoie les modifications au dépôt

Une fusion des modifications du tronc du développeur A dans la branche de copie de travail du développeur B conduit à un conflit dans l'arborescence:

- `Bix.c` est marqué comme étant dans un état normal (non modifé).
- `Bar.c` est marqué comme ajouté avec son historique.
- `Foo.c` est marqué comme manquant et générant une erreur dans l'arborescence.

Pour résoudre ce conflit, le développeur B doit trouver quel fichier renommé/déplacé correspond à la version conflictueuse de `Foo.c` dans le référentiel. Ce peut être fait en utilisant la fenêtre de commentaires pour les sources fusionnées. L'éditeur de conflit ne montre que les commentaires relatifs à la copie de travail puisqu'il ne sait pas quel chemin a été utilisé lors de la fusion, donc vous devrez le découvrir vous même.

Le développeur B doit alors décider quel nouveau nom de `Foo.c`garder - celui du développeur A ou celui issu du renommage qu'il a lui même effectué.

Dès que le développeur B a résolu manuellement le conflit, l'arborescence des conflits doit être marquée comme résolue grâce au bouton approprié dans la fenêtre d'édition des conflits.

4.7. Obtenir des information sur le statut

Pendant que vous travaillez sur votre copie de travail, vous avez souvent besoin de savoir quels fichiers vous avez changé/ajouté/supprimé ou renommé, ou même quels fichiers ont été changé et livré par les autres.

4.7.1. Recouvrement d'icônes

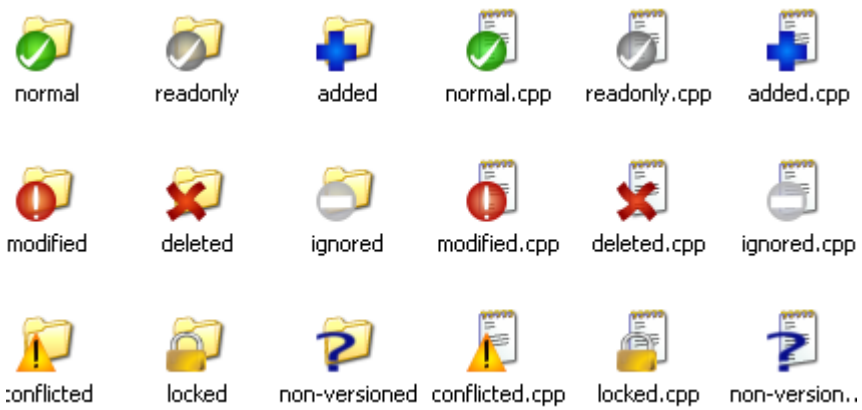


Figure 4.12. L'Explorateur montrant le recouvrement d'icônes

Maintenant que vous avez extrait une copie de travail à partir d'un référentiel Subversion, vous pouvez voir que vos fichiers ont d'autres icones dans l'explorateur Windows. C'est une des raisons pour laquelle TortoiseSVN est si populaire. TortoiseSVN ajoute une icône de recouvrement à chaque icône de fichier chevauchant l'icône de fichier original. Selon le statut Subversion du fichier, l'icône de recouvrement est différente.

Une copie de travail fraîchement extraite a une coche verte comme icône de recouvrement. Cela signifie que le statut Subversion est `normal`.

Dès que vous commencez à éditer un fichier, le statut change en `modifié` et l'icône de recouvrement devient alors un point d'exclamation rouge. De cette façon, vous pouvez facilement voir quels fichiers ont été modifiés depuis votre dernière mise à jour de la copie de travail et ont donc besoin d'être livrés.

Si un `conflit` se produit lors d'une mise à jour alors l'icône de recouvrement devient un point d'exclamation jaune.

Si vous avez mis la propriété `svn:needs-lock` sur un fichier, Subversion met ce fichier en lecture seule jusqu'à ce que vous verrouilliez ce fichier. Ces fichiers ont cette icône de recouvrement pour indiquer que vous devez d'abord les verrouiller pour pouvoir les éditer.

Si vous avez verrouillé un fichier et que le statut de Subversion est `normal`, cette icône de recouvrement vous rappelle que vous devriez relâcher le verrou si vous ne l'utilisez pas pour permettre aux autres utilisateurs de livrer leurs changements.

Cette icône vous montre que quelques fichiers ou dossiers à l'intérieur du dossier ont été marqués comme *supprimés* du contrôle de version ou qu'un fichier est manquant.

Le signe plus vous indique qu'un fichier ou un dossier a été marqué comme étant *ajouté* au contrôle de version.

Le signe "sens interdit" vous indique qu'un fichier ou un dossier a été marqué comme étant à `ignorer` par le contrôle de version. Cette indication est optionnelle.

Cette icone est associée aux éléments qui ne sont pas sous contrôle de version, et qui ne sont pas à ignorer. Cette icone de recouvrement est facultative.

En fait, vous pouvez constater que ces icônes ne sont pas toutes utilisées sur votre système. C'est dû au fait que le nombre de recouvrements permis par Windows est très limité et si vous utilisez aussi TortoiseCVS, alors il n'y a pas assez d'emplacements de recouvrement disponibles. TortoiseSVN essaie d'être un « Bon Citoyen (TM) » et limite son utilisation des recouvrements pour laisser de la place aux autres applications.

Depuis que différent clients Tortoise sont disponibles (TortoiseCVS, TortoiseHG, ...) ces limites deviennent un réel problème. Pour contourner ce souci, le projet TortoiseSVN met à présent à votre disposition une série d'icones dans une DLL, qui peuvent être utilisées par tous les clients Tortoise. Vérifiez avec votre fournisseur pour savoir si cette fonctionnalité a été intégrée :-)

Pour avoir une description des icônes de recouvrement correspondant aux statuts de Subversion et d'autres détails techniques, lisez Section F.1, « Recouvrement d'icônes ».

4.7.2. Les colonnes de TortoiseSVN dans l'explorateur Windows

Les mêmes informations, disponibles avec le recouvrement d'icônes, (et plus encore) peuvent être affichées avec des colonnes supplémentaires dans l'affichage en détail de l'explorateur Windows.

Faites simplement un clic droit sur un des titres d'une colonne, choisissez Autres... dans le menu contextuel. Une boîte de dialogue apparaîtra dans laquelle vous pourrez spécifier les colonnes à afficher dans la « Vue détaillée » ainsi que leur ordre. Faites défiler pour que s'affichent les entrées commençant par SVN. Cochez celles vous voudriez afficher et fermez la boîte de dialogue en appuyant sur OK. Les colonnes seront ajoutées à droite de celles déjà affichées. Vous pouvez les réordonner par glisser-déposer, ou les redimensionner, pour qu'elles correspondent à vos besoins.

Important

Les colonnes supplémentaires dans l'Explorateur Windows ne sont pas disponibles sous Vista, Microsoft n'autorisant plus ce type de colonnes pour *tous* les fichiers, mais plus que pour certains types de fichiers.

Astuce

Si vous voulez que la disposition actuelle soit affichée dans toutes vos copies de travail, vous pouvez en faire la vue par défaut.

4.7.3. Statut local et distant

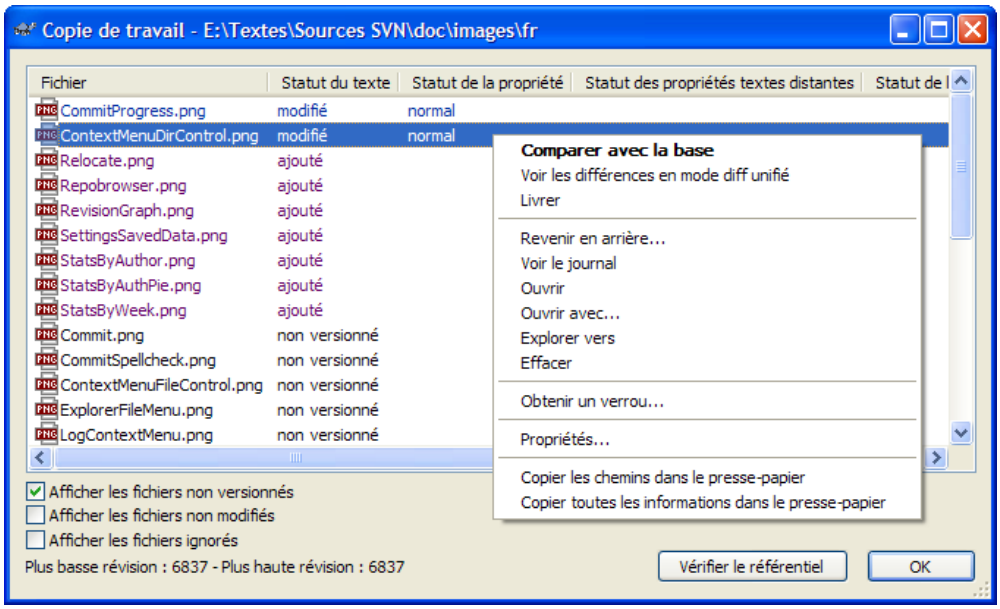


Figure 4.13. Vérifier les modifications

Il est souvent très utile de savoir quels fichiers vous avez changé et aussi quels fichiers ont été changés et livrés par les autres. C'est là où la commande TortoiseSVN → Vérifier les modifications... devient pratique. Cette boîte de dialogue vous montrera les fichiers ayant été modifié dans votre copie de travail, ainsi que les fichiers non versionnés que vous pouvez avoir.

Si vous cliquez sur Vérifier le référentiel alors vous pouvez aussi chercher les changements dans le référentiel. De cette manière vous pouvez vérifier avant une mise à jour s'il y a un conflit potentiel. Vous pouvez aussi mettre à jour certains fichiers du référentiel sans mettre à jour le dossier entier. Par défaut, le bouton Vérifier le référentiel ne récupère que le statut du référentiel à la profondeur de la copie de travail.Si vous voulez voir tous les fichiers et répertoires du référentiel, même ceux qui n'ont pas été extraits, vous devez cliquez sur le bouton Vérifier le référentiel en appuyant sur la touche **Shift**.

La boîte de dialogue utilise un code couleur pour mettre le statut en évidence .

Bleu
: Éléments modifiés localement

Pourpre
: Éléments ajoutés. Les éléments qui ont été ajoutés avec un historique ont un signe + dans la colonne Statut du texte et une info-bulle montre d'où l'article a été copié.

Rouge foncé
: Éléments supprimés ou manquants.

Vert
: Éléments modifiés localement et dans le référentiel. Les changements seront fusionnés lors de la mise à jour. Cela *peut* produire des conflits à la mise à jour.

Rouge clair
: Éléments modifiés localement et supprimés dans le référentiel, ou modifiés dans le référentiel et supprimés localement. Cela *va* produire des conflits à la mise à jour.

Noir
: Éléments inchangés et non versionnés.

C'est la combinaison de couleur par défaut, mais vous pouvez personnaliser ces couleurs en utilisant la boîte de dialogue de configuration. Lisez Section 4.30.1.4, « Configuration des couleurs de TortoiseSVN » pour plus d'informations.

Les éléments qui ont été commutés vers un chemin de référentiel différent sont aussi indiqués en utilisant un marqueur `(s)`. Vous pouvez avoir commuté quelque chose en travaillant sur une branche et avoir oublié de rebasculer sur le tronc. C'est votre signal d'alarme !

À partir du menu contextuel de la boîte de dialogue vous pouvez afficher une comparaison des changements. Vérifiez les changements locaux que *vous* avez fait en utilisant Menu contextuel → Comparer avec la Base. Vérifiez les changements du référentiel faits par les autres en utilisant Menu contextuel → Voir les différences en mode diff unifié.

Vous pouvez aussi annuler des changements dans des fichiers individuels. Si vous avez supprimé un fichier accidentellement, il apparaitra comme *Manquant* et vous pourrez utiliser *Revenir en arrière* pour le récupérer.

Les fichiers non versionnés et les fichiers ignorés peuvent être envoyés dans la corbeille à partir d'ici en utilisant Menu contextuel → Supprimer. Si vous voulez supprimer définitivement les fichiers (sans passer par la corbeille) maintenez la touche **Maj** pendant que vous cliquez sur Supprimer.

Si vous voulez examiner un fichier en détail, vous pouvez le glisser d'ici vers une autre application comme un éditeur de texte ou un IDE.

Les colonnes affichées sont personnalisables. Si vous faites un clic droit sur n'importe quel en-tête de colonne un menu contextuel vous permettant de choisir les colonnes à afficher apparaitra. Vous pouvez aussi changer la largeur de colonne en faisant glisser le bord des entêtes. Ces personnalisations sont préservées, donc vous verrez les mêmes en-tête la prochaine fois.

Si vous travaillez sur plusieurs tâches distinctes en même temps, vous pouvez regrouper les fichiers dans des listes de modification. Lisez Section 4.4.2, « Listes de changements » pour plus d'informations.

En bas de la boîte de dialogue vous pouvez apercevoir un sommaire de l'éventail des révisions en mémoire utilisées dans votre copie active. Celles-ci sont du type *Livrer*, et non de type *mise à jour*; elles représentent l'éventail de révisions où ces fichiers ont été livrés dernièrement, et non les révisions vers lesquelles elles ont été mises à jour. Notez que l'éventail de révisions présentées s'applique uniquement

aux éléments affichés, et non à la copie active entière. Si vous désirez voir cette information pour la copie active dans son intégralité vous devez cocher la case Montrer les éléments non-modifiés.

Astuce

Si vous voulez une vue plate de votre copie de travail, c'est-à-dire qui montre tous les fichiers et les dossiers à chaque niveau de la hiérarchie des dossiers, alors la boîte de dialogue Vérifier les modifications est la façon la plus facile d'y arriver. Cochez seulement la case Afficher les fichiers non modifiés pour afficher tous les fichiers de votre copie de travail.

Réparation des renommages externes

Parfois, les fichiers sont renommés en dehors de Subversion, et son donc affichés comme fichiers manquants et non versionnés. Pour éviter de perdre l'historique vous devez indiquer à Subversion la connexion. Sélectionnez le nom manquant (nom précédent) et le nouveau et utilisez Menu Contextuel → Réparer le Déplacement pour associer les deux fichiers au renommage.

4.7.4. Voir les différences

Souvent vous voulez regarder à l'intérieur de vos fichiers, pour regarder ce que vous avez changé. Vous pouvez accomplir cela en sélectionnant un fichier qui a changé et en choisissant Voir les différences à partir du menu contextuel de TortoiseSVN. Cela démarre le visualisateur externe de différence , qui comparera alors le fichier actuel avec la copie primitive (la révision BASE), qui a été stockée après la dernière extraction ou la dernière mise à jour.

Astuce

Même lorsque vous ne vous trouvez pas dans une copie de travail ou quand vous avez de multiples versions du fichier ici et là, vous pouvez toujours afficher les différences :

Sélectionnez les deux fichiers que vous voulez comparer dans l'explorateur (en utilisant par exemple **Ctrl** et la souris) et sélectionnez Voir les différences à partir du menu de contextuel de TortoiseSVN. Le fichier cliqué en dernier (celui avec le focus, c'est-à-dire le rectangle pointillé) sera considéré comme le plus récent.

4.8. Listes de changements

Dans un monde idéal, vous ne travaillez jamais que sur une seule chose à la fois, et votre copie active ne contient qu'un seul jeu de changements logiques. OK, retour à la réalité. Il arrive souvent que vous ayez à travailler sur plusieurs tâches à la fois, sans rapport les unes avec les autres, et quand vous regardez dans la boîte de dialogue Livrer, toutes les modifications sont mélangées ensemble. L'élément *changelist* vous aide à regrouper les fichiers, ce qui vous aide à voir plus facilement ce que vous faites. Bien sûr, ceci ne peut fonctionner que si les modifications ne se chevauchent pas. Si deux tâches différentes affectent le même fichier, il devient impossible de séparer les modifications.

Important

La fonctionnalité de TortoiseSVN donnant la liste des changements n'est disponible qu'à partir de Windows XP, dans la mesure où elle dépend d'une fonctionnalité du shell qui n'est pas disponible dans Windows 2000. Désolé mais Win2K est vraiment vieux maintenant, donc ne vous plaignez pas.

Vous pouvez voir la liste des modifications à différents endroits, mais les plus importants sont la fenêtre de livraison et la fenêtre de vérification des modifications. Commençons par la fenêtre de vérification des modifications après avoir travaillé sur différentes fonctionnalités dans beaucoup de fichiers. Lorsque vous ouvrez la fenêtre pour la première fois, tous les fichiers modifiés sont listés. Supposons maintenant que vous vouliez tout organiser et grouper ces fichiers par fonctionnalité.

Sélectionnez un ou plusieurs fichiers et utilisez MenuContextuel → Déplacer vers la liste des modifications pour ajouter un élément à une liste de modifications. Initialement il n'y aura pas de liste de modifications, donc la première fois que vous ferez ceci vous devrez créer une nouvelle liste de modifications. Donnez-lui un nom qui décrit ce pour quoi vous l'utilisez, et cliquez sur OK. La boîte de dialogue va maintenant vous montrer des groupes d'éléments.

Dès que vous avez créé une liste de modification vous pouvez y glisser/déposer des éléments depuis une autre liste de modifications, ou depuis Windows Explorer. L'avantage de le faire depuis Explorer est de permettre d'ajouter des éléments avant qu'ils ne soient modifiés. Vous pourriez faire de même depuis la fenêtre de vérification des modifications, mais seulement en affichant tous les fichiers non modifiés.

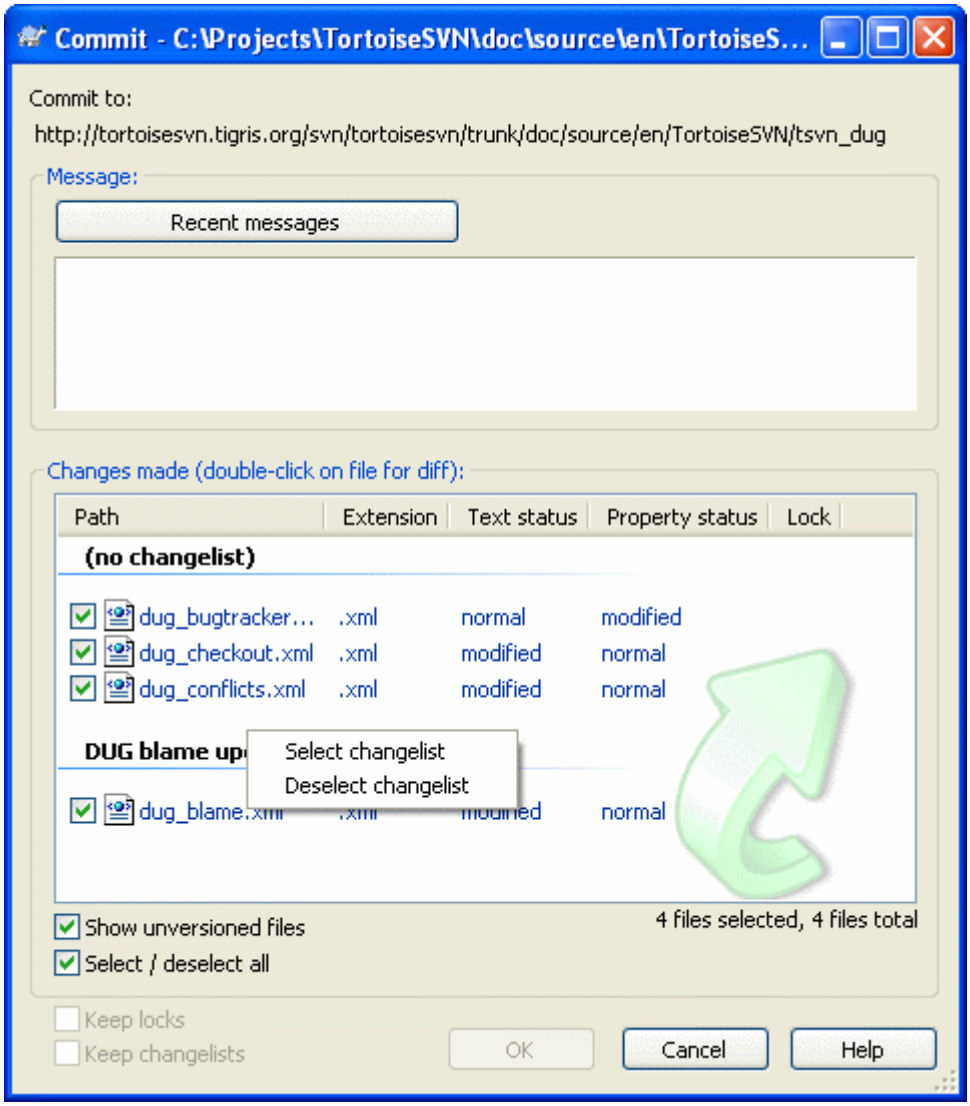


Figure 4.14. Fenêtre de livraison avec les listes de modification

Dans la fenêtre de livraison vous pouvez voir ces mêmes fichiers, regroupés par liste de modifications. En dehors de donner directement une indication sur les regroupements, vous pouvez également vous servir des entêtes des groupes pour sélectionner les fichiers à livrer.

Sous XP, un menu contextuel apparaît lorsque vous faites un clic droit sur l'entête d'un groupe, vous donnant le choix de cocher ou de décocher toutes les entrées. Sous Vista le menu contextuel n'est pas nécessaire. Cliquez sur l'entête du groupe pour tout sélectionner, puis cochez une des entrées sélectionnées pour toutes les cocher.

TortoiseSVN se réserve une liste de modification, appelée `ignore-on-commit`. Elle est utilisée pour marquer les fichiers versionnés que vous souhaitez rarement livrer même si vous les avez modifiés. Cette fonctionnalité est décrite dans Section 4.4.3, « Exclude des éléments de la livraison ».

Lorsque vous livrez des fichiers faisant partie d'une liste de modifications alors vous vous attendez naturellement à ce qu'ils n'en fassent plus partie ensuite. Donc par défaut, les fichiers sont retirés automatiquement de la liste des modifications après avoir été livrés. Si ce comportement ne vous convient pas, utilisez la case à cocher Garder les listes de modifications en bas de la fenêtre de livraison.

Astuce

Les listes de modifications sont une fonctionnalité du client local. Créer et supprimer ces listes n'influencera en rien ni le référentiel, ni les copies de travail des autres. C'est juste un moyen pratique d'organiser vos fichiers.

4.9. La boîte de dialogue du Journal de révision

A chaque modification que vous faites et que vous livrez, vous devriez fournir un commentaire décrivant cette modification. De cette façon, vous pouvez par la suite comprendre quelles modifications vous avez faites et pourquoi, et vous disposez d'un journal détaillé pour votre processus de développement.

La boîte de dialogue du Journal de révision récupère tous ces commentaires de révision et vous les présente. L'affichage est divisé en 3 panneaux.

- Le panneau supérieur présente une liste de révisions où des modifications au fichier/dossier ont été livrées. Ce résumé inclut la date et l'heure, la personne qui a livré la révision et le début du commentaire.

  Les lignes affichées en bleu indiquent que quelque chose a été copié vers cette ligne de développement (peut-être d'une branche).

- Le panneau du milieu montre le commentaire en entier pour la révision choisie.

- Le panneau du bas montre une liste de tous les fichiers et de tous les dossiers qui ont été modifiés lors de la révision sélectionnée.

Mais elle fait beaucoup plus que cela : elle fournit des commandes de menu contextuel que vous pouvez utiliser pour obtenir encore plus d'informations sur l'historique du projet.

4.9.1. Appeler la boîte de dialogue du Journal de révision

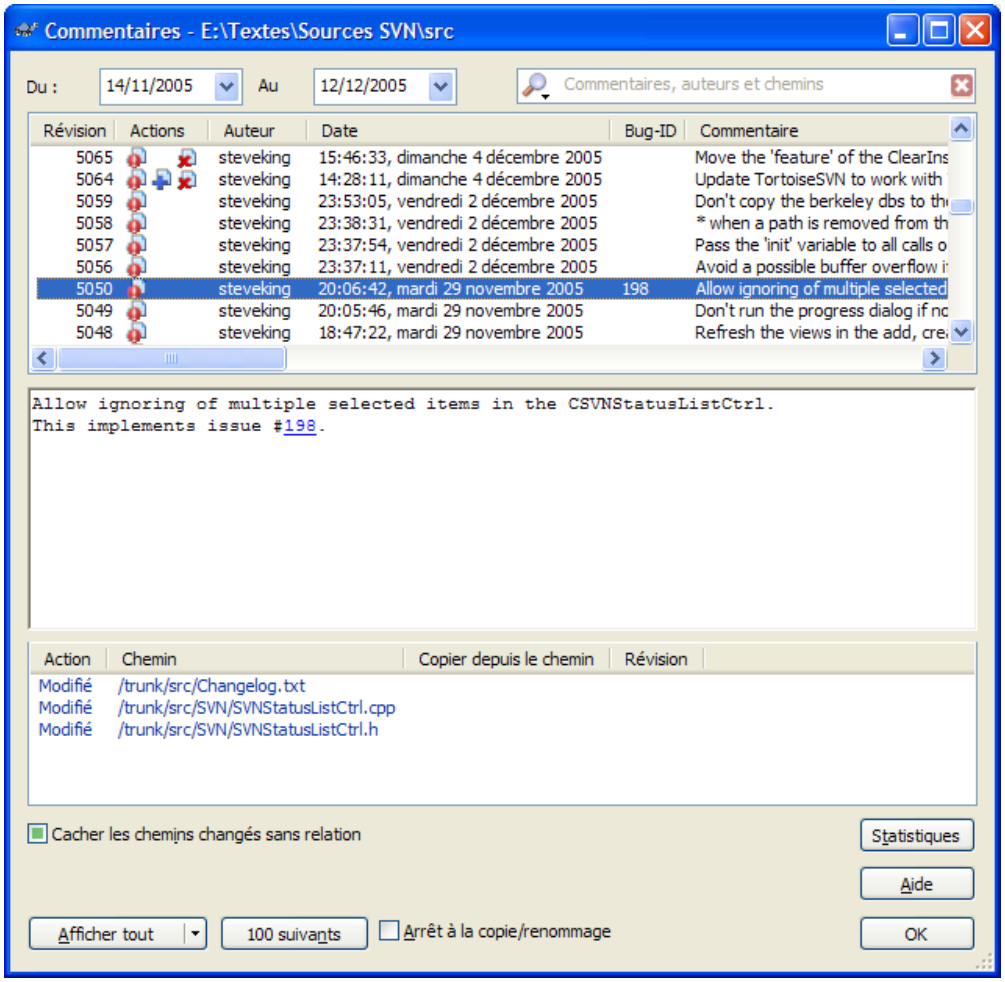


Figure 4.15. La boîte de dialogue du Journal de révision

Il existe plusieurs endroits à partir desquels vous pouvez afficher la boîte de dialogue de Journal :

- À partir du sous-menu contextuel de TortoiseSVN
- À partir de la page de propriété
- À partir de la boîte de dialogue de progression après qu'une mise à jour a fini. Alors la boîte de dialogue de Journal ne montre que les révisions qui ont été modifiées depuis votre dernière mise à jour

Si le référentiel n'est pas joignable vous verrez une fenêtre Travailler hors ligne ?, décrite dans Section 4.9.10, « Mode hors ligne ».

4.9.2. La boîte de dialogue du Journal de révision

Le panneau supérieur a une colonne Actions contenant des icones résumant ce qui a été fait dans cette révision. Il y a quatre icones différents, chacun est affiché dans sa propre colonne.

Si une révision modifie un élément, l'icône *modifié* est affichée dans la première colonne.

Si un élément a été ajouté lors d'une révision, l'icône *ajouté* est affichée dans la seconde colonne.

Si un élément a été supprimé dans une révision, l'icône *deleted* s'affiche dans la troisième colonne.

Si un élément a été remplacé, l'icône *remplacé* est affichée dans la quatrième colonne.

4.9.3. Obtenir des informations supplémentaires

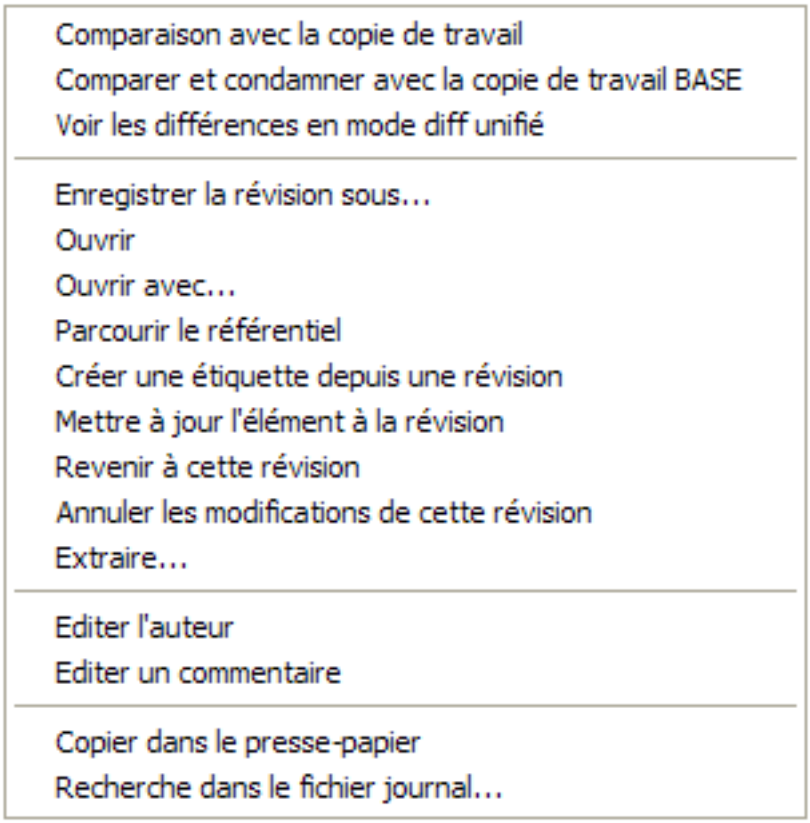


Figure 4.16. Le panneau supérieur de la boîte de dialogue du Journal de révision avec le menu contextuel

De plus amples informations sont disponibles via un menu contextuel dans la panneau supérieur de la boîte de dialogue du Journal de révision. Certains éléments de ce menu sont spécifiques aux fichiers, d'autres aux dossiers.

Comparer avec la copie de travail
: Comparer la révision sélectionnée avec votre copie de travail. L'outil de différenciation par défaut est TortoiseMerge qui est fourni avec TortoiseSVN. Si la boîte de dialogue de journal concerne un dossier, cela vous montrera une liste de fichiers modifiés et vous permettra de passer en revue les modifications apportées à chaque fichier individuellement.

Comparer et annoter avec la BASE de travail
: Annoter la révision sélectionnée et le fichier dans votre base de travail et comparer les résultats en utilisant un outil de différenciation. Lire Section 4.23.2, « Annoter les différences » pour plus de détails (uniquement pour les fichiers).

Voir les modifications comme des différences unifiées
: Voir les modifications faites dans la révision sélectionnée en tant que fichier de différences unifiées (format de patch GNU). Cela montre seulement les différences avec quelques lignes de contexte.

Cela est plus difficile à lire qu'une comparaison visuelle de fichiers, mais cela vous présentera tous les changements dans un format compact.

Comparer avec la révision précédente
: Compare the selected revision with the previous revision. This works in a similar manner to comparing with your working copy. For folders this option will first show the changed files dialog allowing you to select files to compare.

Comparer avec la révision précédente et annoter
: Show the changed files dialog allowing you to select files. Blame the selected revision, and the previous revision, and compare the results using a visual diff tool. (folders only).

Sauvegarder la révision dans...
: Sauvegarder la révision sélectionnée dans un fichier pour avoir une version antérieure de ce fichier (valable uniquement pour les fichiers).

Ouvrir / Ouvrir avec...
: Ouvrir le fichier sélectionné, avec l'application par défaut pour ce type de fichier, ou avec le programme de votre choix (valable uniquement pour les fichiers).

Annoter...
: Annoter le fichier jusqu'à la révision sélectionnée (uniquement pour les fichiers).

Parcourir le référentiel
: Ouvrir l'explorateur de référentiel pour examiner le fichier ou le dossier sélectionné dans le référentiel tel qu'il était à la révision sélectionnée.

Créer une branche/une étiquette depuis la révision
: Créer une branche ou une étiquette à partir d'une révision choisie. C'est utile par exemple si vous avez oublié de créer une étiquette et avez déjà livré des modifications qui n'étaient pas supposées être intégrées à cette révision.

Mettre à jour l'élément à la révision
: Mettre à jour votre copie de travail à la révision sélectionnée. Utile si vous voulez que votre copie de travail soit le reflet d'un état précis du passé ou s'il y a eu depuis d'autres livraisons dans le référentiel et que vous voulez mettre à jour votre copie de travail par étapes successives. Il est préférable de mettre à jour un répertoire entier de votre copie de travail, et pas seulement un fichier, autrement votre copie de travail pourrait être incohérente.

  Si vous voulez annuler définitivement une modification, utilisez Revenir à cette révision.

Revenir à cette révision
: Revenir à une révision antérieure. Si vous avez fait plusieurs modifications et décidez ensuite que vous voulez vraiment remettre les choses telles qu'elles étaient à la révision N, c'est la commande dont vous avez besoin. Les modifications sont annulées dans votre copie de travail : cette opération n'affecte donc *pas* le référentiel tant que vous n'avez pas livré ces changements. Notez que cela annulera *toutes* les modifications faites après la révision sélectionnée, remplaçant le fichier/dossier avec cette révision antérieure.

  Si votre copie de travail n'a pas été modifiée, après avoir fait cette action elle sera marquée comme étant modifiée. Si vous avez déjà des modifications locales, cette commande fusionnera les modifications *annulées* dans votre copie de travail.

  En interne, Subversion annule la fusion de toutes les modifications effectuées après la révision sélectionnée, revenant ainsi à l'état précédant ces livraisons.

  Après avoir effectué cette action, si vous voulez *annuler l'annulation* et retrouver votre copie active dans son état précédent non-modifié, nous vous conseillons d'utiliser TortoiseSVN → Revert à

partir de Windows Explorer, ce qui aura pour effet de rejeter les modifications locales effectuées par cette action de dé-fusion.

Si vous voulez juste avoir un aperçu d'un fichier ou d'un répertoire dans une révision antérieure, utilisez Revenir à cette révision ou Sauvegarder la révision sous....

Annuler les modification pour revenir à cette révision
: Annuler les modifications effectuées à la révision sélectionnée. Les modifications sont annulées dans votre copie de travail donc cette opération n'affecte *pas* du tout le référentiel ! Notez que cela annulera les modifications de cette révision seulement. Cela ne remplace pas votre copie de travail par le fichier complet de la révision précédente. C'est très utile pour annuler une modification antérieure quand d'autres modifications sans rapport ont été faites depuis.

Si votre copie de travail n'a pas été modifiée, après avoir fait cette action elle sera marquée comme étant modifiée. Si vous avez déjà des modifications locales, cette commande fusionnera les modifications *annulées* dans votre copie de travail.

En interne, Subversion annule la fusion de cette révision particulière, annulant l'effet d'une Livraison antérieure.

Vous pouvez *annuler une annulation* comme décrit ci dessus dans Revenir à cette révision.

Fusionner la révision avec...
: Fusionner la (les) révision(s) sélectionnée(s) en une copie active séparée. Une boîte de dialogue de sélection de dossiers vous permet de choisir la copie active qui sera la cible de la fusion, mais attention : il n'y a par la suite aucune demande de confirmation ni aucune possibilité de faire une fusion de test. Il est judicieux de faire la fusion dans une copie active intacte afin de pouvoir annuler les modifications si cela ne fonctionne pas! Cette fonctionnalité se révèle très utile si vous désirez fusionner des révisions choisies d'une branche à une autre.

Extraire...
: Fait une extraction propre de la révision sélectionnée. Cette fonctionnalité déclenche l'affichage d'une fenêtre permettant de confirmer l'URL, le numéro de la révision, et de sélectionner l'emplacement de l'extraction.

Exporter...
: Exporter le fichier/répertoire sélectionné à la révision sélectionnée. Une fenêtre vous demande alors de confirmer l'URL, la révision, et de sélectionner le répertoire où faire exportation.

Modifier l'auteur/le commentaire de révision
: Éditer le commentaire de révision ou l'auteur attaché à une précédente livraison. Lisez Section 4.9.7, « Changer le commentaire et l'auteur » pour découvrir comment cela fonctionne.

Montrer les propriétés de la révision
: Voir ou éditer les propriétés d'une révision, outre le commentaire de révision et l'auteur. Voir Section 4.9.7, « Changer le commentaire et l'auteur ».

Ajouter au presse-papier
: Copier les commentaires de révisions sélectionnées dans le presse-papier. Pour chaque révision, la révision, l'auteur, la date, le commentaire et la liste des éléments modifiés seront alors copiés.

Rechercher les commentaires de révision...
: Chercher les commentaires de révision correspondant au texte que vous saisissez. La recherche s'effectue dans les commentaires de révision que vous avez saisis ainsi que dans les résumés d'action créés par Subversion (présenté dans le panneau du bas). La recherche n'est pas sensible à la casse.

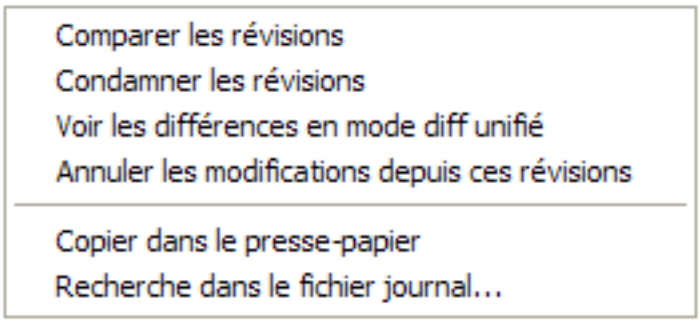


Figure 4.17. Menu contextuel du panneau supérieur avec 2 révisions sélectionnées

Si vous sélectionnez deux révisions en même temps (en utilisant le modificateur habituel **Ctrl**), le menu contextuel change et vous propose moins d'options :

Comparer des révision
: Comparer les deux révisions choisies en utilisant un outil de différenciation visuel. L'outil de différenciation par défaut est TortoiseMerge qui est fourni avec TortoiseSVN.

  Si vous choisissez cette option pour un dossier, une nouvelle boîte de dialogue apparaît, qui liste les fichiers modifiés et vous offre des options de comparaison avancées. Pour en savoir plus sur la comparaison de révisions, reportez-vous Section 4.10.3, « Comparer des répertoires ».

Annoter les révisions
: Annoter les deux révisions et comparer les résultats en utilisant un outil de différenciation visuel. Lisez Section 4.23.2, « Annoter les différences » pour plus de détails.

Voir les différences comme un diff unifié
: Voir les différences entre les deux révisions sélectionnées en tant que fichier de différences unifiées. Cela fonctionne pour les fichiers et les dossiers.

Ajouter au presse-papier
: Copier les commentaires de révision dans le presse-papiers comme décrit ci-dessus.

Rechercher les commentaires de révision...
: Chercher les commentaires de révision comme décrit ci-dessus.

Si vous sélectionnez deux ou plusieurs révisions (en utilisant les modificateurs habituels **Ctrl** ou **Shift**), le menu contextuel inclurera une entrée pour Annuler toutes les modifications qui ont été faites dans ces révisions. C'est la façon la plus facile d'annuler un groupe de révisions en une fois.

Vous pouvez également choisir de fusionner les révisions sélectionnées dans une autre copie de travail, comme décrit ci dessus.

Si toutes les révisions sélectionnées ont le même auteur, vous pouvez éditer le champ auteur en une seule opération.

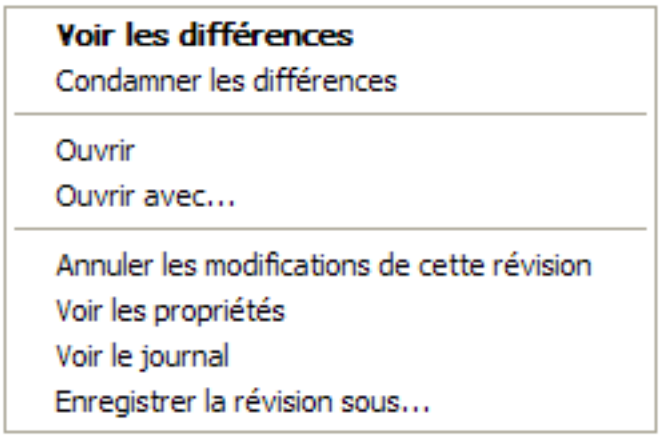


Figure 4.18. Le panneau inférieur de la boîte de dialogue du Journal avec le menu contextuel

Le panneau du bas de la boîte de dialogue Journal a aussi un menu contextuel qui vous permet de :

Voir les modifications
: Show changes made in the selected revision for the selected file. This context menu is only available for files shown as *modified*.

Annoter les modifications
: Annoter la révision sélectionnée et la révision précédente pour le fichier choisi et afficher les différences en utilisant un outil de différenciation visuel. Lisez Section 4.23.2, « Annoter les différences » pour plus de détails.

Voir comme un diff unifié
: Montrer les modifications en tant que fichier de différences unifiées. Ce menu contextuel est seulement disponible pour les fichiers affichés comme `Modifiés`.

Ouvrir / Ouvrir avec...
: Ouvrir le fichier sélectionné, avec le visualisateur par défaut pour ce type de fichier ou avec le programme de votre choix.

Annoter...
: Ouvrir la fenêtre d'annotation, vous permettant d'annoter la révision sélectionnée.

Annuler les modification pour revenir à cette révision
: Annuler les changements effectués au fichier sélectionné dans cette révision.

Voir les propriétés
: Voir les propriétés Subversion pour l'élément sélectionné.

Voir le journal
: Afficher le journal de révision pour le seul fichier choisi.

Récupérer les commentaires de fusion
: Montrer les commentaires de révision pour le seul fichier sélectionné, y compris les modifications fusionnées. Plus de détail dans Section 4.9.6, « Fonctionnalités de Suivi des Fusions ».

Sauvegarder la révision dans...
: Sauvegarder la révision sélectionnée dans un fichier pour que vous ayez une version antérieure de ce fichier.

Astuce

Vous avez peut être remarqué que nous faisons parfois référence à des modifications et parfois à des différences. Quelle différence y a-t-il ?

> Subversion utilise des numéros de révision pour deux raisons. Une révision représente l'état d'un référentiel à un moment donné, mais elle peut aussi matérialiser la liste des modifications ayant mené à cette révision, i.e. « Fait dans r1234 » veux dire que l'implémentation de la fonctionalité X a été faite via les modifications livrées dans la r1234. Pour préciser quel sens est utilisé, nous utilisons deux termes différents.
>
> Si vous sélectionnez deux révisions N et M, le menu contextuel vous permettra de voir les *différences* entre ces deux révisions. Dans Subversion, cette opération correspond à la commande `diff -r M:N`.
>
> Si vous sélectionnez une seule révision N, le menu contextuel vous proposera d'afficher les *modifications* faites dans cette révision. Dans le jargon de Subversion cela se traduit par `diff -r N-1:N` ou `diff -c N`.
>
> Le panneau du bas de la boîte de dialogue du Journal de révision présente les fichiers modifiés dans toutes les révisions sélectionnées. Le menu contextuel propose donc toujours d'afficher les `modifications`.

4.9.4. Obtenir plus de commentaires

Le boîte de dialogue du Journal ne montre pas toujours toutes les modifications faites pour plusieurs raisons :

- Pour un grand référentiel, il peut y avoir des centaines ou même des milliers de modifications et les parcourir toutes pourrait prendre longtemps. Normalement vous n'êtes intéressé que par les modifications les plus récentes. Par défaut, le nombre de commentaires parcourus est limité à 100, mais vous pouvez changer cette valeur dans TortoiseSVN → Configuration (Section 4.30, « Configuration de TortoiseSVN »),
- Quand la case Arrêt à la copie/renommage est cochée, Voir le journal s'arrêtera au moment où le fichier ou le dossier sélectionné ont été copiés d'ailleurs dans le référentiel. Cela peut être utile en regardant les branches (ou les étiquettes) puisque cela s'arrête à la racine de cette branche et donne une indication rapide des modifications faites dans cette branche seulement.

  Normalement vous voudrez laisser cette option décochée.TortoiseSVN se souvient de l'état de la case à cocher, donc il respectera votre préférence.

  Quand la boîte de dialogue Voir le journal est appelée depuis la boîte de dialogue Fusionner, la case est toujours cochée par défaut. La raison en est que la fusion concerne le plus souvent des modifications sur des branches et que revenir au-delà de la racine de la branche n'a alors aucun sens.

  Notez que Subversion implémente actuellement le renommage comme une paire copie/suppression, donc renommer un fichier ou un dossier causera aussi l'arrêt de l'affichage du journal si cette option est cochée.

Si vous voulez voir plus de commentaires de révision, cliquez sur le bouton 100 suivants pour récupérer les 100 commentaires suivants. Vous pouvez répéter cela autant de fois que nécessaire.

À côté de ce bouton il y a un bouton multi-fonctions qui se souvient de la dernière option pour laquelle vous l'avez utilisé. Cliquez sur la flèche pour voir les autres options offertes.

Utilisez Afficher la plage ... si vous voulez consulter une plage spécifique de révisions. Une boîte de dialogue vous demandera alors d'entrer les révisions de début et de fin.

Utilisez Afficher tout si vous voulez voir *tous* les commentaires depuis HEAD jusqu'à la révision 1.

4.9.5. Révision de la Copie de Travail Courante

La fenêtre de commentaires affichant les commentaires depuis HEAD et non de la révision de la copie de travail, il arrive souvent que des commentaires de révisions non encore importées dans votre copie de

travail soient affichés. Pour clarifier, le message de livraison correspondant à la révision de votre copie de travail est affiché en gras.

When you show the log for a folder the revision highlighted is the highest revision found anywhere within that folder, which requires a crawl of the working copy. This can be a slow operation for large working copies, and the log messages are not displayed until the crawl completes. If you want to disable or limit this feature you need to set a registry key `HKCU\Software\TortoiseSVN\RecursiveLogRev` as described in Section 4.30.10, « Réglages dans le registre ».

4.9.6. Fonctionnalités de Suivi des Fusions

Le logiciel Subversion 1.5 et ultérieur

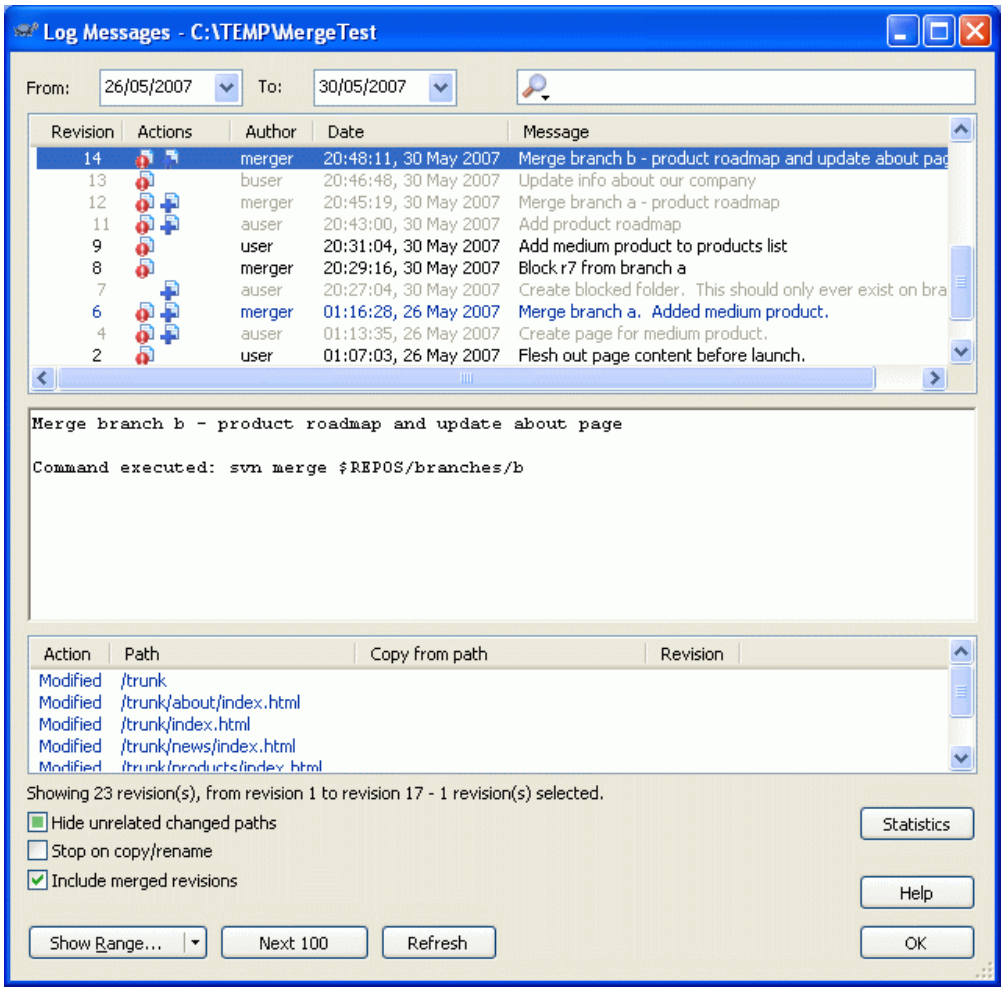


Figure 4.19. La Fenêtre du Journal de révision Montrant les Révisions Fusionnées

Si vous voulez voir quelles révisions ont été fusionnées dans un commit donné, utilisez la case à cocher Inclure les révisions fusionnées. Les commentaires de révision seront à nouveau affichés, mais ils inclueront également les commentaires des révisions fusionnées. Les révisions fusionnées sont affichées en gris car elles représentent des modifications effectuées dans une autre branche de l'arborescence.

Bien sûr, fusionner n'est jamais une mince affaire ! Pendant le développement des fonctionnalités dans une branche il y aura sûrement des fusions à faire avec la tête pour que la branche reste synchronisée avec la version de tête. Donc l'historique des fusions de la branche incluera une autre couche d'historique des fusions. Ces différentes couches sont affichées dans la fenêtre du journal avec une indentation différente.

4.9.7. Changer le commentaire et l'auteur

Les propriétés des révisions sont complètement différentes des propriétés de Subversion de chaque élément. Les Revprops sont des éléments descriptifs qui sont associés à un numéro de révision spécifique dans le référentiel, comme les commentaires de révision, date de livraison et nom de la personne ayant livré (auteur).

Parfois vous pourriez vouloir changer un commentaire que vous avez saisi, peut-être parce qu'il y a une faute d'orthographe dedans ou parce que vous voulez améliorer le message ou le changer pour d'autres raisons. Ou vous voulez changer l'auteur de la livraison parce que vous avez oublié de mettre en place l'authentification ou...

Subversion vous laisse changer les propriétés d'une révision aussi souvent que vous le souhaitez. Mais puisque de tels changements ne peuvent pas être annulés (ces changements ne sont pas versionnés) cette fonctionnalité est désactivée par défaut. Pour faire que cela fonctionne, vous devez mettre en place un hook pre-revprop-change. Veuillez vous référer au chapitre sur *Hook Scripts* [http://svnbook.red-bean.com/en/1.1/ch05s02.html#svn-ch-5-sect-2.1] dans le manuel de Subversion pour plus de détails sur la façon de le faire. Lisez Section 3.3, « Scripts de hook côté serveur » pour plus de détails sur l'implémentation de hooks sur une machine Windows.

Une fois que vous avez mis en place votre serveur avec les hooks requis, vous pouvez changer l'auteur et le commentaire (ou toute autre revprop) de n'importe quelle révision, en utilisant le menu contextuel du panneau supérieur de la boîte de dialogue du Journal. Vous pouvez également éditer un commentaire de révision par l'intermédiaire du menu contextuel du panneau du milieu.

Avertissement

Parce que les propriétés de révision de Subversion ne sont pas versionnées, modifier une telle propriété (par exemple, la propriété du commentaire de livraison `svn:log`) écrasera la valeur précédente de cette propriété *pour toujours*.

4.9.8. Filtrer les commentaires

Si vous voulez limiter les commentaires pour afficher seulement ceux qui vous intéressent, plutôt que de faire défiler une liste de centaines de commentaires, vous pouvez utiliser les commandes de filtre en haut de la boîte de dialogue du Journal. Les dates de début et de fin vous permettent de limiter les résultats à une plage de dates connue. La boîte de recherche vous permet de n'afficher que les messages qui contiennent une expression particulière.

Cliquez sur l'icone de recherche pour sélectionner le type d'information que vous souhaitez rechercher, et pour choisir le mode *regex*. Vous ne devriez avoir besoin que d'une simple recherche de texte, mais si vous souhaitez plus de flexibilité, vous pouvez alors utiliser les expressions regulières. Si vous passez la souris au dessus de la boite, une tooltip vous donnera des informations sur comment utiliser les fonctions d'expressions regulières. Vous pouvez également trouver l'aide en ligne et un tutoriel à cette adresse *http://www.regular-expressions.info/*. Seuls les commentaires de révision passant le filtre (i.e. qui *matchent* le filtre) sont affichés.

Pour faire en sorte que tous les commentaires *ne vérifiant pas* le filtre s'affichent, commencez l'expression régulière par un point d'exclamation ('!'). Par exemple, la chaine `!username` ne montrera que les entrées n'ayant pas été livrées par `username`.

Notez que ces filtres agissent sur les commentaires déjà récupérés. Ils ne provoquent pas de téléchargement de commentaires du référentiel.

Vous pouvez aussi filtrer les noms de chemin dans le panneau du bas en utilisant la case à cocher Cacher les chemins changés sans relation. Les chemins liés sont ceux qui contiennent le chemin utilisé pour montrer le journal. Si vous parcourez le journal pour un dossier, cela signifie tout dans ce dossier ou en dessous. Pour un fichier cela signifie juste ce fichier. La case à cocher a trois états : vous pouvez montrer tous les chemins, griser ceux sans relation, ou cacher complètement les chemins sans relation.

Parfois, vos habitudes de travail nécessiteront que les commentaires de révision aient un format particulier, ce qui signifie que le texte résumant les modifications ne sera pas visible dans le panneau supérieur. La propriété`tsvn:logsummary` peut être utilisée pour afficher dans le panneau supérieur une partie du commentaire de révision. Lisez Section 4.17.2, « Propriétés du projet TortoiseSVN » pour savoir comment utiliser cette propriété.

Pas de Formattage des Commentaires depuis l'Explorateur de Référentiel

Le formattage dépendant des propriétés d'accès de subversion, vous ne verrez le résultat que si vous utilisez une copie de travail exportée. Afficher les propriétés à distance étant une opération lente, vous ne pourrez pas l'effectuer depuis l'explorateur de référentiel.

4.9.9. Informations statistiques

Le bouton Statistiques fait apparaître une boîte montrant des informations intéressantes sur les révisions affichées dans la boîte de dialogue de Journal. Elle montre combien d'auteurs ont travaillé, combien de livraisons ils ont fait, la progression par semaine et beaucoup plus. Maintenant vous pouvez voir d'un coup d'oeil qui a travaillé le plus dur et qui se relâche ;-)

4.9.9.1. Page des statistiques

Cette page vous donne tous les nombres auxquels vous pouvez penser, en particulier la période et le nombre de révisions couvertes et quelques valeurs min/max/moyennes.

4.9.9.2. Page de livraisons par auteur

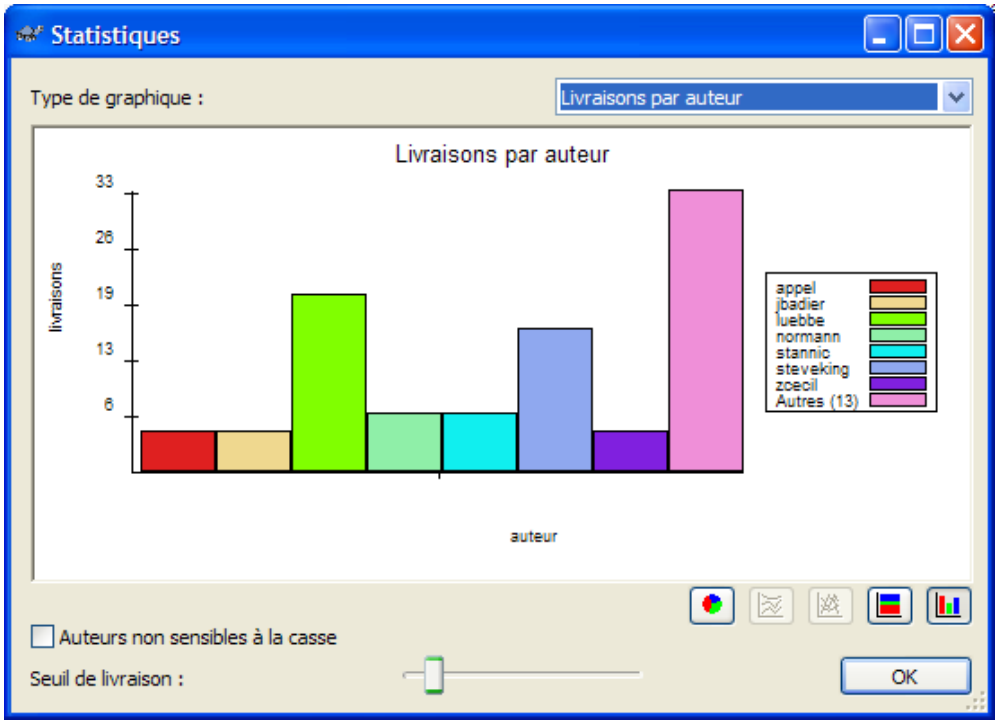


Figure 4.20. Histogramme de livraisons par auteur

Ce graphique vous montre quels auteurs ont été actifs sur le projet par un simple histogramme, un histogramme empilé ou un camembert.

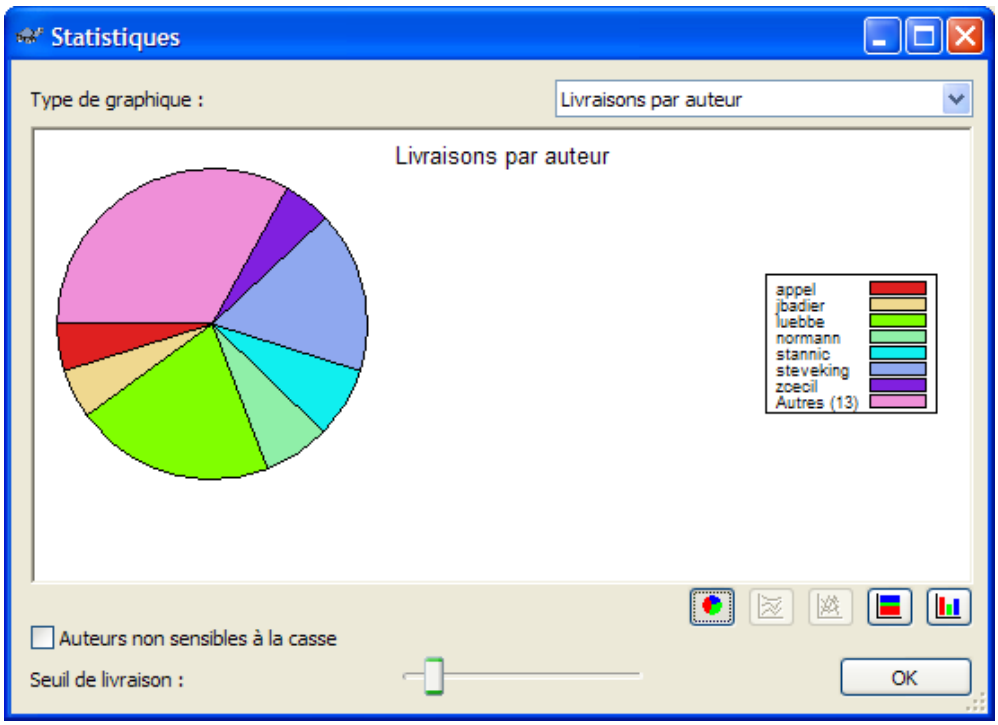


Figure 4.21. Camembert de livraisons par auteur

Quand il y a quelques auteurs majeurs et beaucoup de contributeurs mineurs, le nombre de segments minuscules peut rendre le graphique plus difficile à lire. Le slider en bas vous permet de définir un seuil (en pourcentage du total des livraisons) en dessous duquel l'activité est groupée dans une catégorie *Autres*.

4.9.9.3. Page de livraisons par date

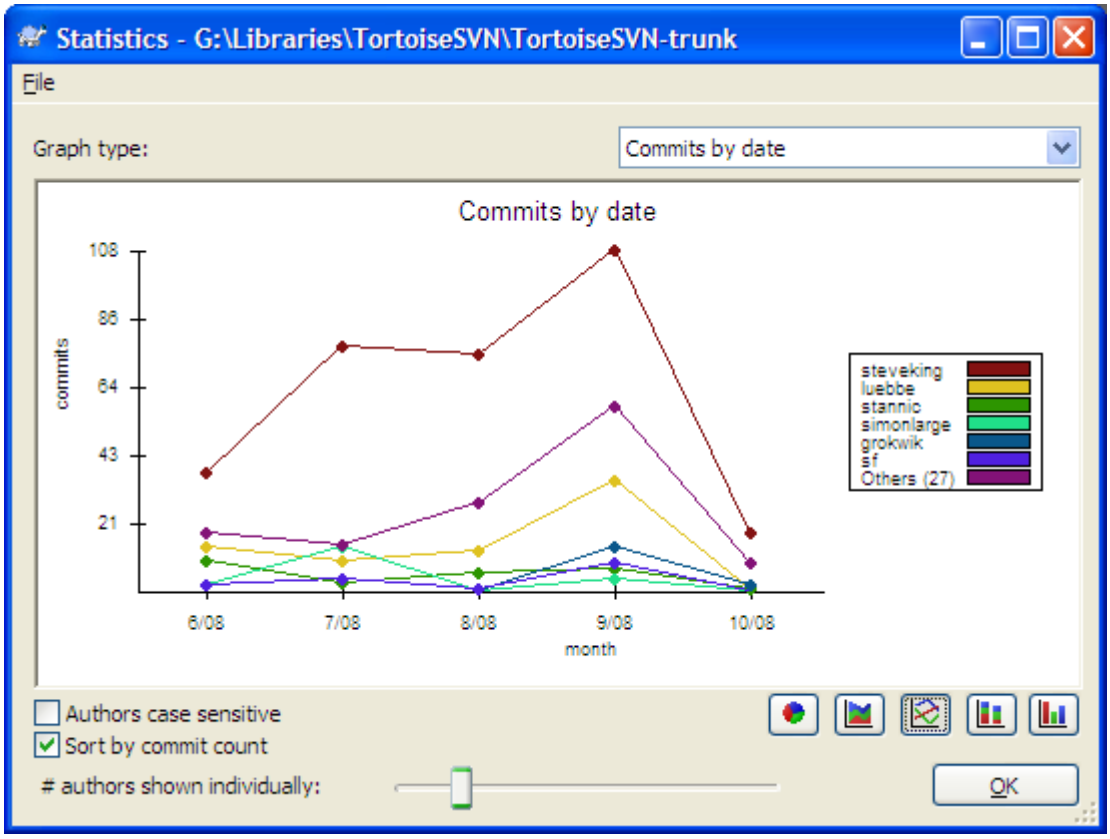


Figure 4.22. Graphique de livraisons par date

Cette page vous donne une représentation graphique de l'activité du projet en termes de nombre de livraisons *et* d'auteurs. Cela donne une certaine idée de quand un projet est actif et de qui travaillait à quel moment.

Quand il y a plusieurs auteurs, vous obtiendrez beaucoup de lignes sur le graphique. Il y a deux vues disponibles ici : *normale*, où l'activité de chaque auteur est relative à la ligne de base et *empilée*, où l'activité de chaque auteur est relative à la ligne d'en dessous. La dernière option évite les lignes qui se traversent, ce qui peut rendre le graphique plus facile à lire, mais la production d'un auteur moins facile à voir.

Par défaut l'analyse est sensible à la casse donc les utilisateurs `PeterEgan` et `PeteRegan` sont considérés comme des auteurs différents. Cependant, dans de nombreux cas les noms des utilisateurs ne sont pas sensibles à la casse et sont parfois saisis de façon variable, donc vous pourriez vouloir que `DavidMorgan` et `davidmorgan` soient considérés comme la même personne. Utilisez la case à cocher Auteurs non sensibles à la casse pour contrôler la manière dont cela est traité.

Notez que les statistiques couvre la même période que la boîte de dialogue du Journal. Si elle ne montre qu'une révision alors les statistiques ne vous diront pas grand chose.

4.9.10. Mode hors ligne

Si le serveur est inaccessible, et que vous avez activé la mise en cache des commentaires de révision, vous pouvez utiliser la boîte de dialogue du Journal et le graphique des révisions en mode hors ligne. Ces outils utiliseront alors les données mises en cache, ce qui vous permet de continuer à travailler même si les informations sont incomplètes ou ne sont pas à jour.

Vous avez ici trois choix :

Mode hors ligne
: Fini l'opération courante en mode hors ligne, mais réessaie avec le référentiel à la prochaine demande d'informations.

Constamment hors ligne
: Reste en mode hors ligne jusqu'à la prochaine demande explicite de vérification. Voir Section 4.9.11, « Refraîchissement de l'affichage ».

Annuler
: Si vous ne voulez pas continuer l'opération en cours avec le risque d'avoir des données bancales, annulez.

La case à cocher Définir par défaut empêche cette fenêtre de réapparaitre et retient toujours l'option suivante choisie. Vous pouvez toujours modifier (ou supprimer) cette valeur par défaut plus tard dans le menu TortoiseSVN → Paramètres.

4.9.11. Refraîchissement de l'affichage

Si vous voulez vérifier si de nouveaux commentaires sont arrivés sur le serveur, vous pouvez simplement rafraîchir l'interface en appuyant sur **F5**. Si vous utilisez un cache pour les commentaires (ce qui est le comportement par défaut), cette vérification se fera et seuls les nouveaux commentaires seront rappatriés. Si le cache des commentaires était en mode hors ligne, il tentera de se reconnecter.

Si vous utilisez le cache du journal et si vous pensez que le contenu du message ou son auteur ont pu changer, vous pouvez utiliser **Shift+F5** ou **Ctrl-F5** pour récupérer à nouveau les messages affichés à partir du serveur et mettre à jour le cache journal. Notez que ceci affecte uniquement les messages affichés à cet instant et que ceci n'invalide pas le cache complet pour ce référentiel.

4.10. Voir les différences

Une des exigences les plus courantes dans le développement de projet est de voir ce qui a changé. Vous pourriez vouloir regarder les différences entre deux révisions du même fichier, ou les différences entre deux fichiers séparés. TortoiseSVN fournit un outil intégré appelé TortoiseMerge pour voir les différences dans les fichiers texte. Pour voir les différences dans les fichiers image, TortoiseSVN a aussi un outil appelé TortoiseIDiff. Bien sûr, vous pouvez utiliser votre propre programme de comparaison favori si vous le souhaitez.

4.10.1. Différences de fichier

Changements locaux
: Si vous voulez voir quels changements *vous* avez fait dans votre copie de travail, utilisez juste le menu contextuel de l'explorateur et choisissez TortoiseSVN → Voir les différences.

Comparaison avec une autre branche/étiquette
: Si vous voulez voir ce qui a changé sur le tronc (si vous travaillez sur une branche) ou sur une branche spécifique (si vous travaillez sur le tronc), vous pouvez utiliser le menu contextuel de l'explorateur. Maintenez juste la touche **Maj** tandis que vous faites un clic droit sur le fichier. Sélectionnez alors TortoiseSVN → Diff avec l'URL. Dans la boîte de dialogue suivante, spécifiez l'URL dans le référentiel avec laquelle vous voulez comparer votre fichier local.

  Vous pouvez aussi utiliser l'explorateur de référentiel et sélectionner deux arborescences à comparer, deux étiquettes peut-être, ou une branche/étiquette et le tronc. Le menu contextuel vous permet là de les comparer en utilisant Comparer les révisions. Plus d'informations dans Section 4.10.3, « Comparer des répertoires ».

Comparaison avec une révision précédente
: Si vous voulez voir les différences entre une révision particulière et votre copie de travail, utilisez la boîte de dialogue de Journal de révision, sélectionnez la révision qui vous intéresse, puis choisissez Comparaison avec la copie de travail dans le menu contextuel.

Si vous voulez voir la différence entre la dernière révision et votre copie active, en supposant que la copie active n'ait pas été modifiée, faites simplement un clic droit sur le fichier. Ensuite sélectionnez TortoiseSVN → Différences avec la version précédente. Ceci calculera les différences entre la révision avant la date de dernière Livraison (comme sauvegardé dans votre copie active) et la BASE active. Ceci vous montre la denière modification effectuée sur ce fichier et qui l'a amené à l'état dans lequel vous le voyez maintenant dans votre copie active. Ceci ne va pas montrer les modifications ultérieures à votre copie active.

Comparaison entre deux révisions précédentes
: Si vous voulez voir les différences entre deux révisions déjà livrées, utilisez la boîte de dialogue de Journal de révision et sélectionnez les deux révisions que vous voulez comparer (en utilisant le modificateur habituel **Ctrl**). Puis choisissez Comparer les révisions à partir du menu contextuel.

  Si vous avez fait ceci depuis le journal sur un répertoire, une boîte de dialogue Comparer les révisions apparaît, affichant la liste des fichiers dans ce dossier. Plus d'informations dans Section 4.10.3, « Comparer des répertoires ».

Tous les changements faits dans une livraison
: Si vous voulez voir les changements faits à tous les fichiers dans une révision particulière en une vue, vous pouvez utiliser la sortie en mode Diff-Unifié (format de patch GNU). Cela montre seulement les différences replacées dans leur contexte. Il est plus difficile à lire qu'une comparaison de fichier visuelle, mais montrera tous les changements ensemble. À partir de la boîte de dialogue du Journal de Révision sélectionnez la révision qui vous intéresse, puis choisissez Voir les différences en mode diff unifié à partir du menu contextuel.

Comparaison entre fichiers
: Si vous voulez voir les différences entre deux fichiers différents, vous pouvez le faire directement dans l'explorateur en sélectionnant les deux fichiers (en utilisant le modificateur habituel **Ctrl**). Alors à partir du menu contextuel de l'explorateur, sélectionnez TortoiseSVN → Voir les différences.

Différence entre fichiers/répertoires de la CdT et une URL
: Si vous voulez voir les différences entre un fichier de votre copie de travail, et un fichier dans n'importe quel référentiel Subversion, vous pouvez le faire directement dans l'explorateur en sélectionnant le fichier puis en maintenant enfoncé la touche **Shift** en faisant un clic droit pour obtenir le menu contextuel. Sélectionnez TortoiseSVN → Diff avec l'URL. Vous pouvez faire la même chose pour un répertoire de la copie de travail. TortoiseMerge vous montre ces différences de la même manière qu'un patch - une liste des fichiers modifiés que vous pouvez voir un par un.

Comparaison avec les informations d'annotation
: Si vous voulez ne pas voir que les différences mais aussi l'auteur, la révision et la date des modifications effectuées, vous pouvez combiner les rapports de différenciation et d'annotation au sein de la boîte de dialogue du journal de révision. Lisez Section 4.23.2, « Annoter les différences » pour plus de détails.

Comparaison entre répertoires
: Les outils intégrés fournis avec TortoiseSVN ne supportent pas l'observation de différences entre les hiérarchies de répertoire. Mais si vous avez un outil externe qui supporte vraiment cette fonctionnalité, vous pouvez l'utiliser à la place. Dans Section 4.10.5, « Outils de différenciation/fusion externes » nous vous parlerons de quelques outils que nous avons utilisés.

Si vous avez fourni un outil de diff alternatif, vous pouvez utiliser la touche **Shift** quand vous sélectionnez la commande Diff pour utiliser l'outil alternatif. Lisez Section 4.30.5, « Réglages des programmes externes » pour savoir comment configurer les autres outils de diff.

4.10.2. Options de fins de ligne et d'espacement

Parfois dans la vie d'un projet il se peut que vous modifiiez les fins de ligne de CRLF en LF, ou que vous changiez l'alinéa d'une section. Malheureusement ceci va marquer un grand nombre de lignes comme étant modifiées, bien qu'il n'y ait pas de variation dans la signification du code. Ces options ici vont aider à gérer ces modifications lorsqu'il s'agira de comparer et d'appliquer des différences. Vous pourrez

voir ces réglages dans les boîtes de dialogue Fusionner et Annoter, tout comme dans les paramètres de FusionTortoise.

Ignorer les caractères de fins de ligne exclue les modifications seulement dues aux style des caractères de fin de ligne.

Compare les caractères d'espacement inclue les modifications d'indentation et d'espaces à la liste des lignes ajoutées/supprimées.

Ignore les modifications des caractères d'espacement exclue les modifications étant seulement dues au style ou à la quantité de caractères d'espacement, i.e. changement dans l'indentation ou remplacement des tabulations par des espaces. Ajouter des espaces là où il n'y en avait pas, ou tous les retirer est toujours affiché comme étant une modification.

Ignorer tous les caractères d'espacement exclue toutes les modifications dues aux caractères d'espacement.

Naturellement, chaque ligne modifiée est toujours incluse dans le diff.

4.10.3. Comparer des répertoires

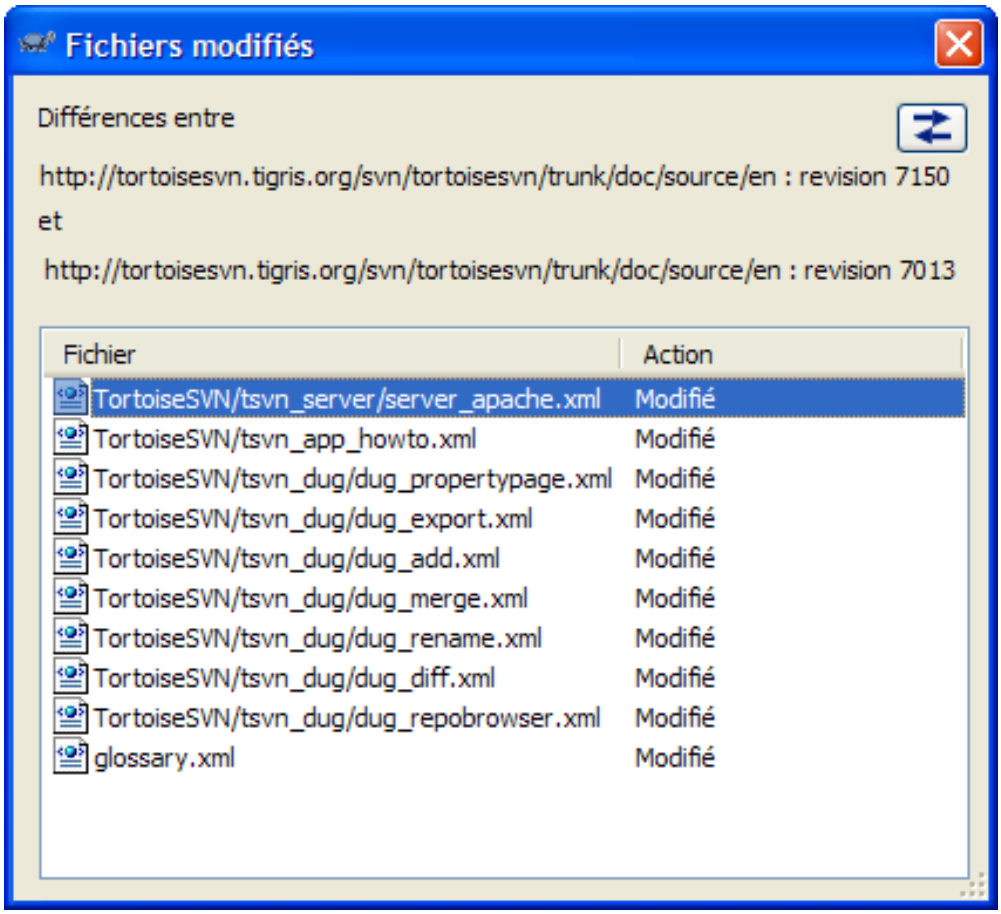


Figure 4.23. La boîte de dialogue Comparer les révisions

Quand vous sélectionnez deux arborescences dans l'explorateur de référentiel ou quand vous sélectionnez deux révisions d'un répertoire dans la boîte de dialogue du journal, vous pouvez Menu contextuel → Compare les révisions.

Cette boîte de dialogue affiche une liste des fichiers modifiés et vous permet de les comparer ou de les annoter individuellement en utilisant le menu contextuel.

Vous pouvez exporter un *arbre des modifications*, qui est utile si vous avez besoin d'envoyer la structure de l'arbre de votre projet à un tiers, contenant uniquement les fichiers qui ont été modifiés. Cette opération fonctionne sur les fichiers sélectionnés uniquement, ainsi vous devez sélectionner les fichiers qui vous intéressent - ce qui signifie souvent tous les fichiers - puis ensuite sélectionner Menu Contextuel → Exporter la sélection vers.... Vous serez invités à chosir un emplacement où sauvegarder l'arbre des modifications.

Vous pouvez également exporter la *liste* des fichiers modifiés dans un fichier texte en utilisant le menu contextuel → Enregistrer la liste de fichiers sélectionnés vers....

Si vous voulez exporter la listes des fichiers *et* aussi les actions (modifié, ajouté, supprimé), vous pouvez le faire en utilisant le menu contextuel → Copier la sélection dans le presse-papier.

Le bouton en haut vous permet de modifier le sens de la comparaison. Vous pouvez afficher les changements nécessaires pour arriver de A vers B, ou si vous préférez, de B vers A.

Les boutons avec les numéros de révision peuvent être utilisés pour passer à un éventail de révisions différent. Lorsque vous changez d'éventail, la liste des éléments qui diffèrent entre les deux révisions sera mise à jour automatiquement.

Si la liste des noms de fichier est très longue, vous pouvez utiliser le champ de recherche pour réduire sa taille. Remarquez qu'une simple recherche texte est utilisée, donc si vous voulez restreindre la liste aux fichiers C vous devrez rechercher `.c` et non `*.c`.

4.10.4. Comparaison des images en utilisant TortoiseIDiff

Il y a beaucoup d'outils disponibles pour comparer des fichiers texte, incluant notre propre TortoireMerge, mais souvent, nous voulons également voir les modifications effectuées sur un fichier image. C'est pourquoi nous avons créé TortoiseIDiff.

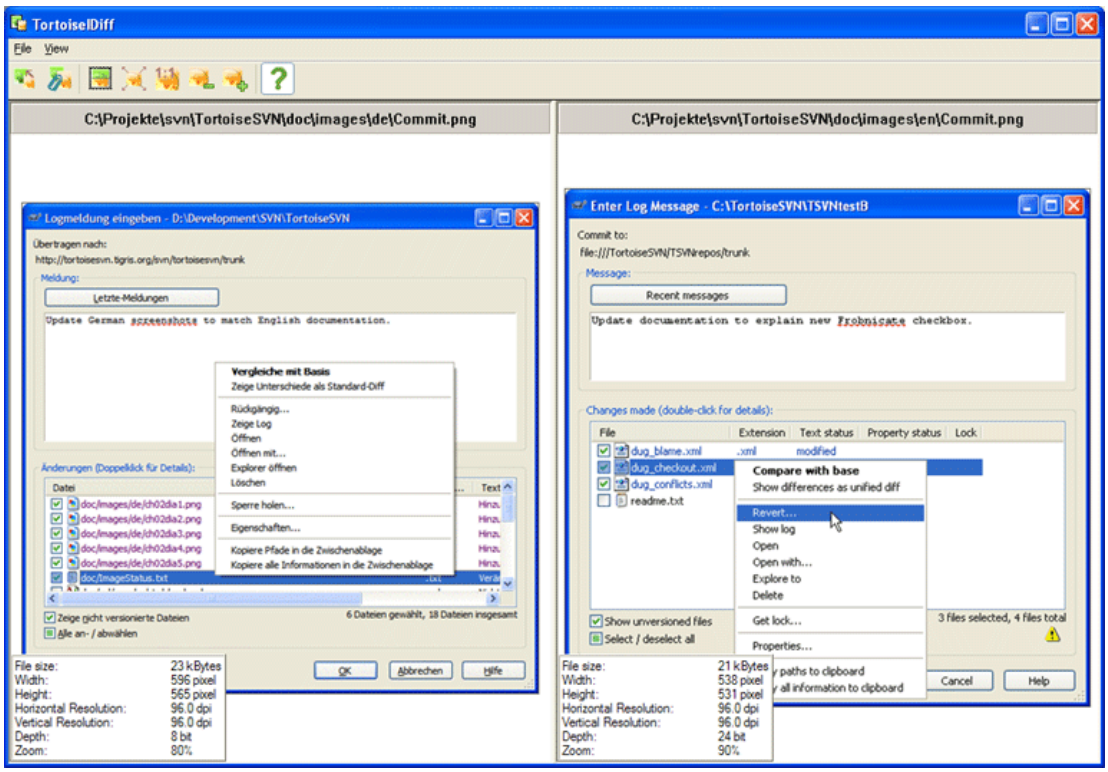


Figure 4.24. Le visualiseur de différences d'images

TortoiseSVN → Voir les différences pour n'importe quel format d'image commun démarrera TortoiseIDiff pour afficher les différences d'image. Par défaut les images sont affichées côte à côte mais vous pouvez utiliser le menu Affichage ou la barre d'outils pour basculer vers une vue haut-bas à la place, ou si vous préférez, vous pouvez superposer les images et utiliser une visionneuse.

Naturellement vous pouvez aussi faire un zoom avant, arrière tout en vous déplacant sur l'image. Vous pouvez également effectuer un mouvement de l'image à gauche simplement en la faisant glisser. Si vous sélectionnez l'option Lier les images ensemble, alors les contrôles de déplacement (barre de défilement, molette) des deux images sont reliés.

Dans l'image, une boîte d'informations affiche les détails concernant le fichier image, comme la taille en pixels, la résolution et la profondeur des couleurs. Si cette boîte se trouve dans le chemin, utilisez Affichage → Informations de l'image pour la masquer. Vous pouvez obtenir la même information dans une info-bulle lorsque vous passez la souris sur la barre de titre de l'image.

Lorsque les images sont recouvertes, l'intensité relative des images (transparence) est contrôlée grâce à un ascenseur situé sur la gauche. Vous pouvez directement cliquer n'importe où sur l'ascenseur, ou faire glisser le curseur pour modifier interactivement sa valeur. **Ctrl+Shift**-Roulette pour modifier la transparence.

Le bouton au dessus de l'ascenseur permet de passer de 0% à 100% de transparence, et si vous double-cliquez dessus, la transparence change automatiquement chaque seconde jusqu'à ce que vous cliquiez de nouveau. Ce qui peut être utile lorsque vous cherchez beaucoup de petites modifications.

Parfois vous voulez voir une différence plutot qu'un mélange. Vous pourriez avoir les fichiers d'images de deux révisions d'un circuit imprimé et dont vous souhaitez voir quelles pistes ont changées. Si vous désactivez le mode alpha blend, la différence sera présentée comme un *XOR* de la valeur des pixels de couleur. Les zones inchangées seront de couleur blanche et les changements seront colorés.

4.10.5. Outils de différenciation/fusion externes

Si les outils que nous fournissons ne font pas ce dont vous avez besoin, essayez un des nombreux programmes open-source ou commerciaux disponibles. Chacun a ses préférences et cette liste n'est en aucun cas exhaustive, mais en voici quelques-uns intéressants :

WinMerge
: *WinMerge* [http://winmerge.sourceforge.net/] est un grand outil de comparaison open-source qui peut aussi manipuler les répertoires.

Perforce Merge
: Perforce est un RCS commercial, mais vous pouvez télécharger l'outil de différenciation/fusion gratuitement . Obtenez plus d'informations sur *Perforce* [http://www.perforce.com/perforce/products/merge.html].

KDiff3
: KDiff3 est un outil de comparaison gratuit qui peut aussi manipuler des répertoires. Vous pouvez le télécharger *ici* [http://kdiff3.sf.net/].

ExamDiff
: ExamDiff Standard est un logiciel gratuit. Il peut manipuler les fichiers, mais pas les répertoires. ExamDiff Pro est shareware et ajoute un certain nombre de goodies incluant la comparaison de répertoires et des possibilités d'édition. Dans les deux, les versions 3.2 et supérieures peuvent gérer l'unicode. Vous pouvez les télécharger chez *PrestoSoft* [http://www.prestosoft.com/].

Beyond Compare
: Semblable à ExamDiff Pro, c'est un outil de différenciation shareware excellent qui peut manipuler les comparaisons de répertoires et l'unicode. Téléchargez-le chez *Scooter Software* [http://www.scootersoftware.com/].

Araxis Merge
: Araxis Merge est un outil commercial utile pour comparer et fusionner tant les fichiers que les dossiers. Il fait la comparaision à trois voies pour les fusions et a des liens de synchronisation à utiliser si vous avez changé l'ordre des fonctions. *Araxis* [http://www.araxis.com/merge/index.html].

SciTE
: Cet éditeur de texte inclut la coloration syntaxique pour les diffs unifié, les rendant beaucoup plus faciles à lire. Téléchargez-le chez *Scintilla* [http://www.scintilla.org/SciTEDownload.html].

Notepad2
: Notepad2 est conçu comme un remplaçant pour le programme standard de Bloc-notes Windows et est basé sur le contrôle d'édition open-source de Scintilla. En plus d'être bon pour voir les diffs unifiés, il est beaucoup mieux que le bloc-notes Windows pour la plupart des travaux. Téléchargez-le gratuitement *ici* [http://www.flos-freeware.ch/notepad2.html].

Lisez Section 4.30.5, « Réglages des programmes externes » pour des informations sur la façon de configurer TortoiseSVN pour utiliser ces outils.

4.11. Ajouter de nouveaux fichiers et répertoires

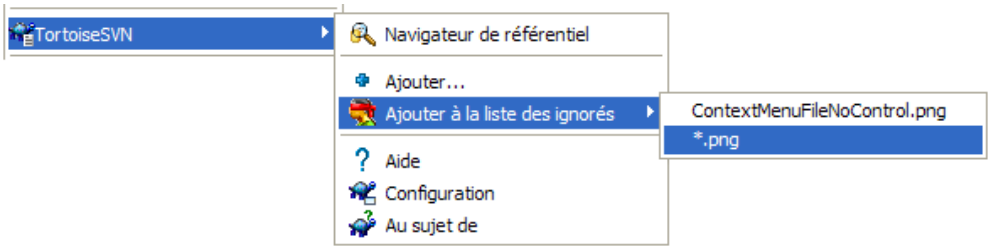


Figure 4.25. Menu contextuel de l'explorateur pour les fichiers non versionnés

Si vous avez créé de nouveaux fichiers et/ou de nouveaux répertoires pendant votre processus de développement alors vous devez aussi les ajouter au contrôle de source. Sélectionnez les fichiers et/ou les répertoires et utilisez TortoiseSVN → Ajouter.

Après que vous ayez ajouté les fichiers/répertoires au contrôle de source, le fichier apparaît avec une icône de recouvrement `ajouté` qui veut dire que vous devez d'abord livrer votre copie de travail pour rendre ces fichiers/répertoires disponibles aux autres développeurs. L'ajout d'un fichier/répertoire n'affecte *pas* le référentiel !

Plusieurs ajouts

Vous pouvez aussi utiliser la commande Ajouter sur des dossier déjà versionnés. Dans ce cas, la boîte de dialogue ajouter vous montrera tous les fichiers non versionnés à l'intérieur de ce dossier versionné. C'est utile si vous avez beaucoup de nouveaux fichiers et que vous avez besoin de les ajouter en une fois.

Pour ajouter des fichiers de l'extérieur de votre copie de travail, vous pouvez utiliser le glisser-déplacer :

1. sélectionnez les fichiers que vous voulez ajouter
2. glissez-déposer avec le bouton droit ces fichiers dans le nouvel emplacement à l'intérieur de la copie de travail
3. relâchez le bouton droit de la souris

4. sélectionnez Menu contextuel → SVN Ajouter les fichiers à cette CdT. Les fichiers seront alors copiés dans la copie de travail et ajoutés au contrôle de version.

Vous pouvez également ajouter des fichiers dans la copie de travail en les faisant simplement glisser dans la fenêtre de livraison.

Si vous ajoutez un élément par erreur, vous pouvez annuler l'opération avant de livrer en utilisant la commande TortoiseSVN → Annuler l'ajout....

4.12. Copier/Déplacer/Renommer des Fichiers et des Dossiers

Il arrive souvent que les fichiers dont vous avez besoin soient dans une autre projet du référentiel, et que vous vouliez simplement en récupérer une copie. Vous pourriez simplement les copier puis les ajouter comme décrit ci dessus, mais vous perdriez ainsi tout l'historique. Et si plus tard vous corrigez un bogue dans les fichiers originaux, vous ne pouvez que fusionner le correctif.

Le moyen le plus simple de copier des fichiers et/ou des répertoires au sein de la copie de travail est d'utiliser le menu contextuel clique droit. Lorsque vous glissez/déposez un fichier/répertoire en utilisant le bouton droit de la souris d'une copie de travail vers une autre, voire à l'intérieur d'un même répertoire, un menu contextuel s'affiche quand vous relachez le bouton de la souris.

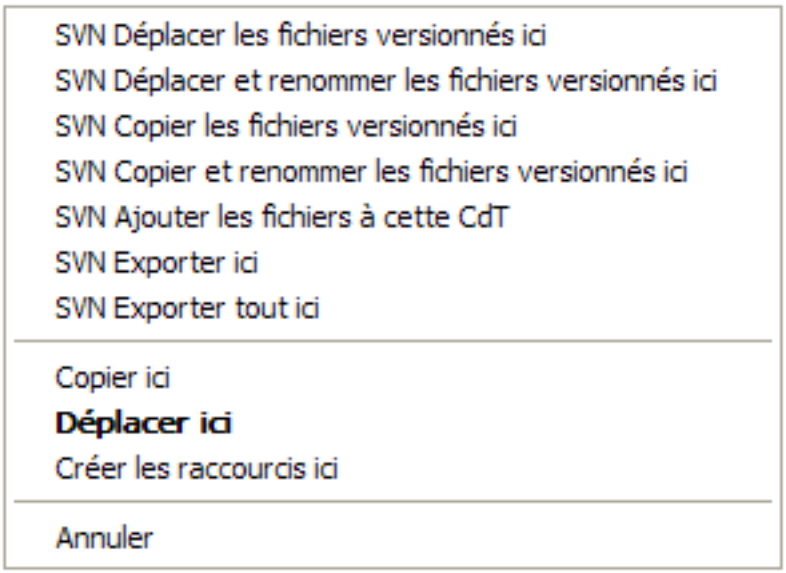


Figure 4.26. Menu pour un répertoire sous contrôle de version lors d'un glisser-déplacer avec le bouton droit

Vous pouvez à présent copier du contenu déjà versionné à un autre endroit, et éventuellement le renommer d'une même action.

Vous pouvez également copier ou déplacer des fichiers versionnés à l'intérieur de la copie de travail, ou entre deux copies de travail, en utilisant le maintenant familier copier/coller. Utilisez Copier ou Couper pour copier un ou plusieurs éléments versionnés dans le presse papier. Si celui ci contient du contenu versionné, vous pouvez alors utiliser TortoiseSVN → Coller (note: PAS l'action Windows standard Coller) pour copier ou déplacer ces éléments au nouvel endroit dans la copie de travail.

Vous pouvez copier des fichiers ou des répertoires depuis votre copie de travail vers un autre endroit du référentiel en utilisant TortoiseSVN → Branche/Tag. Lisez Section 4.19.1, « Créer une branche ou une étiquette » pour plus d'informations.

Vous pouvez localiser l'ancienne version d'un fichier ou d'un répertoire dans la fenêtre de commentaires et la copier directement à un autre endroit du référentiel en utilisant Menu Contextuel → Créer une branche/étiquette depuis la révision. Lisez Section 4.9.3, « Obtenir des informations supplémentaires » pour plus d'informations.

Vous pouvez également utiliser l'explorateur de référentiel pour localiser du contenu, et le copier directement dans votre copie de travail, ou entre deux endroits du référentiel. Lisez Section 4.24, « l'explorateur de référentiel » pour plus d'informations.

Impossible de copier entre référentiels

Tandis que vous pouvez copier ou déplacer des fichiers et dossiers contenus *dans* un référentiel, *vous ne pouvez pas* copier ou déplacer d'un référentiel à l'autre tout en préservant l'historique en utilisant TortoiseSVN. Même si les dépôts sont sur le même serveur. Tout ce que vous pouvez faire est de copier le contenu dans son état actuel et l'ajouter comme nouveau contenu au second référentiel.

Si vous n'êtes pas sûr de savoir si deux URL pointant sur le même serveur font référence à un même référentiel, utilisez l'explorateur de référentiel pour ouvrir une des deux et localiser sa racine. Si vous voyez les deux chemins dans la même fenêtre d'explorateur de référentiel alors ils sont dans le même référentiel.

4.13. Ignorer des fichiers et des répertoires

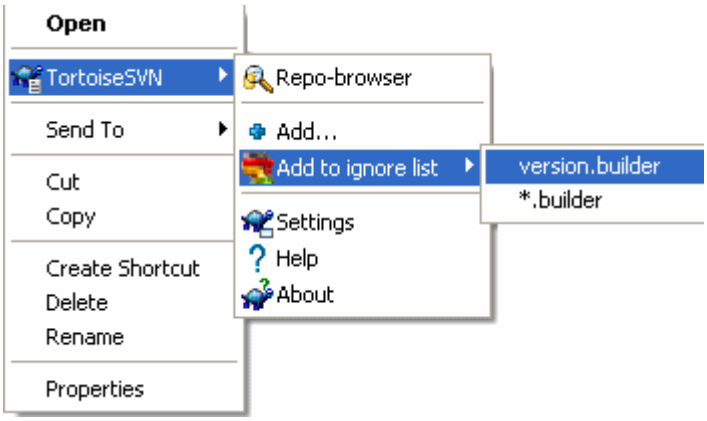


Figure 4.27. Menu contextuel de l'explorateur pour les fichiers non versionnés

Dans la plupart des projets vous aurez des fichiers et des dossiers qui n'auront pas à être sous contrôle de version. Par exemple, les fichiers de compilation, `*.obj`, `*.lst`, peut-être un dossier destiné à recevoir l'exécutable. Chaque fois que vous livrez des changements, TSVN vous montre les fichiers non versionnés, encombrant la liste des fichiers dans la fenêtre de livraison. Vous pouvez bine sûr désactiver cet affichage, mais vous pourriez alors oublier d'ajouter un nouveau fichier source.

La meilleure façon d'éviter ces problèmes est d'ajouter les fichiers à la liste des ignorés du projet. De cette manière, ils ne s'afficheront jamais dans la fenêtre de livraison, mais les vrais fichiers source non versionnés le seront toujours.

Si vous faites un clic droit sur un seul fichier non versionné et sélectionnez la commande TortoiseSVN → Ajouter à la liste des ignorés à partir du menu contextuel, un sous-menu apparaît vous permettant de ne choisir que ce fichier, ou tous les fichiers avec la même extension. Si vous sélectionnez plusieurs fichiers, il n'y a pas de sous-menu et vous pouvez seulement ajouter ces fichiers/dossiers spécifiques.

Si vous voulez supprimer un ou plusieurs éléments de la liste des ignorés, faites un clic droit sur ces éléments et sélectionnez TortoiseSVN → Retirer de la liste des ignorés Vous pouvez aussi directement accéder à la propriété `svn:ignore` d'un dossier. Cela vous permet de spécifier des modèles plus généraux en utilisant des jokers, décrit dans la section ci-dessous. Lisez Section 4.17, « Configuration des projets » pour plus d'informations sur la définition directe des propriétés. Notez qu'il faut une règle de filtrage par ligne. Les séparer d'un espace ne fonctinone pas.

La liste des ignorés globale

Une autre façon d'ignorer des fichiers est de les ajouter à la *liste des ignorés globale*. La grande différence ici, c'est que la liste des ignorés globale est une propriété client. Elle s'applique à *tous* les projets Subversion, mais sur le PC client uniquement. En général, c'est mieux d'utiliser la propriété `svn:ignore` où c'est possible, parce qu'elle peut être appliquée à des secteurs spécifiques du projet et elle fonctionne pour tous ceux qui extraient le projet. Lisez Section 4.30.1, « Configuration générale » pour plus d'informations.

Ignorer les éléments versionnés

Les fichiers et les répertoires versionnés ne peuvent jamais être ignorés - c'est une fonctionnalité de Subversion. Si vous avez versionné un fichier par erreur, lisez Section B.8, « Ignorer les fichiers déjà versionnés » pour savoir comment le « déversionner ».

4.13.1. L'utilisation des pattern matching dans la liste des fichier à ignorer

Les modèles d'exclusion de Subversion se servent de l'expansion des jokers (globbing) dans les noms de fichier, une technique à l'origine utilisée sous Unix pour spécifier des fichiers utilisant des méta-caractères comme caractères de remplacement. Les caractères suivants ont une signification spéciale :

*
: Correspond à n'importe quelle chaîne de caractères (de 0 à n caractères), y compris la chaîne vide (aucun caractère).

?
: Correspond à n'importe quel caractère.

[...]
: Correspond à n'importe lequel des caractères inclus dans les crochets. Dans les crochets, une paire de caractères séparés par « - » correspond à n'importe quel caractère lexicalement entre les deux. Par exemple `[AGm-p]` correspond à `A`, `G`, `m`, `n`, `o` ou `p`.

La correspondance des modèles est sensible à la casse, ce qui peut causer des problèmes sous Windows. Vous pouvez forcer la non sensibilité à la casse en dur en appareillant des caractères, par exemple pour ignorer `*.tmp` indépendamment de la casse, vous pourriez utiliser un modèle comme `*.[Tt][Mm][Pp]`.

Si vous voulez une définition officielle pour l'expansion de jokers (globbing), vous pouvez la trouver dans les spécifications IEEE pour le langage de commande d'interpréteur de commandes *Pattern Matching Notation* [http://www.opengroup.org/onlinepubs/009695399/utilities/xcu_chap02.html#tag_02_13].

Pas de Chemins dans la Liste des Fichiers Ignorés

Vous ne devez pas inclure de chemin d'accès dans vos patterns. Le pattern matching a pour vocation d'être utilisé à la place des noms de fichiers ou de répertoire. Si vous souhaitez ignorer tous les répertoires `CVS`, ajoutez juste `CVS` à la liste des ignorés. Il n'y a pas besoin

de spécifier `CVS */CVS` comme vous le faisiez dans les versions antérieures. Si vous souhaitez ignorer tous les répertoires `tmp` lorsqu'ils sont dans un répertoire `prog` mais pas quand ils sont dans un répertoire `doc` vous devez utiliser la propriété `svn:ignore` à la place. Il n'y a pas de manière sûre d'avoir ce type de règle de filtrage.

4.14. Supprimer, déplacer et renommer

À la différence de CVS, Subversion permet de renommer et de déplacer des fichiers et des dossiers. Il y a donc des entrées de menu pour supprimer et renommer dans le sous-menu de TortoiseSVN.

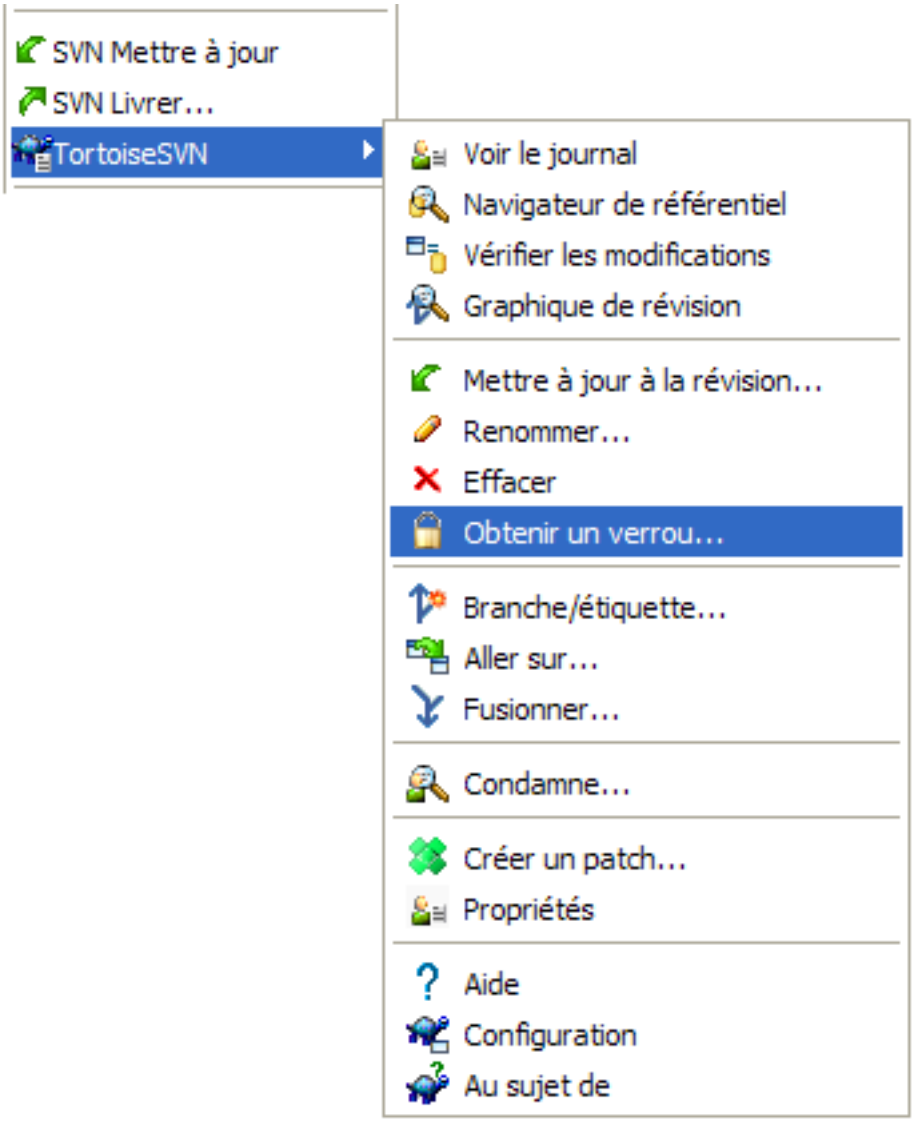


Figure 4.28. Menu contextuel de l'explorateur pour les fichiers non versionnés

4.14.1. Supprimer des fichiers et des dossiers

Utiliser TortoiseSVN → Supprimer pour effacer des fichiers ou répertoires dans subversion.

Lorsque vous utilisez la commande TortoiseSVN → Supprimer sur un fichier, il est supprimé immédiatement de votre copie de travail tout en étant marqué pour suppression dans le référentiel jusqu'au prochain envoi au dépôt. Il s'affiche sur le dossier parent du fichier un icône « supprimé ». Jusqu'à ce que

vous envoyez les modifications au dépot, vous pouvez toujours récupérer le fichier comme à l'origine si vous appelez TortoiseSVN → Revenir en arrière sur le dossier parent.

Lorsque vous utilisez la commande TortoiseSVN → Supprimer sur un répertoire, il est conservé dans votre copie de travail, mais l'icône change pour indiquer qu'il est marqué pour suppression. Jusqu'à ce que vous envoyez les modifications au référentiel, vous pouvez revenir en arrière et récupérer le dossier en faisantTortoiseSVN → Revenir en arrière sur le dossier lui-même. Cette différence de comportement entre les fichiers et dossiers est une caractéristique de Subversion, pas de TortoiseSVN.

Si vous voulez supprimer un élément dans le référentiel, mais le conserver localement comme un fichier/répertoire sans version, utilisez Menu Contextuel Avancé → Supprimer (conserver localement). Vous devez tenir la touche **Maj** enfoncée tout eeffectuant un clic droit avec la souris sur l'élément désiré dans le volet de l'explorateur (volet droit) afin de voir dans le menu contextuel avancé.

Si un *fichier* est supprimé via l'explorateur au lieu d'utiliser le menu contextuel TortoiseSVN, la boîte de dialogue Livrer affiche ces fichiers et vous laisse aussi les supprimer du contrôle de version avant la livraison. Cependant, si vous mettez à jour votre copie de travail, Subversion découvrira le fichier manquant et le remplacera par la dernière version du référentiel. Si vous devez supprimer un fichier sous contrôle de version, utilisez toujours TortoiseSVN → Supprimer pour que Subversion n'ait pas à deviner ce que vous voulez vraiment faire.

Si un *dossier* est supprimé via l'explorateur au lieu d'utiliser le menu contextuel TortoiseSVN, votre copie de travail sera cassée et vous serez incapables de livrer. Si vous mettez à jour votre copie de travail, Subversion remplacera le dossier manquant avec la dernière version du référentiel et vous pourrez alors le supprimer correctement en utilisant TortoiseSVN → Supprimer.

Récupérer un fichier ou un répertoire supprimé

Si vous avez supprimé un fichier ou un dossier et avez déjà livré cette opération de suppression au référentiel, alors un normal TortoiseSVN → Revenir en arrière normal ne peut plus le ramener. Mais le fichier ou le dossier n'est pas perdu pour autant. Si vous connaissez la révision où le fichier ou le dossier a été supprimé (sinon, utilisez la boîte de dialogue Journal pour le savoir) ouvrez l'explorateur de référentiel et allez à cette révision. Choisissez alors le fichier ou le dossier que vous avez supprimé, faites un clic-droit et choisissez Menu contextuel → Copier vers... et comme cible pour cette opération de copie, choisissez le chemin de votre copie de travail.

4.14.2. Déplacer des fichiers et des dossiers

Si vous voulez faire un simplement renommer un fichier ou un dossier, utilisez Menu contextuel → Renommer... Entrez le nouveau nom de l'objet et vous avez terminé.

Si vous voulez déplacer des fichiers à l'intérieur d'une copie de travail, peut-être vers un différent sous-répertoire, utilisez une nouvelle fois le glisser-déplacer :

1. sélectionnez les fichiers ou les répertoires que vous voulez déplacer
2. glissez-déposer avec le bouton droit ces fichiers dans le nouvel emplacement à l'intérieur de la copie de travail
3. relâchez le bouton droit de la souris
4. Dans le menu qui apparaît, sélectionnez Menu contextuel → SVN Déplacer les fichiers dans Subversion ici

Livrez le répertoire parent

Puisque les renommages et les déplacements sont gérés comme une suppression suivie d'un ajout vous devez livrer le dossier parent du fichier renommé/déplacé pour que la partie supprimée du renommage/déplacement apparaisse dans la boîte de dialogue Livrer. Si vous ne livrez pas la partie supprimée du renommage/déplacement, il restera dans le référentiel et une mise à jour par vos collègues ne supprimera pas le vieux fichier. C'est-à-dire qu'ils auront *les deux*, les vieilles et les nouvelles copies.

Vous *devez* livrer un renommage de dossier avant de changer l'un des fichiers de ce dossier, autrement votre copie de travail peut être vraiment salie.

Vous pouvez aussi utiliser le navigateur du référentiel pour déplacer des éléments.

Ne faites pas SVN Déplacer sur les externes

Vous ne devriez *pas* utiliser les commandes Déplacer ou Renommer de TortoiseSVN sur un dossier qui a été créé en utilisant `svn:externals`. Cette action causerait la suppression de l'élément externe de son référentiel parent, en dérangeant probablement beaucoup d'autres personnes. Si vous devez déplacer un dossier externe, vous devez utiliser un déplacement ordinaire, ajuster ensuite les propriétés `svn:externals` des répertoires parents de la source et de la destination.

4.14.3. Modifier la casse dans le nom d'un fichier.

Effectuer des modifications à un nom de fichier est plus délicat avec Subversion sous Windows, parce que pendant un court laps de temps lors du renommage, les deux noms de fichiers existent. Comme Windows a un cas de système de fichiers non sensible à la casse, cela ne fonctionne pas en utilisant la commande habituelle Renommer.

Heureusement, il y a (au moins) deux façons de renommer un fichier sans perdre son historique de journal. C'est important de le renommer au sein de Subversion. Le renommer seulement dans l'explorateur corrompra votre copie de travail!

Solution A) (recommandée)

1. Livrez les changements dans votre copie de travail.
2. Renommez le fichier MAJUScules en majusCULES directement dans le référentiel en utilisant l'explorateur de référentiel.
3. Mettez à jour votre copie de travail.

Solution B)

1. Renommer MAJUScules en MAJUScules_ avec la commande renommer du sous-menu TortoiseSVN.
2. Livrez les changements.
3. Renommer de MAJUScules_ en majusCULES.
4. Livrez les changements.

4.14.4. Gestion des conflits de nom de fichier.

Dans le cas où vous avez deux fichiers dans le référentiel avec le même nom mais qui ne diffèrent qu'avec la casse (par exemple `TEST.TXT` et `test.txt`), vous ne pouvez plus mettre à jour ou extraire le

répertoire parent sous Windows. Bien que Subversion supporte la casse sur les noms de fichiers, avec Windows cela n'est pas possible.

Cela arrive parfois, lorsque deux personnes envoi les modifications au dépôt, à partir de copies de travail séparées, les fichiers qui se trouvent avoir le même nom, mais avec une casse différence. Cela peut également être le cas lorsque les fichiers sont envoyés au dépôt depuis un système gérant par défaut la casse de fichiers, comme Linux.

Dans ce cas, vous devez décider lequel des deux vous voulez conserver et supprimer (ou renommer) l'autre du référentiel.

Éviter que deux fichiers aient le même nom

Il existe un script hook pour le serveur disponible à : *http://svn.collab.net/repos/svn/trunk/contrib/hook-scripts/* qui préviendra les livraisons qui résultent en conflits de casse.

4.14.5. Réparer les renommages de fichier

Parfois, votre environnement de développement va renommer des fichiers pour vous dans le cadre d'une restructuration des sources, et bien sûr il ne le dit pas à Subversion. Si vous essayez d'envoyer au dépôt vos modifications, Subversion peut voir l'ancien nom de fichier comme manquant et le nouveau pas encore versionné. Vous pourriez simplement ajouter le nouveau nom de fichier dans le référentiel, mais on perdrait alors l'historique, Subversion ne sait pas que les fichiers sont liés.

Une meilleure solution est d'informer Subversion que ce changement est en fait un renommage, et vous pouvez le faire par les boites de dialogues Livrer et Vérifier les modifications. Il suffit de sélectionner à la fois l'ancien nom (manquant) et le nouveau nom (sans version) et d'utiliser Menu contextuel → Réparation Déplacer pour lier les deux fichiers en tant qu'un renommé.

4.14.6. Supprimer les fichiers non versionnés

Généralement vous devrez placer votre liste de fichiers ignorés tel que tous les fichiers générés sont ignorés dans Subversion. Mais que faire si vous souhaitez effacer tous les éléments ignorés pour produire une génération propre? Habituellement, vous devez définir cela dans votre makefile, mais si vous débogez le makefile ou changez le système de construction, il est utile d'avoir un moyen de nettoyer la plate-forme.

TortoiseSVN fournit juste une telle option à l'aide Menu contextuel Etendu → Supprimer les éléments non versionnés.... Vous devez tenir enfoncé la touche **Maj** tout en effectuant un clic droit sur un dossier dans le volet de l'explorateur (volet droit) afin de le voir apparaitre dans le menu contextuel étendu. Cela a pour effet d'ouvrir une boîte de dialogue qui répertorie tous les fichiers non versionnés n'importe où dans votre copie de travail. Vous pouvez ensuite sélectionner ou désélectionner les éléments à enlever.

Lorsque ces éléments sont supprimés, la corbeille est utilisée, donc si vous faites une erreur ici et supprimez un fichier qui aurait dû être versionné, vous pouvez toujours le récupérer.

4.15. Annuler les changements

Si vous voulez défaire tous les changements que vous avez fait dans un fichier depuis la dernière mise à jour, vous devez sélectionner le fichier, faites un clic droit pour faire apparaître le menu contextuel et sélectionnez ensuite la commande TortoiseSVN → Revenir en arrière Une boîte de dialogue apparaîtra vous montrant les fichiers que vous avez changés et que vous pouvez restaurer. Choisissez ceux que vous voulez restaurer et cliquez sur OK.

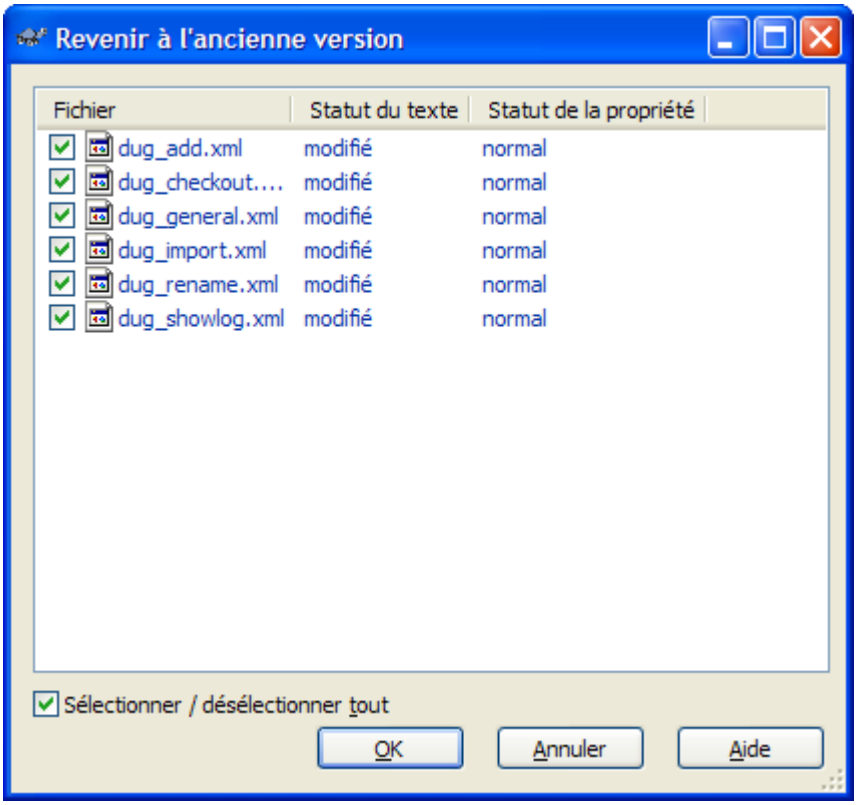


Figure 4.29. La boîte de dialogue Revenir en arrière

Si vous souhaitez annuler une suppression ou un renommage, vous devez utiliser "Revenir en arrière" sur le dossier parent car l'élément supprimé n'existe pas pour vous permettre de faire un clic droit dessus.

Si vous souhaitez annuler l'ajout d'un élément, cela apparaît dans le menu contextuel comme TortoiseSVN → Annuler l'ajout.... C'est vraiment un retour arrière, mais le nom a été changé pour le rendre plus évident.

Les colonnes dans cette boîte de dialogue peuvent être personnalisées de la même manière que les colonnes dans la boîte de dialogue Vérifier les modifications. Lisez Section 4.7.3, « Statut local et distant » pour plus de détails.

Annuler les changements qui ont été livrés

Revenir en arrière n'annulera que vos changements locaux. Cela n'annulera *pas* les changements déjà livrés. Si vous voulez annuler tous les changements livrés dans une révision spécifique, lisez Section 4.9, « La boîte de dialogue du Journal de révision » pour plus d'informations.

Le retour en arrière est Lent

Lorsque vous faites un retour arrière sur les modifications et que vous trouverez mais que l'opération prend beaucoup plus longtemps que prévu. C'est parce que la version modifiée du fichier est envoyé à la corbeille, vous pouvez donc récupérer vos modifications si vous faite un retour arrière par erreur. Toutefois, si votre corbeille est pleine, Windows prend beaucoup de temps pour trouver un endroit pour mettre le fichier. La solution est simple : soit vider la corbeille ou désactiver l'option Utiliser la corbeille lors d'un retour en arrière dans les paramètres de configuration de TortoiseSVN.

4.16. Nettoyer

Si une commande Subversion ne peut pas s'achever avec succès, peut-être en raison de problèmes serveur, votre copie de travail peut être laissée dans un état incohérent. Dans ce cas, vous devez utiliser TortoiseSVN → Nettoyer sur le dossier. C'est une bonne idée de le faire au niveau supérieur de la copie de travail.

Le nettoyage a un autre effet secondaire utile. Si une date de fichier change mais pas son contenu, Subversion ne peut dire s'il a vraiment changé qu'en faisant une comparaison octet-par-octet avec la copie primitive. Si vous avez beaucoup de fichiers dans cet état, cela rend la récupération du statut très lente, ce qui fera que beaucoup de boîtes de dialogue seront lentes à répondre. Exécuter un nettoyage sur votre copie de travail réparera ces horodatages « cassés » et rendra aux contrôles de statut leur pleine vitesse.

Utiliser les horodatages de livraison

Des sorties précédentes de Subversion ont été affectées par un bug qui a causé des incohérences dans l'horodatage quand vous extrayez avec l'option Utiliser les horodatages de livraison cochée. Utilisez la commande Nettoyer pour accélérer ces copies de travail.

4.17. Configuration des projets

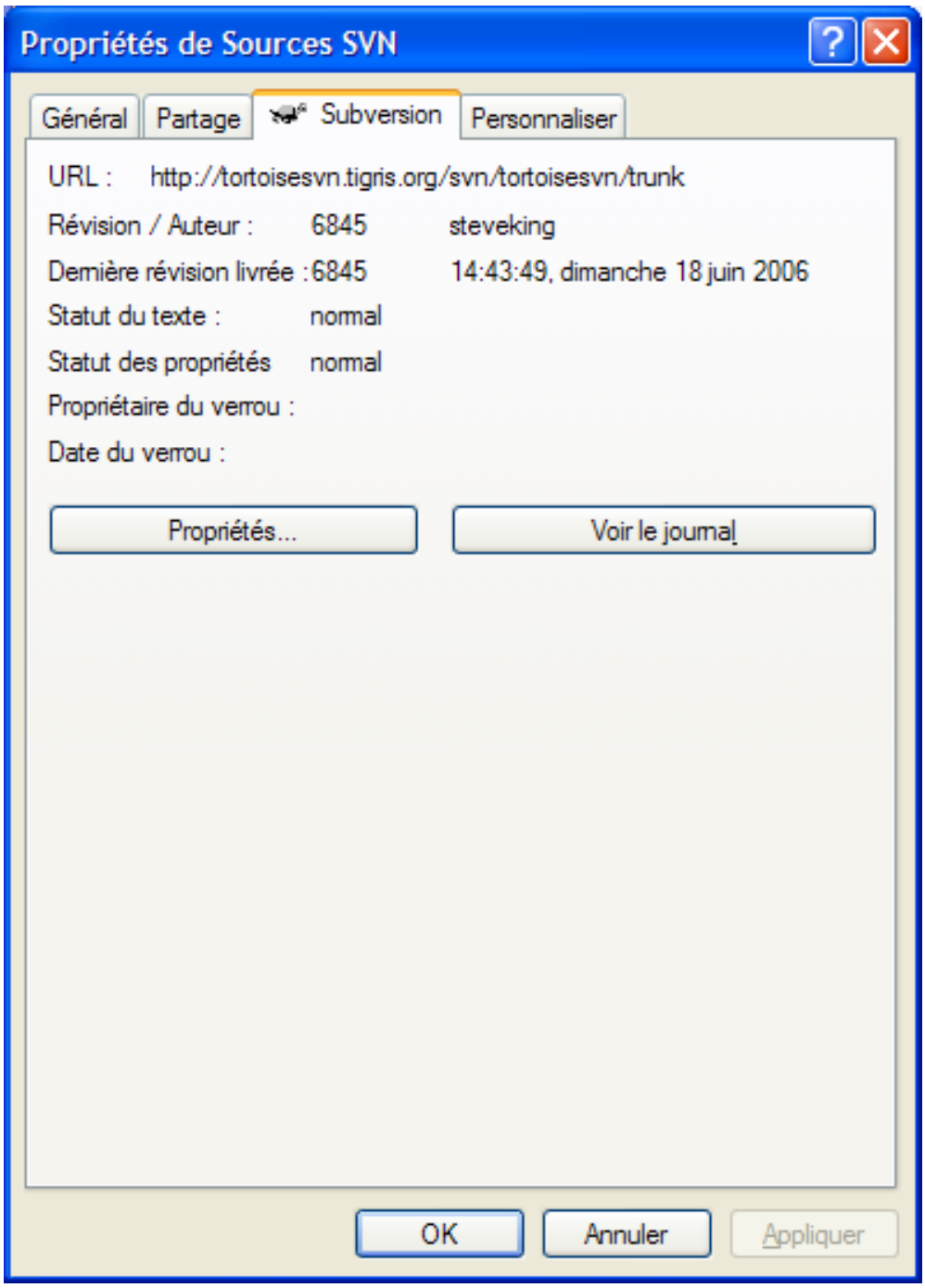


Figure 4.30. Page de propriétés de l'explorateur, onglet Subversion

Parfois vous voulez avoir des informations plus détaillées sur un fichier/répertoire qu'un recouvrement d'icône. Vous pouvez obtenir toutes les informations que Subversion fournit dans la boîte de dialogue de propriétés de l'explorateur. Sélectionnez seulement le fichier ou le répertoire et choisissez Menu Windows → Propriétés dans le menu contextuel (note : c'est l'entrée de menu de propriétés standard que l'explorateur fournit, pas celle du sous-menu TortoiseSVN !). Dans la boîte de dialogue de propriétés, TortoiseSVN a ajouté une nouvelle page de propriété pour les fichiers/dossiers sous le contrôle de Subversion où vous verrez toutes les informations pertinentes concernant le fichier/répertoire sélectionné.

4.17.1. Propriétés Subversion

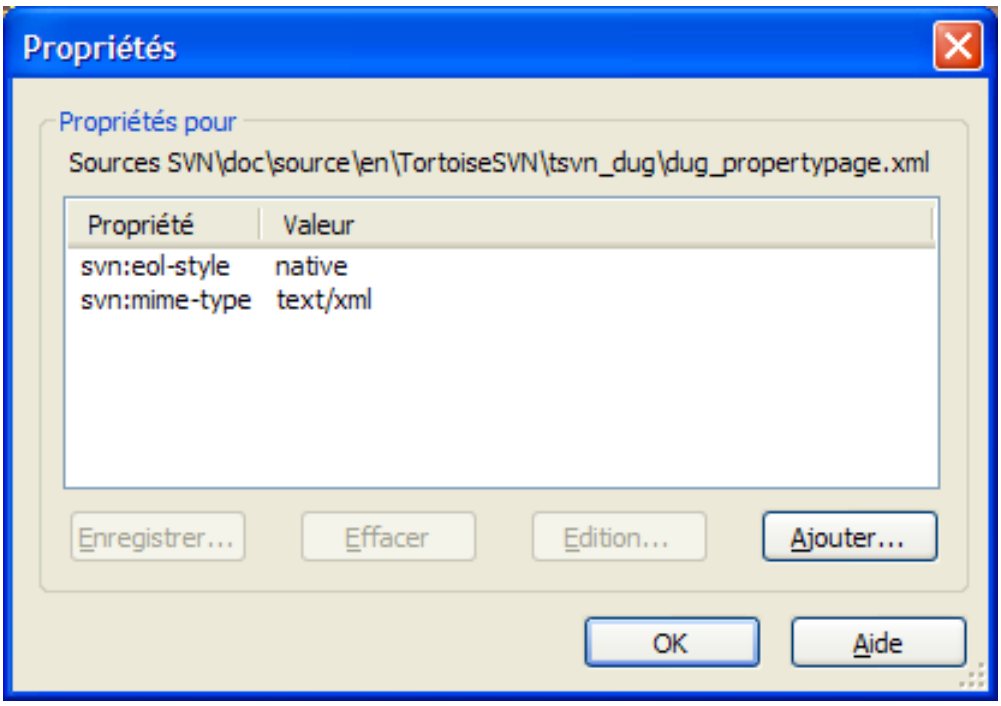


Figure 4.31. Page de propriété de subversion

You can read and set the Subversion properties from the Windows properties dialog, but also from TortoiseSVN → properties and within TortoiseSVN's status lists, from Context menu → properties.

You can add your own properties, or some properties with a special meaning in Subversion. These begin with `svn:`. `svn:externals` is such a property; see how to handle externals in Section 4.18, « Eléments externes ».

4.17.1.1. svn:keywords

Subversion supports CVS-like keyword expansion which can be used to embed filename and revision information within the file itself. Keywords currently supported are:

$Date$
: Dernière date de livraison connue. Cette information est obtenue quand vous mettez à jour votre copie de travail. Elle *ne vérifie pas* le référentiel pour trouver d'éventuels changements récents.

$Revision$
: Révision de la dernière livraison connue.

$Author$
: Auteur qui a fait la dernière livraison connue.

$HeadURL$
: Le chemin complet de ce fichier dans le référentiel.

Id
: Une combinaison raccourcie des quatres mot clés précédents.

To find out how to use these keywords, look at the *svn:keywords section* [http://svnbook.red-bean.com/en/1.5/svn.advanced.props.special.keywords.html] in the Subversion book, which gives a full description of these keywords and how to enable and use them.

For more information about properties in Subversion see the *Special Properties* [http://svnbook.red-bean.com/en/1.5/svn.advanced.props.html].

4.17.1.2. Ajouter et Modifier les propriétés

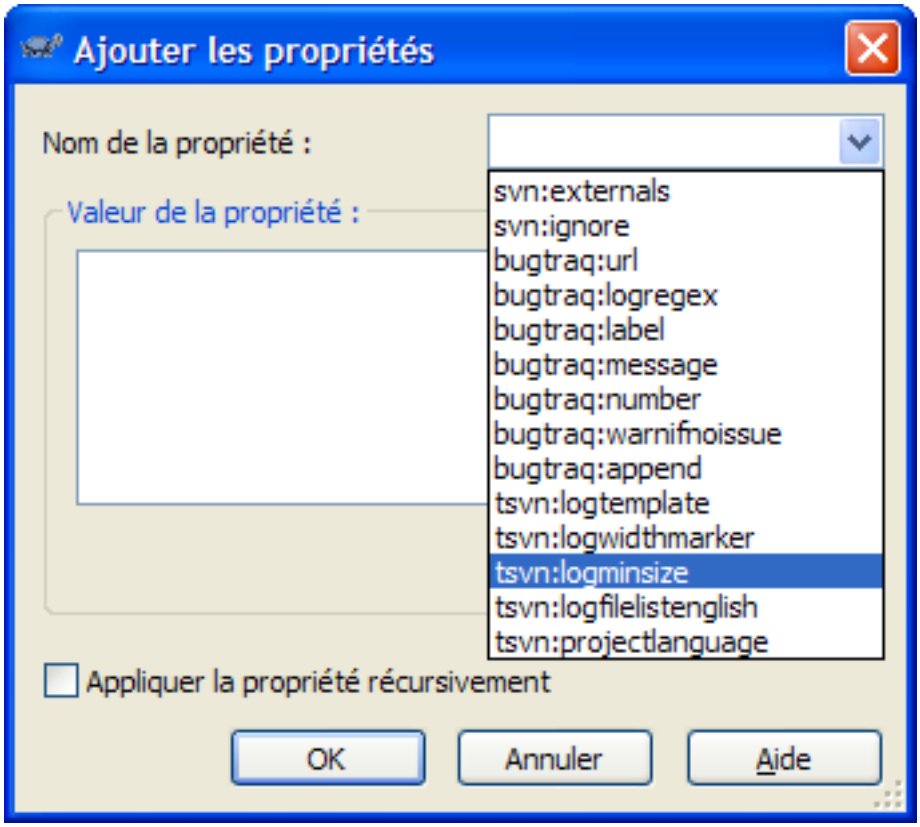


Figure 4.32. Ajouter des propriétés

Pour ajouter une nouvelle propriété, cliquez d'abord sur Ajouter.... Choisissez le nom de la propriété requise depuis la liste, ou tapez le nom de votre choix, entrez ensuite une valeur dans le champ en dessous. Les propriétés qui prennent des valeurs multiples, comme la liste des ignorés, peuvent être saisies sur plusieurs lignes. Cliquez sur OK pour ajouter la propriété à la liste.

Si vous voulez appliquer une propriété à plusieurs éléments à la fois, sélectionnez les fichiers/dossiers dans l'explorateur, puis choisissez Menu contextuel → Propriétés

Si vous voulez appliquer la propriété à *tous* les fichiers et dossiers au-dessous du dossier actuel dans la hiérarchie , cochez la case Récursive.

Certaines propriétés, par exemple `svn:needs-lock`, peuvent seulement être appliquées aux fichiers, donc le nom de la propriété n'apparaît pas dans la liste déroulante pour les dossiers. Vous pouvez toujours appliquer une telle propriété récursivement à tous les fichiers d'une hiérarchie, mais vous devez taper le nom de la propriété vous même.

Si vous voulez éditer une propriété existante, sélectionnez cette propriété dans la liste des propriétés existantes, puis cliquez sur Editer....

Si vous voulez supprimer une propriété existante, sélectionnez cette propriété dans la liste des propriétés existantes, puis cliquez sur Effacer.

The `svn:externals` property can be used to pull in other projects from the same repository or a completely different repository. For more information, read Section 4.18, « Eléments externes ».

4.17.1.3. Exporter et Importer les Propriétés

Often you will find yourself applying the same set of properties many times, for example `bugtraq:logregex`. To simplify the process of copying properties from one project to another, you can use the Export/Import feature.

From the file or folder where the properties are already set, use TortoiseSVN → properties, select the properties you wish to export and click on Export.... You will be prompted for a filename where the property names and values will be saved.

From the folder(s) where you wish to apply these properties, use TortoiseSVN → properties and click on Import.... You will be prompted for a filename to import from, so navigate to the place you saved the export file previously and select it. The properties will be added to the folders non-recursively.

If you want to add properties to a tree recursively, follow the steps above, then in the property dialog select each property in turn, click on Edit..., check the Apply property recursively box and click on OK.

The Import file format is binary and proprietary to TortoiseSVN. Its only purpose is to transfer properties using Import and Export, so there is no need to edit these files.

4.17.1.4. Propriétés

TortoiseSVN peut manipuler des valeurs de propriété binaires en utilisant des fichiers. Pour lire une valeur de propriété binaire, Enregistrer... vers un fichier. Pour mettre une valeur binaire, utilisez un éditeur hexadécimal ou un autre outil approprié pour créer un fichier avec le contenu dont vous avez besoin, puis Charger... ce fichier.

Bien que les propriétés binaires ne soient pas très utilisées, elles peuvent être utiles dans certaines applications. Par exemple si vous stockez d'énormes fichiers graphiques ou si l'application utilisée pour charger le fichier est énorme, vous pourriez vouloir stocker un aperçu en tant que propriété pour obtenir une prévisualisation rapide.

4.17.1.5. Configuration automatique des propriétés

You can configure Subversion and TortoiseSVN to set properties automatically on files and folders when they are added to the repository. There are two ways of doing this.

You can edit the subversion configuration file to enable this feature on your client. The General page of TortoiseSVN's settings dialog has an edit button to take you there directly. The config file is a simple text file which controls some of subversion's workings. You need to change two things: firstly in the section headed `miscellany` uncomment the line `enable-auto-props = yes`. Secondly you need to edit the section below to define which properties you want added to which file types. This method is a standard subversion feature and works with any subversion client. However it has to be defined on each client individually - there is no way to propagate these settings from the repository.

An alternative method is to set the `tsvn:autoprops` property on folders, as described in the next section. This method only works for TortoiseSVN clients, but it does get propagated to all working copies on update.

Whichever method you choose, you should note that auto-props are only applied to files at the time they are added to the repository. Auto-props will never change the properties of files which are already versioned.

If you want to be absolutely sure that new files have the correct properties applied, you should set up a repository pre-commit hook to reject commits where the required properties are not set.

Livrer les propriétés

Les propriétés de Subversion sont versionnées. Après avoir changé ou ajouté une propriété, vous devez livrer vos changements.

Conflits sur les propriétés

S'il y a un conflit en livrant les changements, parce qu'un autre utilisateur a changé la même propriété, Subversion génére un fichier `.prej`. Supprimez ce fichier après avoir résolu le conflit.

4.17.2. Propriétés du projet TortoiseSVN

TortoiseSVN a quelques propriétés spéciales de son cru et elles commencent par `tsvn:`.

- `tsvn:logminsize` définit la longueur minimale d'un commentaire pour une livraison. Si vous entrez un message plus court qu'indiqué ici, la livraison est désactivée. Cette fonctionnalité est très utile pour vous rappeler de fournir un message descriptif approprié à chaque livraison. Si cette propriété n'est pas définie, ou si la valeur est zéro, les commentaires vides sont autorisés.

  `tsvn:lockmsgminsize` définit la longueur minimale d'un commentaire pour un commentaire de verrou. Si vous entrez un message plus court qu'indiqué ici, le verrouillage est désctivé. Cette fonctionnalité est très utile pour vous rappeler de fournir un message descriptif approprié à chaque verrou. Si cette propriété n'est pas définie, ou si la valeur est zéro, les commentaires de verrou vides sont autorisés.

- `tsvn:logwidthmarker` est utilisé avec les projets qui exigent que les commentaires soient formatés avec une certaine largeur maximale (généralement 80 caractères) avant un saut de ligne. Définir cette propriété à une valeur non nulle fera 2 choses dans la boîte de dialogue d'entrée de commentaire : cela placera un marqueur pour indiquer la largeur maximale et cela désactivera le retour à la ligne automatique dans l'affichage, pour que vous puissiez voir si le texte que vous avez saisi est trop long. Note : cette fonction ne fonctionnera correctement que si vous avez choisi une police à largeur de caractères fixe pour les commentaires.

- `tsvn:logtemplate` est utilisé avec les projets qui ont des règles de formatage des commentaires. La propriété contient une chaîne de caractères multi-ligne qui sera insérée dans le champ de message de la livraison quand vous commencez une livraison. Vous pouvez alors l'éditer pour inclure les informations requises. Note : si vous utilisez aussi `tsvn:logminsize`, assurez-vous de définir une longueur plus longue que le modèle ou vous perdrez le mécanisme de protection.

- Subversion allows you to set « autoprops » which will be applied to newly added or imported files, based on the file extension. This depends on every client having set appropriate autoprops in their subversion configuration file. `tsvn:autoprops` can be set on folders and these will be merged with the user's local autoprops when importing or adding files. The format is the same as for subversion autoprops, e.g. `*.sh = svn:eol-style=native;svn:executable` sets two properties on files with the `.sh` extension.

  S'il y a un conflit en les propriétés autoprops locales et `tsvn:autoprops`, les propriétés du projet ont la priorité car elles lui sont spécifiques.

- In the Commit dialog you have the option to paste in the list of changed files, including the status of each file (added, modified, etc). `tsvn:logfilelistenglish` defines whether the file status is inserted in English or in the localized language. If the property is not set, the default is `true`.

- TortoiseSVN can use spell checker modules which are also used by OpenOffice and Mozilla. If you have those installed this property will determine which spell checker to use, i.e. in which language the log messages for your project should be written. `tsvn:projectlanguage` sets the language module the spell checking engine should use when you enter a log message. You can find the values for your language on this page: *MSDN: Language Identifiers* [http://msdn2.microsoft.com/en-us/library/ms776260.aspx].

  Vous pouvez saisir cette valeur en décimal, ou en hexadécimal si préfixée avec `0x`. Par exemple l'Anglais (US) peut être entré comme `0x0409` ou `1033`.

- La propriété `tsvn:logsummary` est utilisée pour extraire une partie des commentaires étant destinée à être utilisée comme résumé dans la fenêtre des commentaires.

  La valeur de la propriété `tsvn:logsummary` doit être une expression régulière d'une ligne contenant un groupe. Tout ce qui correspond à ce groupe sera utilisé comme résumé.

  Un exemple : `\[SUMMARY\]:\s+(.*)` gardera tout ce qui est après « [SUMMARY] » dans le commentaire et l'utilisera comme résumé.

- When you want to add a new property, you can either pick one from the list in the combo box, or you can enter any property name you like. If your project uses some custom properties, and you want those properties to appear in the list in the combo box (to avoid typos when you enter a property name), you can create a list of your custom properties using `tsvn:userfileproperties` and `tsvn:userdirproperties`. Apply these properties to a folder. When you go to edit the properties of any child item, your custom properties will appear in the list of pre-defined property names.

Some `tsvn:` properties require a `true/false` value. TortoiseSVN also understands `yes` as a synonym for `true` and `no` as a synonym for `false`.

TortoiseSVN can integrate with some bug tracking tools. This uses project properties that start with `bugtraq:`. Read Section 4.28, « Intégration avec des systèmes de bug tracking / traqueurs d'incidents » for further information.

It can also integrate with some web-based repository browsers, using project properties that start with `webviewer:`. Read Section 4.29, « Intégration avec des explorateur de référentiel de type web. » for further information.

Fixer les propriétés du projet dans les dossiers

These special project properties must be set on *folders* for the system to work. When you commit a file or folder the properties are read from that folder. If the properties are not found there, TortoiseSVN will search upwards through the folder tree to find them until it comes to an unversioned folder, or the tree root (eg. `C:\`) is found. If you can be sure that each user checks out only from e.g `trunk/` and not some sub-folder, then it is sufficient to set the properties on `trunk/`. If you can't be sure, you should set the properties recursively on each sub-folder. A property setting deeper in the project hierarchy overrides settings on higher levels (closer to `trunk/`).

For project properties *only* you can use the Recursive checkbox to set the property to all sub-folders in the hierarchy, without also setting it on all files.

Lorsque vous ajoutez un nouveau sous répertoire via TortoiseSVN, toutes les propriétés du projet présentes dans le répertoire parent seront automatiquement ajoutées au nouveau sous répertoire.

Attention

Although TortoiseSVN's project properties are extremely useful, they only work with TortoiseSVN, and some will only work in newer versions of TortoiseSVN. If people working on your project use a variety of Subversion clients, or possibly have old versions of TortoiseSVN, you may want to use repository hooks to enforce project policies. project properties can only help to implement a policy, they cannot enforce it.

4.18. Eléments externes

Il est parfois utile de construire une copie de travail faite d'un certain nombre d'extractions différentes. Par exemple, vous pouvez vouloir que des fichiers ou des sous-répertoires différents viennent d'emplacements

différents dans un référentiel, ou peut-être même de référentiels différents. Si vous voulez que chaque utilisateur ait la même disposition, vous pouvez définir les propriétés `svn:externals` de façon à positionner les ressources spécifiées dans les emplacements cibles souhaités.

4.18.1. Répertoires externes

Si vous souhaitez extraire une copie de travail du `/projet1` dans `D:\dev\projet1`. Sélectionnez le dossier `D:\dev\projet1`, faites un click droit et choisissez Menu Windows → Propriétés dans le menu contextuel. La boîte de dialogue Propriétés apparaît. Allez alors dans l'onglet Subversion. Il vous permet de modifier les propriétés. Cliquez Ajouter.... Choisissez la propriété `svn:externals` dans la liste déroulante et écrivez dans la boîte de saisie l'URL du référentiel au format `url dossier` ou si vous voulez spécifier une révision particulière, `-rREV url dossier`. Vous pouvez ajouter plusieurs projets externes, 1 par ligne. Supposons que vous ayez saisi les propriétés suivantes sur `D:\dev\projet1` :

```
http://sounds.red-bean.com/repos sounds
http://graphics.red-bean.com/repos/fast%20graphics "quick_graphs"
-r21 http://svn.red-bean.com/repos/skin-maker  skins/toolkit
```

Cliquez ensuite sur Définir et livrez vos changements. Quand vous (ou tout autre utilisateur) mettez à jour votre copie de travail, Subversion créera un sous-dossier `D:\dev\projet1\sounds` et extraira le projet des sons, un autre sous-dossier `D:\dev\projet1\quick graphs` contenant le projet des graphiques et finalement un sous-dossier imbriqué `D:\dev\projet1\skins\toolkit` contenant la révision 21 du projet de fabrication des thèmes.

Notez que les URL doivent être correctement saisies ou elles ne fonctionneront pas. Par exemple vous devez remplacer chaque espace par `%20`, comme dans le second exemple ci-dessus.

Si vous souhaitez que le chemin local contienne des espaces ou d'autres caractères spéciaux, vous pouvez les encadrer avec des doubles cotes, ou vous pouvez utiliser le caractère `\` (backslash) comme un type de caractère shell Unix pour distinguer tout caractère spécial qu'il précède. Bien entendu, cela signifie que vous devez utiliser `/` (slash) comme délimiteur de chemin. Notez que ce comportement est nouveau dans Subversion 1.6 et ne fonctionnera pas avec les versions antérieures.

Utiliser des numéros de révision explicites

Vous devriez vous astreindre à l'utilisation de numéros de révision explicites dans toutes vos définitions externes, comme décrit ci-dessus. Cela revient à décider quand choisir un instantané différent d'informations externes et quel instantané particulier intégrer. En plus du bon sens consistant à ne pas être pas étonné par les changements aux référentiels tiers sur lesquels vous pourriez n'avoir aucun contrôle, utiliser des numéros de révision explicites signifie aussi que lorsque vous ramenez votre copie de travail à une révision précédente, vos définitions externes réintègrent également l'état qu'elles avaient dans cette révision précédente, ce qui signifie en conséquence que les copies de travail externes seront mises à jour pour correspondre à l'état qu'*elles* avaient quand votre référentiel était à cette révision précédente. Pour des projets logiciels, ce pourrait être la différence entre une réussite et un échec lors de la construction d'un instantané antérieur de votre complexe base de code.

définitions svn:externals plus anciennes

Le format présenté ici a été introduit dans Subversion 1.5. Vous pouvez également rencontrer l'ancien format, constitué des mêmes informations dans un ordre différent. Le nouveau format est préféré, car il supporte plusieurs fonctionnalités utiles décrites ci-dessous, mais il ne fonctionne pas avec les clients des versions précédentes. Les différences sont présentées dans le *Subversion Book* [http://svnbook.red-bean.com/en/1.5/svn.advanced.externals.html].

Si le projet externe est dans le même référentiel, les changements que vous y faites seront inclus dans la liste de livraisons quand vous livrerez votre projet principal.

Si le projet externe est dans un référentiel différent, les changements que vous faites au projet externe seront signalés quand vous livrerez le projet principal, mais vous devez livrer ces changements externes séparément.

Si vous utilisez des URL absolues dans la valeur de l'option `svn:externals` and que vous déplacez votre copie de travail (i.e., si l'URL de votre référentiel change), alors vos références externes ne seront pas mises à jour et ne fonctionneront plus.

Pour éviter de tels problèmes, à partir de la version 1.5 du client, Subversion permet d'utiliser les chemins relatifs dans les références externes. Quatre méthodes différentes de spécifications d'URL relatives sont supportées. Dans les exemples suivants, considérez qu'il y a deux référentiels: un premier ici `http://exemple.com/svn/repos-1` et un second là `http://exemple.com/svn/repos-2`. Nous avons une extraction de `http://exemple.com/svn/repos-1/project/trunk` dans `C:\Working` et la propriété `svn:externals` est activée sur le répertoire trunk.

Relatif au répertoire parent.
: Ces URL commencent toujours par la chaine de caractère `../` par exemple:

```
../../widgets/foo  common/foo-widget
```

Extraira `http://exemple.com/svn/repos-1/widgets/foo` dans `C:\Working\common\foo-widget`.

Notez que l'URL est relative à l'URL du répertoire ayant la propriété `svn:externals`, pas à l'emplacement physique où le répertoire ayant la référence externe est stocké.

Relatif à la racine du référentiel
: Ces URLs commencent toujours par la chaine de caractère `^/` par exemple:

```
^/widgets/foo  common/foo-widget
```

Extraira `http://exemple.com/svn/repos-1/widgets/foo` dans `C:\Working\common\foo-widget`.

Vous pouvez facilement faire référence à d'autres référentiels ayant le même `SVNParentPath` (un répertoire commun hébergeant plusieurs référentiels). Par exemple:

```
^/../repos-2/hammers/claw  common/claw-hammer
```

Extraira `http://exemple.com/svn/repos-2/hammers/claw` dans `C:\Working\common\claw-hammer`.

Relatif au thème
: Les URL qui commencent par `//` ne comportent que la partie "scheme" de l'URL. C'est utile quand l'hôte est accédé de différentes manières selon l'emplacement du client ; i.e. on utilise `http://` depuis l'intranet alors que le protocole `svn+ssh://` est nécessaire depuis l'extérieur. Par exemple :

```
//exemple.com/svn/repos-1/widgets/foo  common/foo-widget
```

Extraira `http://exemple.com/svn/repos-1/widgets/foo` ou `svn+ssh://exemple.com/svn/repos-1/widgets/foo` selon la méthode utilisée pour faire la livraison de `C:\Working`.

Relatif au nom de domaine du serveur
: Les URL qui commencent par la chaine `/` ne contiennent que la partie "scheme" et le nom d'hôte de l'URI, par exemple :

```
/svn/repos-1/widgets/foo  common/foo-widget
```

Extraira `http://exemple.com/svn/repos-1/widgets/foo` dans `C:\Working\common\foo-widget`. Mais si vous extrayez votre copie de travail depuis un autre serveur `svn+ssh://autre.mirroir.net/svn/repos-1/projet1/trunk` alors la référence externe extraira `svn+ssh://autre.mirroir.net/svn/repos-1/widgets/foo`.

Vous pouvez également précisé une révision "peg" après l'URL, si nécessaire, par exemple: `http://sounds.red-bean.com/repos@19`.

Si vous avez besoin de plus d'informations sur la façon dont TortoiseSVN manipule les propriétés, lisez Section 4.17, « Configuration des projets ».

Pour découvrir des méthodes différentes d'accéder aux sous-projets communs, lisez Section B.6, « Inclure un sous-projet commun ».

4.18.2. Fichiers externes

A partir de Subversion 1.6 vous pouvez ajouter des références externes relatives à des fichiers en utilisant la même syntaxe que pour les dossiers. Il y a néanmoins quelques limitations.

- Le chemin vers le fichier en référence externe doit placer le fichier dans un dossier existant versionné. En général, il est logique de placer le fichier directement dans le dossier dont la propriété `svn:externals` est activée, mais cela peut être dans un sous-dossier versionné si nécessaire. A l'inverse, les répertoires externes vont créer automatiquement tout dossier intermédiaire non versionné requis.

- L'URL pour un fichier en référence externe doit se trouver dans le même référentiel que l'URL dans laquelle le fichier externe sera inséré; les fichiers en référence externe inter-référentiel ne sont pas supportés.

Par beaucoup d'aspects, les fichiers en référence externe se comportent comme n'importe quel autre fichier versionné. Néanmoins, ils ne peuvent pas être déplacés ou supprimés avec les commandes standards; il faut passer par la modification de la propriété `svn:externals`.

Le support des fichiers en référence externe est incomplet dans Subversion 1.6

Dans Subversion 1.6, il n'est pas possible de retirer un fichier en référénce externe de votre copie de travail une fois que vous l'y avez ajouté, même si vous supprimer complètement la propriété `svn:externals`. Vous devez extraire une nouvelle copie de travail pout retirer le fichier.

4.19. Brancher / Étiqueter

La capacité d'isoler des changements sur une ligne de développement séparée est une des fonctionnalités des systèmes de contrôle de version. Cette ligne est connue comme une *branche*. Les branches sont souvent utilisées pour expérimenter de nouvelles fonctionnalités sans déranger la ligne de développement principale avec des erreurs de compilation et des bugs. Dès que la nouvelle fonction est assez stable alors la branche de développement est *fusionnée* vers la branche principale (le tronc).

Une autre fonctionnalité des systèmes de contrôle de version est la capacité de marquer des révisions particulières (par exemple une version déployée), donc vous pouvez à tout moment recréer une certaine version de votre application ou un environnement. Ce processus est connu comme l'*étiquetage*.

Subversion n'a pas de commandes spéciales pour créer des branches ou des étiquettes, mais utilise de prétendues `copies bon marché` à la place. Les copies bon marché sont semblables aux hard links d'Unix, ce qui signifie qu'au lieu de faire une copie complète dans le référentiel, un lien interne est créé, pointant vers un arbre/révision spécifique. En conséquence, branches et étiquettes sont très rapides à créer et ne prennent presque aucun espace supplémentaire dans le référentiel.

4.19.1. Créer une branche ou une étiquette

Si vous avez importé votre projet avec la structure de répertoire recommandée, créer une branche ou une étiquette est très simple :

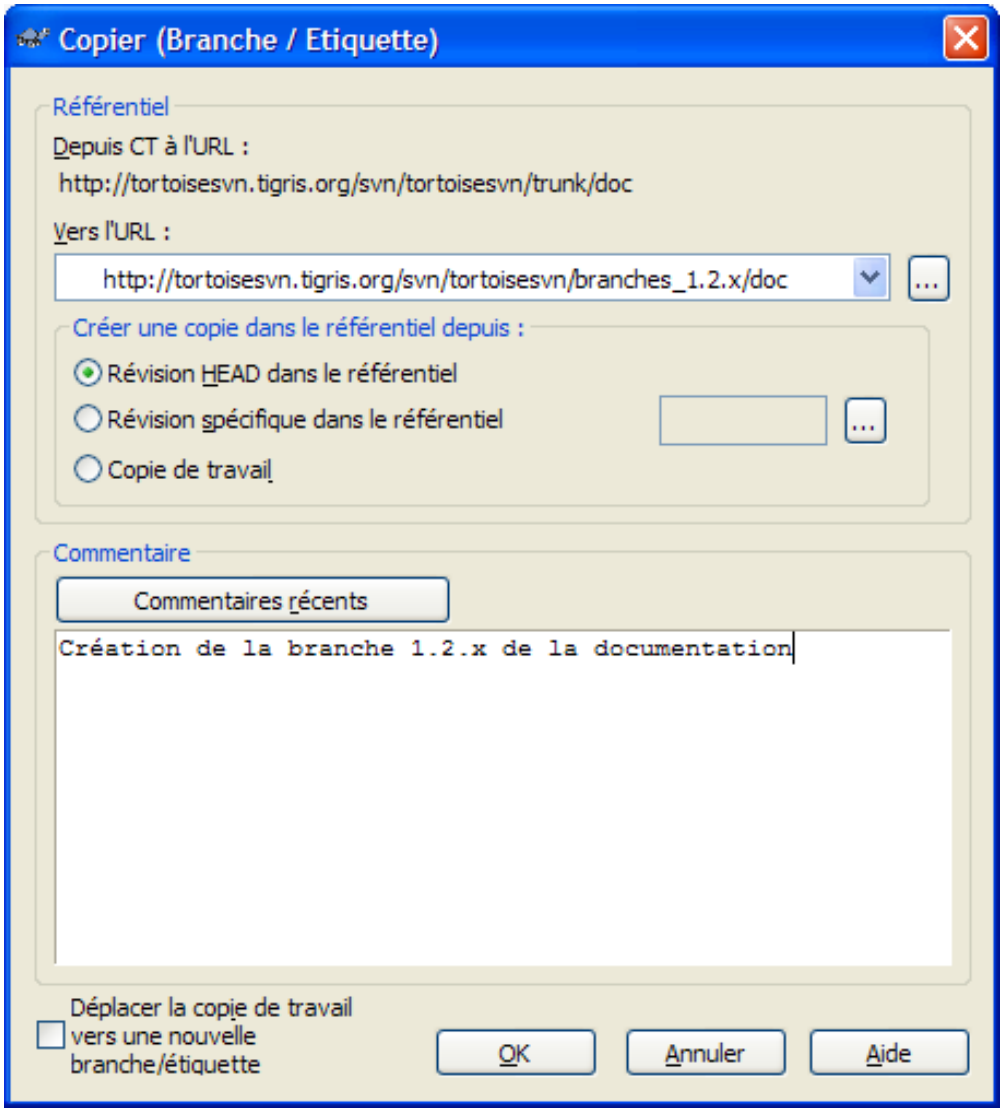


Figure 4.33. La boîte de dialogue Branche/Etiquette

Sélectionnez le dossier de votre copie de travail que vous voulez copier dans une branche ou une étiquette, choisissez ensuite la commande TortoiseSVN → Branche/Étiquette....

L'URL de destination par défaut de la nouvelle branche sera l'URL source sur laquelle votre copie de travail est basée. Vous devrez éditer cette URL vers le nouveau chemin de votre branche/étiquette. Ainsi au lieu de

```
http://svn.collab.net/repos/ProjectName/trunk
```

vous pourriez maintenant utiliser quelque chose comme

```
http://svn.collab.net/repos/ProjectName/tags/Release_1.10
```

Si vous ne pouvez pas vous rappeler la convention de nommage que vous avez utilisée la dernière fois, cliquer sur le bouton à droite pour ouvrir l'explorateur de référentiel pour voir la structure du référentiel existante.

Maintenant vous devez choisir la source de la copie. Ici vous avez trois options :

Révision HEAD dans le référentiel
: La nouvelle branche est copiée directement dans le référentiel de la révision HEAD. Aucune donnée n'a besoin d'être transférée depuis votre copie de travail et la branche est créée très rapidement.

Révision spécifique dans le référentiel
: La nouvelle branche est copiée directement dans le référentiel mais vous pouvez choisir une révision plus vieille. C'est utile si vous avez oublié de faire une étiquette quand vous avez sorti votre projet la semaine dernière. Si vous ne pouvez pas vous rappeler le numéro de révision, cliquez sur le bouton à droite pour afficher le journal de révision et choisir le numéro de révision de là. Là encore, aucune donnée n'est transférée de votre copie de travail et la branche est créée très rapidement.

Copie de travail
: La nouvelle branche est une copie identique à votre copie de travail locale. Si vous avez mis à jour quelques fichiers à une révision plus ancienne dans votre CdT, ou si vous avez fait des changements locaux, c'est exactement ceux-ci qui vont dans la copie. Naturellement cette sorte d'étiquette complexe peut impliquer des transferts de données de votre CdT vers le référentiel si elles n'y existent pas déjà.

Si vous voulez que votre copie de travail soit basculée automatiquement sur la branche nouvellement créée, utilisez la case à cocher Déplacer la copie de travail vers une nouvelle branche/étiquette. Mais si vous le faites, assurez-vous d'abord que votre copie de travail ne contient pas de modifications. Si c'est le cas, ces changements seront fusionnés dans la CdT de branche quand vous basculerez.

Appuyez sur OK pour livrer la nouvelle copie dans le référentiel. N'oubliez pas de mettre un commentaire. Notez que la copie est créée *dans le référentiel*.

Notez qu'à moins que vous choisissiez de basculer votre copie de travail vers la branche juste créée, créer une branche ou une étiquette *ne modifie pas* votre copie de travail. Même si vous créez la branche depuis votre CdT, ces changements sont livrés dans la nouvelle branche, pas dans le tronc, donc votre CdT peut toujours être marquée comme modifiée sans altérer le tronc.

4.19.2. Extraire ou aller sur...

... telle (n')est (pas) la question. Tandis qu'une extraction extraie tout de la branche désirée dans votre répertoire de travail, TortoiseSVN → Aller sur... transfère seulement les données changées dans votre copie de travail. Bon pour la charge réseau, bon pour votre patience. :-)

Pour pouvoir travailler avec votre branche ou votre tag récemments générés, vous avez plusieurs méthodes. Vous pouvez :

- TortoiseSVN → Extraire pour faire une extraction récente dans un dossier vide. Vous pouvez extraire à n'importe quel emplacement sur votre disque local et vous pouvez créer autant de copies de travail de votre référentiel que vous souhaitez.

- Basculez votre copie de travail courante vers la copie nouvellement créée dans le référentiel. Choisissez de nouveau le dossier de niveau supérieur de votre projet et utilisez TortoiseSVN → Aller sur... du menu contextuel.

  Dans la boîte de dialogue suivante, entrez l'URL de la branche que vous venez juste de créer. Choisissez le bouton radio Révsion HEAD et cliquez sur OK. Votre copie de travail est basculée vers la nouvelle branche/étiquette.

  Aller sur... fonctionne comme Mettre à jour dans le sens où il ne se débarrasse jamais de vos changements locaux. Les changements que vous avez faits à votre copie de travail qui n'ont pas encore été livrés seront fusionnés quand vous faites Aller sur. Si vous ne voulez pas que cela arrive alors vous devez ou livrer les changements avant la bascule, ou faire revenir votre copie de travail à une révision déjà livrée (typiquement HEAD).

- Si vous voulez travailler sur le tronc et une branche, mais sans faire une extraction, vous pouvez utiliser Windows Explorer pour copier votre extraction du tronc dans un autre répertoire, puis utilisez la commande TortoiseSVN → Aller sur... sur cette copie pour en faire la branche.

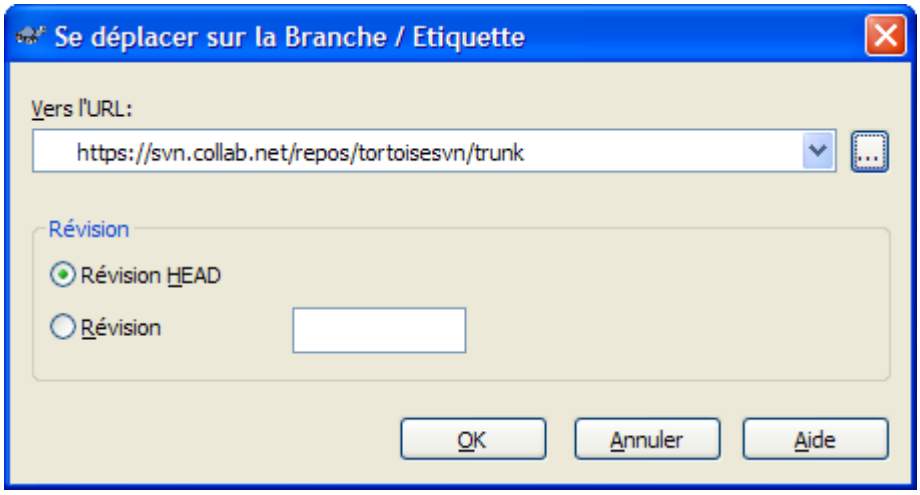


Figure 4.34. La boîte de dialogue Aller sur

Bien que Subversion lui-même ne fasse aucune distinction entre les étiquettes et les branches, la manière dont elles sont typiquement utilisées diffère un peu.

- Les étiquettes sont typiquement utilisées pour garder un état figé du projet à une étape particulière. Ils ne sont normalement pas utilisés comme tel pour le développement - contrairement aux branches, c'est la raison pour laquelle nous avons recommandé la structure de référentiel `/trunk /branches /tags` en premier lieu. Travailler sur une révision d'étiquette *n'est pas une bonne idée*, mais dans la mesure où vos fichiers locaux ne sont pas protégés en écriture, il n'y a rien pour vous en empêcher. Cependant, si vous essayez de livrer vers un chemin dans le référentiel qui contient `/tags/`, TortoiseSVN vous avertira.

- Il peut arriver que vous deviez faire de nouveaux changements à une version déployée que vous avez déjà étiquetée. La façon correcte de le gérer est de créer d'abord une nouvelle branche à partir de l'étiquette et de livrer cette branche. Faites vos changements sur cette branche et créez ensuite une nouvelle étiquette depuis cette nouvelle branche, par exemple `Version_1.0.1`.

- Si vous modifiez une copie de travail créée à partir d'une branche et livrez, alors tous les changements seront luvrés sur la nouvelle branche et *pas* sur le tronc. Seules les modifications sont stockées. Le reste reste une copie peu coûteuse.

4.20. Fusionner

Quand les branches sont utilisées pour maintenir des lignes séparées de développement, à une certaine étape vous voudrez fusionner les changements faits sur une branche vers le tronc, ou vice versa.

Il est important de comprendre comment les branches et la fusion fonctionnent dans Subversion avant de les utiliser, car cela peut devenir assez complexe. Il est fortement recommandé de lire le chapitre *Branching and Merging* [http://svnbook.red-bean.com/en/1.5/svn.branchmerge.html] dans le manuel de Subversion, lequel donne une description complète et beaucoup d'exemples d'utilisation.

Il faut aussi noter que fusionner arrive *toujours* à l'intérieur d'une copie de travail. Si vous voulez fusionner les modifications *dans* une branche, vous devez avoir une copie de travail pour la branche extraite, et appeler l'assistant de fusion depuis cette copie de travail en utilisant TortoiseSVN → Fusionner....

En général, c'est une bonne idée d'exécuter une fusion dans une copie de travail inchangée. Si vous avez fait d'autres changements dans votre CdT, livrez les d'abord. Si la fusion ne se déroule pas comme prévu, vous pouvez vouloir annuler les changements et la commande Revenir en arrière supprimera *tous* les changements en incluant ceux effectués avant la fusion.

Il y a trois cas d'utilisation de la fusion qui sont gérés de façons légèrement différentes, comme décrit ci-dessous. La première page de l'assistant de fusion vous demande de quelle méthode vous avez besoin.

Fusionner une plage de révisions
: Cette méthode couvre le cas où vous avez fait une ou plusieurs révisions sur une branche (ou sur le tronc) et vous voulez reporter ces changements vers une autre branche.

  Ce que vous demandez à Subversion est de : « Calculer les modifications nécessaires pour passer [DE] la révision 1 de la branche A [A] la révision 7 de la branche A, et d'appliquer ces modifications à la copie de travail (de la racine ou de la branche B). »

Réintégrer une branche
: Cette méthode gère le cas où vous avez développé une nouvelle fonctionnalité dans une branche comme conseillé dans le manuel de Subversion. Semaine après semaine, tous les changements du tronc ont été reportés dans cette branche, et maintenant que la fonctionnalité est terminée, vous voulez l'intégrer dans le tronc. Vu que vous avez gardé la branche synchronisée avec le tronc, les dernières versions de la branche et du tronc seront absolument identiques à part vos modifications de la branche.

  C'est un cas particulier de fusion de l'arborescence comme décrite plus haut, et (normalement) seule l'URL des sources à fusionner avec votre branche de développement est nécessaire.Cette fusion utilise la fonctionnalité deIt uses the merge-tracking features of Subversion to calculate the correct revision ranges to use, and perform additional checks which ensure that the branch has been fully updated with trunk changes.Cette technique empêche de défaire du that you don't accidentally undo work that others have committed to trunk since you last synchronized changes.

  Après la fusion, toutes les branches de développement ont été complètement fusionnées avec la version de tête. La branches est donc redondante et peut être supprimée.

  Once you have performed a reintegrate merge you should not continue to use it for development. The reason for this is that if you try to resynchronize your existing branch from trunk later on, merge tracking will see your reintegration as a trunk change that has not yet been merged into the branch, and will try to merge the branch-to-trunk merge back into the branch! The solution to this is simply to create a new branch from trunk to continue the next phase of your development.

Fusionner deux arborescences différentes
: This is a more general case of the reintegrate method. What you are asking Subversion to do is: « Calculate the changes necessary to get [FROM] the head revision of the trunk [TO] the head revision of the branch, and apply those changes to my working copy (of the trunk). » The net result is that trunk now looks exactly like the branch.

  If your server/repository does not support merge-tracking then this is the only way to merge a branch back to trunk. Another use case occurs when you are using vendor branches and you need

to merge the changes following a new vendor drop into your trunk code. For more information read the chapter on *vendor branches* [http://svnbook.red-bean.com/en/1.5/svn.advanced.vendorbr.html] in the Subversion Book.

4.20.1. Fusionner une plage de révisions

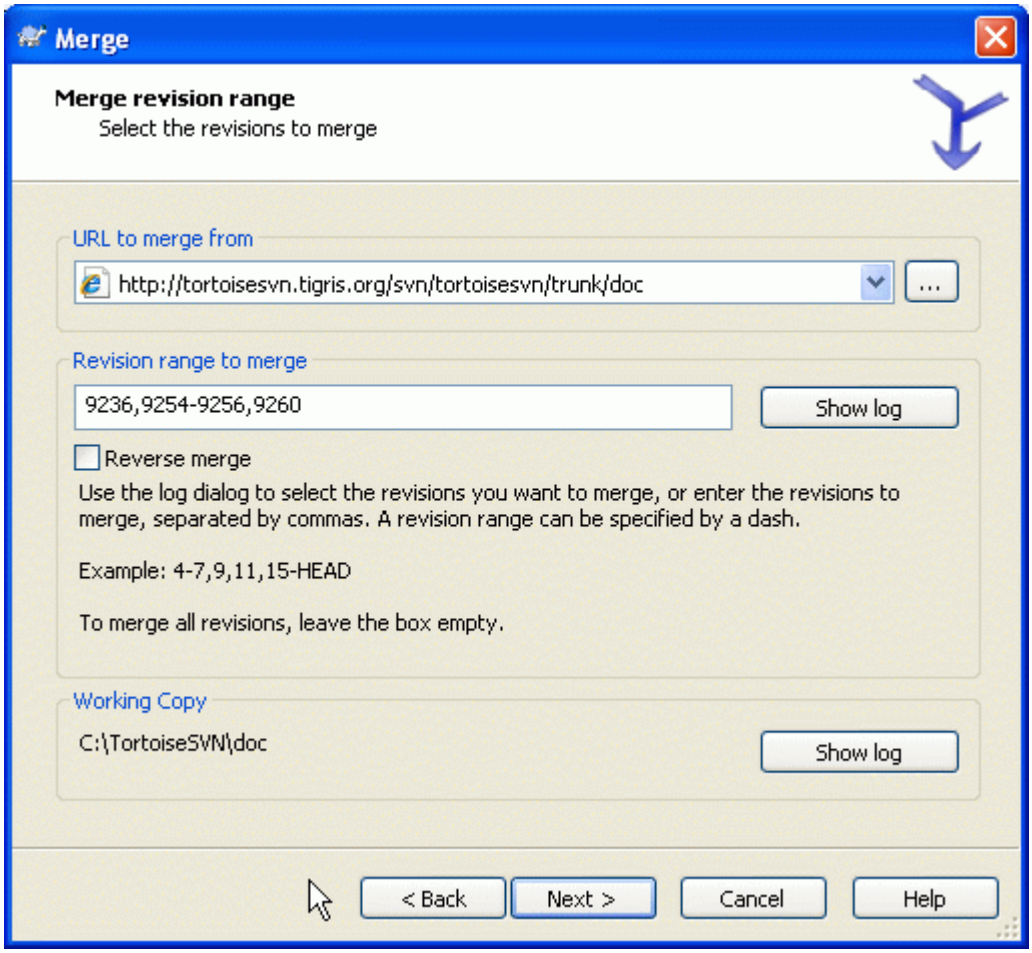


Figure 4.35. Assistant de fusion - Sélectionner une gamme de révisions

In the From: field enter the full folder URL of the branch or tag containing the changes you want to port into your working copy. You may also click ... to browse the repository and find the desired branch. If you have merged from this branch before, then just use the drop down list which shows a history of previously used URLs.

In the Revision range to merge field enter the list of revisions you want to merge. This can be a single revision, a list of specific revisions separated by commas, or a range of revisions separated by a dash, or any combination of these.

Important

There is an important difference in the way a revision range is specified with TortoiseSVN compared to the command line client. The easiest way to visualise it is to think of a fence with posts and fence panels.

Avec le client en ligne de commande vous spécifiez les modifications à fusionner avec deux révisions « limites » qui spécifient les bornes *avant* et *après*.

> With TortoiseSVN you specify the changeset to merge using « fence panels ». The reason for this becomes clear when you use the log dialog to specify revisions to merge, where each revision appears as a changeset.
>
> If you are merging revisions in chunks, the method shown in the subversion book will have you merge 100-200 this time and 200-300 next time. With TortoiseSVN you would merge 100-200 this time and 201-300 next time.
>
> This difference has generated a lot of heat on the mailing lists. We acknowledge that there is a difference from the command line client, but we believe that for the majority of GUI users it is easier to understand the method we have implemented.

The easiest way to select the range of revisions you need is to click on Show Log, as this will list recent changes with their log comments. If you want to merge the changes from a single revision, just select that revision. If you want to merge changes from several revisions, then select that range (using the usual **Shift**-modifier). Click on OK and the list of revision numbers to merge will be filled in for you.

Si vous voulez *revenir* sur des modifications déjà livrées de votre copie de travail, sélectionnez les révisions à annuler et vérifiez bien que la case Fusion inversée est cochée.

If you have already merged some changes from this branch, hopefully you will have made a note of the last revision merged in the log message when you committed the change. In that case, you can use Show Log for the Working Copy to trace that log message. Remembering that we are thinking of revisions as changesets, you should Use the revision after the end point of the last merge as the start point for this merge. For example, if you have merged revisions 37 to 39 last time, then the start point for this merge should be revision 40.

If you are using the merge tracking features of Subversion, you do not need to remember which revisions have already been merged - Subversion will record that for you. If you leave the revision range blank, all revisions which have not yet been merged will be included. Read Section 4.20.6, « Suivi des fusions » to find out more.

Si d'autres personnes peuvent livrer des changements alors soyez prudents en utilisant de la révision HEAD. Elle peut ne pas faire référence à la révision à laquelle vous pensez si quelqu'un d'autre a fait une livraison après votre dernière mise à jour.

Cliquez le bouton Suivant et allez à Section 4.20.4, « Options de fusion »

4.20.2. Réintégrer une branche

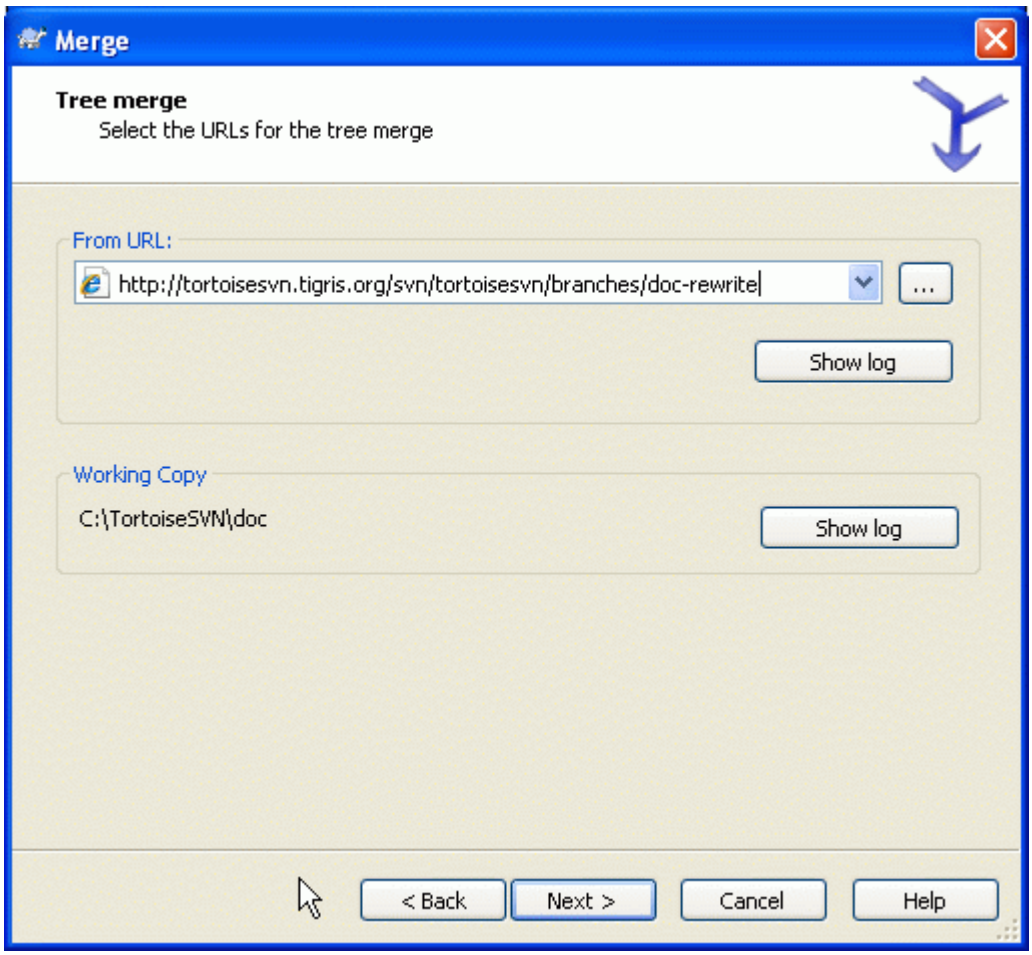


Figure 4.36. Assistant de fusion - Réintégration

To merge a feature branch back into the trunk you must start the merge wizard from within a working copy of the trunk.

In the From URL: field enter the full folder URL of the branch that you want to merge back. You may also click ... to browse the repository.

There are some conditions which apply to a reintegrate merge. Firstly, the server must support merge tracking. The working copy must be of depth infinite (no sparse checkouts), and it must not have any local modifications, switched items or items that have been updated to revisions other than HEAD. All changes to trunk made during branch development must have been merged across to the branch (or marked as having been merged). The range of revisions to merge will be calculated automatically.

4.20.3. Fusionner deux arbres différents

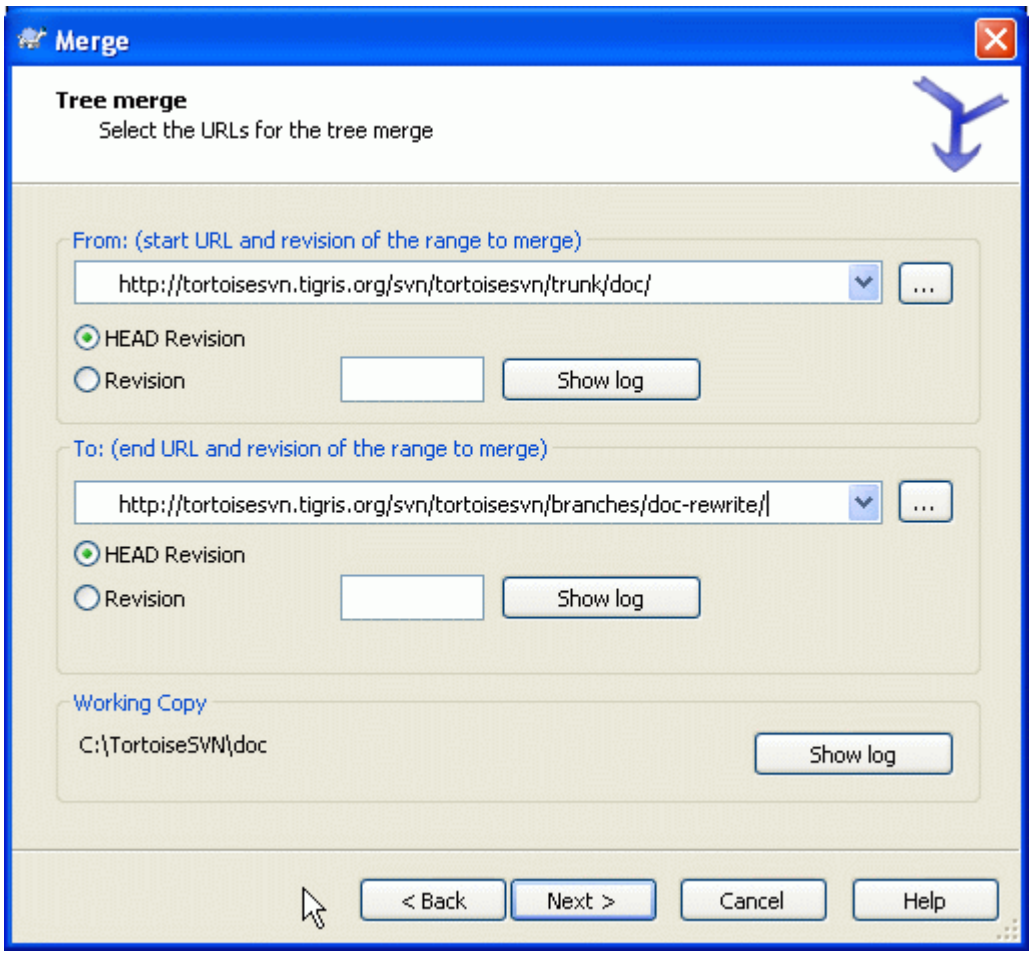


Figure 4.37. Assitant de de fusion - Fusion d'arborescence

If you are using this method to merge a feature branch back to trunk, you need to start the merge wizard from within a working copy of trunk.

In the From: field enter the full folder URL of the *trunk*. This may sound wrong, but remember that the trunk is the start point to which you want to add the branch changes. You may also click ... to browse the repository.

Remplissez le champ To: par l'adresse complète de la branche.

Dans les deux champs Depuis la révision et À la révision, entrez le numéro de la dernière révision à laquelle les deux arbres ont été synchronisés. Si vous êtes sûrs que personne d'autre ne fait de livraisons vous pouvez utiliser la révision HEAD dans les deux cas. S'il y a une chance que quelqu'un d'autre puisse avoir fait une livraison depuis cette synchronisation, utilisez le numéro de la révision particulière pour éviter pour perdre des livraisons plus récentes.

Vous pouvez également utiliser Voir les Messages de Log pour sélectionner la révision.

4.20.4. Options de fusion

This page of the wizard lets you specify advanced options, before starting the merge process. Most of the time you can just use the default settings.

You can specify the depth to use for the merge, i.e. how far down into your working copy the merge should go. The depth terms used are described in Section 4.3.1, « Profondeur d'extraction ». The default depth is Working copy, which uses the existing depth setting, and is almost always what you want.

Most of the time you want merge to take account of the file's history, so that changes relative to a common ancestor are merged. Sometimes you may need to merge files which are perhaps related, but not in your repository. For example you may have imported versions 1 and 2 of a third party library into two separate directories. Although they are logically related, Subversion has no knowledge of this because it only sees the tarballs you imported. If you attempt to merge the difference between these two trees you would see a complete removal followed by a complete add. To make Subversion use only path-based differences rather than history-based differences, check the Ignore ancestry box. Read more about this topic in the Subversion book, *Noticing or Ignoring Ancestry* [http://svnbook.red-bean.com/en/1.5/svn.branchmerge.advanced.html#svn.branchmerge.advanced.ancestry]

You can specify the way that line ending and whitespace changes are handled. These options are described in Section 4.10.2, « Options de fins de ligne et d'espacement ». The default behaviour is to treat all whitespace and line-end differences as real changes to be merged.

If you are using merge tracking and you want to mark a revision as having been merged, without actually doing the merge here, check the Only record the merge checkbox. There are two possible reasons you might want to do this. It may be that the merge is too complicated for the merge algorithms, so you code the changes by hand, then mark the change as merged so that the merge tracking algorithm is aware of it. Or you might want to prevent a particular revision from being merged. Marking it as already merged will prevent the merge occurring with merge-tracking-aware clients.

Now everything is set up, all you have to do is click on the Merge button. If you want to preview the results Test Merge performs the merge operation, but does *not* modify the working copy at all. It shows you a list of the files that will be changed by a real merge, and notes those areas where conflicts will occur.

The merge progress dialog shows each stage of the merge, with the revision ranges involved. This may indicate one more revision than you were expecting. For example if you asked to merge revision 123 the progress dialog will report « Merging revisions 122 through 123 ». To understand this you need to remember that Merge is closely related to Diff. The merge process works by generating a list of differences between two points in the repository, and applying those differences to your working copy. The progress dialog is simply showing the start and end points for the diff.

4.20.5. Prévisualiser les résultats de la fusion

The merge is now complete. It's a good idea to have a look at the merge and see if it's as expected. Merging is usually quite complicated. Conflicts often arise if the branch has drifted far from the trunk.

For Subversion clients and servers prior to 1.5, no merge information is stored and merged revisions have to be tracked manually. When you have tested the changes and come to commit this revision, your commit log message should *always* include the revision numbers which have been ported in the merge. If you want to apply another merge at a later time you will need to know what you have already merged, as you do not want to port a change more than once. For more information about this, refer to *Best Practices for Merging* [http://svnbook.red-bean.com/en/1.4/svn.branchmerge.copychanges.html#svn.branchmerge.copychanges.bestprac] in the Subversion book.

If your server and all clients are running Subversion 1.5 or higher, the merge tracking facility will record the revisions merged and avoid a revision being merged more than once. This makes your life much simpler as you can simply merge the entire revision range each time and know that only new revisions will actually be merged.

La gestion de branches est importante. Si vous voulez conserver cette branche à jour avec le tronc, vous devriez vous assurer de fusionner souvent pour que la branche et le tronc ne dérivent pas trop loin. Bien sûr, vous devriez toujours éviter la fusion répétée de changements, comme expliquée ci-dessus.

Astuce

If you have just merged a feature branch back into the trunk, the trunk now contains all the new feature code, and the branch is obsolete. You can now delete it from the repository if required.

Important

Subversion can't merge a file with a folder and vice versa - only folders to folders and files to files. If you click on a file and open up the merge dialog, then you have to give a path to a file in that dialog. If you select a folder and bring up the dialog, then you must specify a folder URL for the merge.

4.20.6. Suivi des fusions

Subversion 1.5 introduced facilities for merge tracking. When you merge changes from one tree into another, the revision numbers merged are stored and this information can be used for several different purposes.

- Vous pouvez éviter ke danger de fusionner deux fois la même révision(problème de fusions répétées). Dès qu'une révision a été marquée comme ayant été fusionnée, les fusions futures incluant cette révision dans leur plage de révisions l'ignoreront.
- Quand vous fusionnez une branche dans le trunk, la fenêtre de log peut vous montrer les livraisons de la branche comme faisant partie des messages de log du trunk, donnant une meilleure traçabilité des modifications.
- When you show the log dialog from within the merge dialog, revisions already merged are shown in grey.
- Quand l'information de bannissement d'un fichier est affichée, vous pouvez choisir de montrer les auteurs des révisions fusionnées, au lieu de la personne ayant fait la fusion.
- Vous pouvez marquer les révisions comme n'étant *pas à fusionner* en les incluant dans la liste des révisions fusionnées mais sans vraiment faire la fusion.

Merge tracking information is stored in the `svn:mergeinfo` property by the client when it performs a merge. When the merge is committed the server stores that information in a database, and when you request merge, log or blame information, the server can respond appropriately. For the system to work properly you must ensure that the server, the repository and all clients are upgraded. Earlier clients will not store the `svn:mergeinfo` property and earlier servers will not provide the information requested by new clients.

Find out more about merge tracking from Subversion's *Merge tracking documentation* [http://subversion.tigris.org/merge-tracking/index.html].

4.20.7. Gérer les conflits durant la fusion.

Merging does not always go smoothly. Sometimes there is a conflict, and if you are merging multiple ranges, you generally want to resolve the conflict before merging of the next range starts. TortoiseSVN helps you through this process by showing the *merge conflict callback* dialog.

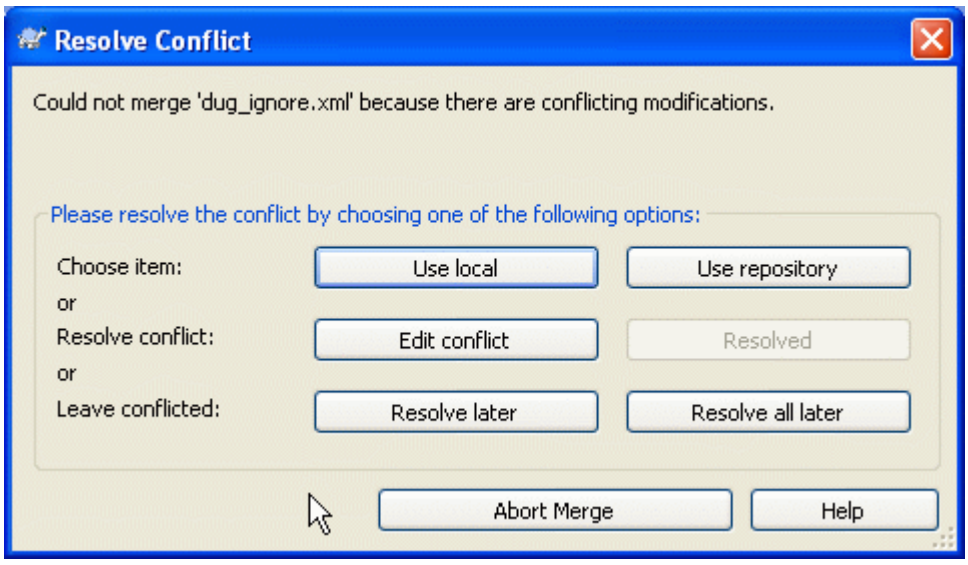


Figure 4.38. La boîte de dialogue de conflit de fusion

Quand la fusion génère un conflit, vous avez trois moyens de le gérer :

1. You may decide that your local changes are much more important, so you want to discard the version from the repository and keep your local version. Or you might discard your local changes in favour of the repository version. Either way, no attempt is made to merge the changes - you choose one or the other.

2. Normally you will want to look at the conflicts and resolve them. In that case, choose the Edit Conflict which will start up your merge tool. When you are satisfied with the result, click Resolved.

3. The last option is to postpone resolution and continue with merging. You can choose to do that for the current conflicted file, or for all files in the rest of the merge. However, if there are further changes in that file, it will not be possible to complete the merge.

If you do not want to use this interactive callback, there is a checkbox in the merge progress dialog Merge non-interactive. If this is set for a merge and the merge would result in a conflict, the file is marked as in conflict and the merge goes on. You will have to resolve the conflicts after the whole merge is finished. If it is not set, then before a file is marked as conflicted you get the chance to resolve the conflict *during* the merge. This has the advantage that if a file gets multiple merges (multiple revisions apply a change to that file), subsequent merges might succeed depending on which lines are affected. But of course you can't walk away to get a coffee while the merge is running ;)

4.20.8. Fusionner une branche complétée

If you want to merge all changes from a feature branch back to trunk, then you can use the TortoiseSVN → Merge reintegrate... from the extended context menu (hold down the **Shift** key while you right click on the file).

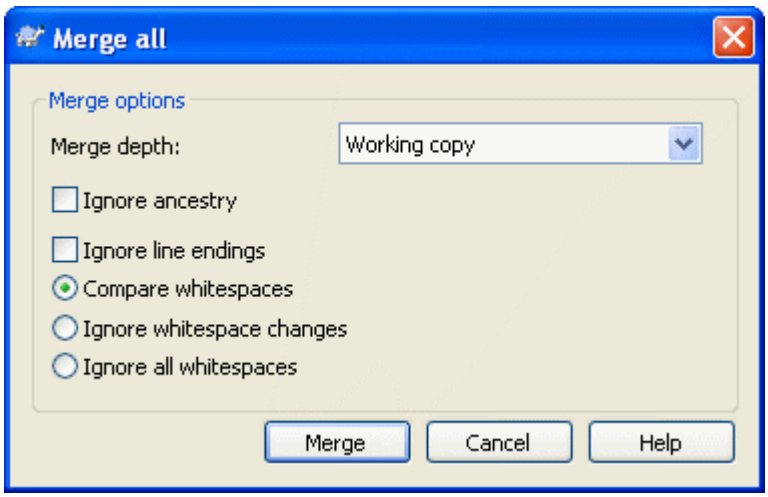


Figure 4.39. La boîte de dialogue de fusion

This dialog is very easy. All you have to do is set the options for the merge, as described in Section 4.20.4, « Options de fusion ». The rest is done by TortoiseSVN automatically using merge tracking.

4.20.9. Branche de maintenance d'une fonctionnalité

Quand vous développez une nouvelle fonctionnalité dans une branche séparée c'est une bonne pratique de garder en tête la réintégration de cette branche lorsque la nouvelle fonctionnalité sera prête. Si la branche principale (la `trunk`) évolue en même temps, il est probable que les différences seront nombreuses, rendant l'intégration cauchemardesque.

Si la fonctionnalité est relativement simple et que le développement ne prend pas beaucoup de temps, vous pouvez adoptez une solution simple, qui consiste à garder une branche distincte, que vous intégrerez lorsque le développement sera fini. Dans l'assistant de fusion, ce devrait n'être qu'un simple Fusionner une gamme de révisions, la gamme de révisions couvrant la période de développement de la branche.

If the feature is going to take longer and you need to account for changes in `trunk`, then you need to keep the branch synchronised. This simply means that periodically you merge trunk changes into the branch, so that the branch contains all the trunk changes *plus* the new feature. The synchronisation process uses Merge a range of revisions. When the feature is complete then you can merge it back to `trunk` using either Reintegrate a branch or Merge two different trees.

4.21. Verrouiller

Subversion fonctionne généralement mieux sans verrouillage, en utilisant les méthodes « Copier-Modifier-Fusionner » décrites plus tôt dans Section 2.2.3, « La solution Copier-Modifier-Fusionner ». Cependant il y a quelques cas où vous pouvez devoir mettre en oeuvre une certaine forme de politique de verrouillage.

- Vous utilisez des fichiers « non fusionnables », par exemple, des fichiers graphiques. Si deux personnes changent le même fichier, la fusion n'est pas possible, donc l'un d'entre elles perdra ses changements.

- Votre société a toujours utilisé un VCS verrouillant par le passé et il y a eu une décision de la part des gestionnaires que « le verrouillage est la meilleure solution ».

Premièrement, vous devez vous assurer que votre serveur Subversion est mis à jour au moins à la version 1.2. Les versions précédentes ne supportent pas le verrouillage. Si vous utilisez un accès de type `file://`, alors il est sûr que votre client doit être mis à jour.

4.21.1. Comment le verrouillage fonctionne dans Subversion

Par défaut, rien n'est verrouillé et quiconque a un accès en livraison peut livrer des changements à n'importe quel fichier à tout moment. Les autres mettront à jour leurs copies de travail périodiquement et les changements du référentiel seront fusionnés avec les changements locaux.

Si vous *obtenez un verrou* sur un fichier, alors vous seul pouvez livrer ce fichier. Les livraisons des autres utilisateurs seront bloquées jusqu'à ce que vous relâchiez le verrou. Un fichier verrouillé ne peut être modifié d'aucune façon dans le référentiel, il ne peut donc pas être supprimé ou renommé non plus, sauf par le propriétaire du verrou.

Toutefois, les autres utilisateurs ne sauront pas nécessairement que vous avez pris un verrou. À moins qu'ils ne vérifient le statut du verrou régulièrement, ils le sauront la première fois lorsque leur livraison échouera, ce qui n'est pas très utile dans la plupart des cas. Pour rendre la gestion des verrous plus facile, il existe une nouvelle propriété Subversion `svn:needs-lock`. Quand cette propriété est définie (avec n'importe quelle valeur) sur un fichier, chaque fois que le fichier est extrait ou mis à jour, la copie locale est mise en lecture seule *à moins que* la copie de travail ne détienne un verrou pour le fichier. Cela agit comme un avertissement pour ne pas éditer ce fichier à moins que vous n'ayez d'abord acquis un verrou. Les fichiers qui sont versionnés et en lecture seule sont marqués avec un recouvrement spécial dans TortoiseSVN pour indiquer que vous devez acquérir un verrou avant l'édition.

Les verrous sont enregistrées par emplacement de copie de travail et par propriétaire. Si vous avez plusieurs copies de travail (à la maison, au travail) alors vous pouvez seulement détenir un verrou dans *une* de ces copies de travail.

Si l'un de vos collaborateurs acquiert un verrou et part ensuite en vacances sans le relâcher, que faites-vous ? Subversion fournit un moyen de forcer les verrous. Relâcher un verrou détenu par quelqu'un d'autre est connu comme *Casser* le verrou et l'acquisition de force d'un verrou déjà détenu par quelqu'un d'autre s'appelle *Voler* le verrou. Naturellement, ce ne sont pas des choses que vous devriez faire à la légère si vous voulez rester ami avec vos collaborateurs.

Les verrous sont enregistrés dans le référentiel et un jeton de verrouillage est créé dans votre copie de travail locale. S'il y a une incohérence, par exemple si quelqu'un d'autre a cassé le verrou, le jeton de verrouillage local devient invalide. Le référentiel est toujours la référence définitive.

4.21.2. Obtenir un verrou

Sélectionnez les fichiers dans votre copie de travail pour lesquels vous voulez acquérir un verrou, puis sélectionnez la commande TortoiseSVN → Obtenir un verrou....

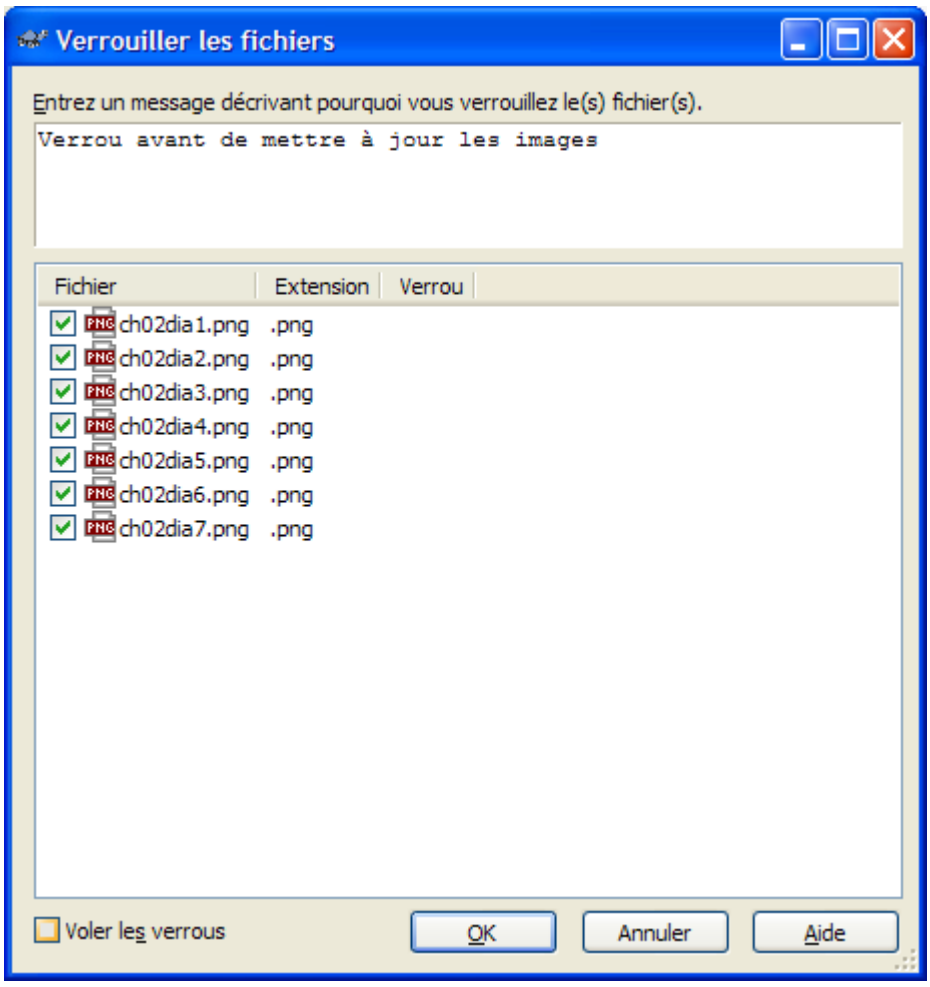


Figure 4.40. La boîte de dialogue Verrouiller

Une boîte de dialogue apparaît, vous permettant de saisir un commentaire, pour que les autres puissent voir pourquoi vous avez verrouiller le fichier. Le commentaire est facultatif et seulement utilisé avec des référentiels basés sur Svnserve actuellement. Si (et *seulement* si) vous deviez voler le verrou de quelqu'un d'autre, cochez la case Voler les verrous, cliquez ensuite sur OK.

Si vous sélectionnez un dossier et utilisez ensuite TortoiseSVN → Obtenir un verrou... la boîte de dialogue Verrouiller s'ouvrira avec *tous* les fichiers de *tous* les sous-dossiers choisis pour les verrouiller. C'est la bonne manière de faire si vous voulez vraiment verrouiller une arborescence complète, cependant vous pourriez devenir très impopulaire avec vos collaborateurs si vous les empêchez d'accéder au projet. À utiliser avec parcimonie...

4.21.3. Relâcher un verrou

Pour vous assurer de ne pas oubliez de relâcher un verrou dont vous n'avez plus besoin, les fichiers verrouillés sont affichés dans la boîte de dialogue de livraison et sélectionnés par défaut. Si vous continuez la livraison, les verrous que vous détenez sur les fichiers sélectionnés sont supprimés, même si les fichiers n'ont pas été modifiés. Si vous ne voulez pas relâcher les verrous de certains fichiers, vous pouvez les décocher (s'ils ne sont pas modifiés). Si vous voulez garder un verrou sur un fichier que vous avez modifié, vous devez activer la case à cocher Garder les verrous avant livrer vos changements.

Pour relâcher un verrou manuellement, sélectionnez les fichiers dans votre copie de travail pour lesquels vous voulez relâcher les verrous, choisissez ensuite la commande TortoiseSVN → Relâcher un verrou Il n'y a rien d'autre à faire donc TortoiseSVN va contacter le référentiel et relâcher les verrous. Vous pouvez aussi utiliser cette commande sur un dossier pour relâcher tous les verrous récursivement.

4.21.4. Vérifier le statut des verrous

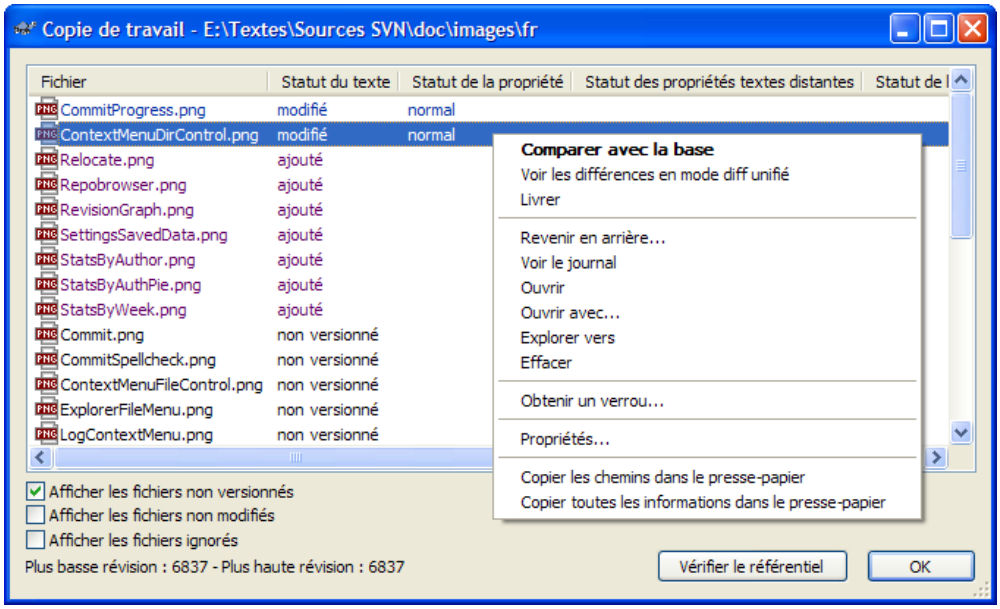


Figure 4.41. La boîte de dialogue Vérifier les modifications

Pour voir quels verrous les autres et vous détenez, vous pouvez utiliser TortoiseSVN → Vérifier les modifications.... Les jetons des verrous détenus localement s'afficheront immédiatement. Pour vérifier les verrous détenus par les autres (et voir si l'un de vos verrous est cassé ou volé) vous devez cliquer sur Vérifier le référentiel.

À partir du menu contextuel ici, vous pouvez aussi bien obtenir et relâcher des verrous, que casser et voler des verrous détenus par d'autres.

Évitez de casser et de voler les verrous

Si vous cassez ou volez le verrou de quelqu'un d'autre sans le lui dire, vous pourriez potentiellement causer une perte du travail. Si vous travaillez avec des types de fichier non fusionnables et que vous volez le verrou de quelqu'un d'autre, une fois que vous relâchez le verrou, ils sont libres de livrer leurs changements et d'écraser les vôtres. Subversion ne perd pas de données, mais vous avez perdu la protection du travail en équipe que le verrouillage vous a donnée.

4.21.5. Mettre les fichiers non verrouillés en Lecture seule

Comme mentionné ci-dessus, la façon la plus efficace d'utiliser le verrouillage est de mettre la propriété `svn:needs-lock` sur les fichiers. Référez-vous à Section 4.17, « Configuration des projets » pour les instructions sur la façon de définir les propriétés. Les fichiers avec cette propriété définie seront toujours extraits et mis à jour avec l'attribut Lecture seule défini à moins que votre copie de travaill ne détienne un verrou.

Pour vous le rappeler, TortoiseSVN utilise un recouvrement spécial pour l'indiquer.

Si vous pratiquez une politique où chaque fichier doit être verrouillé, vous pouvez trouver alors plus facile d'utiliser la fonctionnalité auto-props de Subversion pour définir la propriété automatiquement à chaque fois vous ajoutez de nouveaux fichiers. Lisez Section 4.30, « Configuration de TortoiseSVN » pour plus d'informations.

4.21.6. Les scripts hook de verrouillage

Quand vous créez un nouveau référentiel avec Subversion 1.2 ou supérieur, quatre modèles de hooks sont créés dans le répertoire `hooks` du référentiel. Ceux-ci sont appelés avant et après l'obtention d'un verrou et avant et après la libération d'un verrou.

C'est une bonne idée d'installer un script hook `post-lock` et `post-unlock` sur le serveur qui envoie un e-mail indiquant que le fichier a été verrouillé. Avec une tel script en place, tous vos utilisateurs peuvent être informés si quelqu'un verrouille/déverrouille un fichier. Vous pouvez trouver un script hook exemple `hooks/post-lock.tmpl` dans le dossier de votre référentiel.

Vous pourriez aussi utiliser des hooks pour rejeter la casse ou le vol de verrous, ou les limiter peut-être à un administrateur nommé. Ou peut-être vous voulez envoyer un email au propriétaire quand un de ses verrous est cassé ou volé.

Lisez Section 3.3, « Scripts de hook côté serveur » pour en savoir plus.

4.22. Créer et appliquer des patches

Pour les projets open-source (comme celui-ci), tout le monde a accès au référentiel en lecture et tout le monde peut contribuer au projet. Alors comment ces contributions sont-elles contrôlées ? Si tout le monde pouvait livrer des changements, le projet serait instable en permanence et probablement cassé en permanence. Dans cette situation, le changement est géré en soumettant un fichier *patch* à l'équipe de développement, qui a accès en écriture. Ils peuvent passer en revue le patch d'abord et ensuite le soumettre au référentiel ou le renvoyer à l'auteur.

Les fichiers patch sont simplement des fichiers de différences unifiées montrant les différences entre votre copie de travail et la révision de base.

4.22.1. Créer un patch

D'abord vous devez faire *et tester* vos changements. Alors au lieu d'utiliser TortoiseSVN → Livrer... sur le dossier parent, vous sélectionnez TortoiseSVN → Créer un patch...

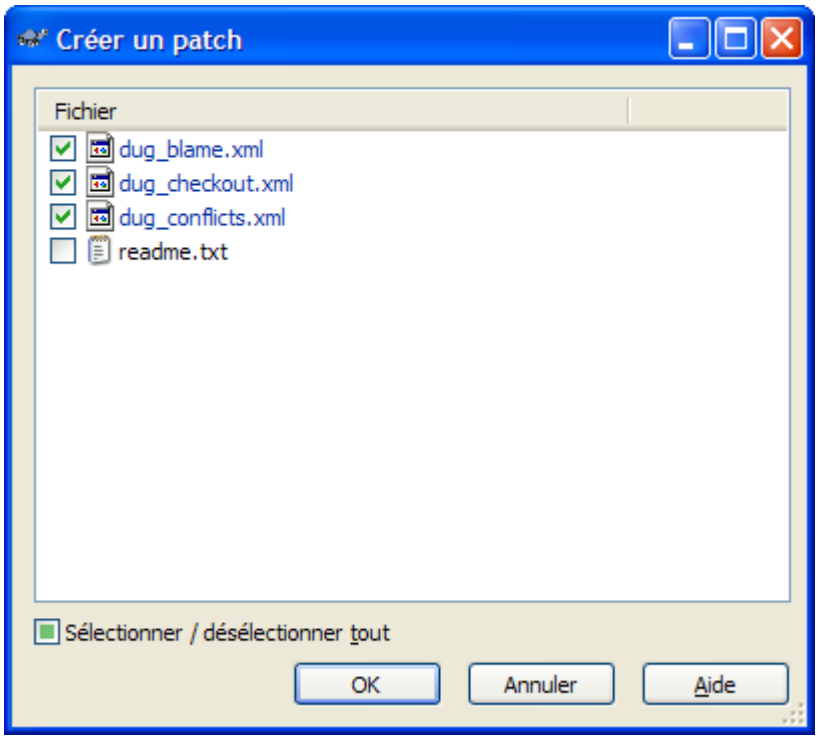


Figure 4.42. La boîte de dialogue Créer un patch

Vous pouvez maintenant choisir les fichiers que vous voulez inclure dans le patch, comme vous le feriez avec une livraison complète. Cela produira un unique fichier contenant un résumé de tous les changements que vous avez faits aux fichiers sélectionnés depuis la dernière mise à jour du référentiel.

Les colonnes dans cette boîte de dialogue peuvent être personnalisées de la même manière que les colonnes dans la boîte de dialogue Vérifier les modifications. Lisez Section 4.7.3, « Statut local et distant » pour plus de détails.

Vous pouvez produire des patches séparés contenant des changements sur des jeux différents de fichiers. Bien sûr, si vous créez un fichier patch, faire un peu plus de modifications aux *mêmes* fichiers et créez ensuite un autre patch, le deuxième fichier patch inclura les *deux* jeux de changements.

Sauvegardez juste le fichier en utilisant un nom de fichier de votre choix. Les patches peuvent avoir l'extension de votre choix, mais selon la convention ils devraient utiliser les extensions `.patch` ou `.diff`. Vous êtes maintenant prêt à soumettre votre patch.

You can also save the patch to the clipboard instead of to a file. You might want to do this so that you can paste it into an email for review by others. Or if you have two working copies on one machine and you want to transfer changes from one to the other, a patch on the clipboard is a convenient way of doing this.

4.22.2. Appliquer un patch

Patch files are applied to your working copy. This should be done from the same folder level as was used to create the patch. If you are not sure what this is, just look at the first line of the patch file. For example, if the first file being worked on was `doc/source/english/chapter1.xml` and the first line in the patch file is `Index: english/chapter1.xml` then you need to apply the patch to the `doc/source/` folder. However, provided you are in the correct working copy, if you pick the wrong folder level, TortoiseSVN will notice and suggest the correct level.

Pour appliquer un patch à votre copie de travail, vous devez avoir au moins l'accès en lecture pour le référentiel. La raison à cela est que le programme de fusion doit faire référence aux changements de la révision avec laquelle ils ont été faits par le développeur distant.

From the context menu for that folder, click on TortoiseSVN → Apply Patch... This will bring up a file open dialog allowing you to select the patch file to apply. By default only `.patch` or `.diff` files are shown, but you can opt for « All files ». If you previously saved a patch to the clipboard, you can use Open from clipboard... in the file open dialog.

Alternativement, si le fichier patch a une extension `.patch` ou `.diff`, vous pouvez faire un clic droit dessus directement et sélectionnez TortoiseSVN → Appliquer un patch.... Dans ce cas, vous serez invités à entrer un emplacement de copie de travail.

Ces deux méthodes offrent juste des façons différentes de faire la même chose. Avec la première méthode, vous choisissez la CdT et naviguez jusqu'au patch. Avec la deuxième, vous choisissez le patch et naviguez jusqu'à la CdT.

Une fois que vous avez sélectionné le fichier patch et l'emplacement de la copie de travail, TortoiseMerge se lance pour fusionner les changements du fichier patch avec votre copie de travail. Une petite fenêtre liste les fichiers qui ont été changés. Double-cliquez sur chacun à son tour, passez en revue les changements et sauvegardez les fichiers fusionnés.

Le patch du développeur distant a maintenant été appliqué à votre copie de travail, donc vous devez livrer pour permettre à tous les autres d'avoir accès aux changements depuis le référentiel.

4.23. Qui a changé quelle ligne ?

Parfois, vous avez besoin de connaître non seulement quelles lignes ont changé, mais aussi exactement qui a changé des lignes spécifiques dans un fichier. C'est à ce moment que la commande TortoiseSVN → Annoter..., parfois aussi mentionnée comme la commande *annoter* devient pratique.

Cette commande liste, pour chaque ligne d'un fichier, l'auteur et la révision où la ligne a été changée.

4.23.1. Annoter pour les fichiers

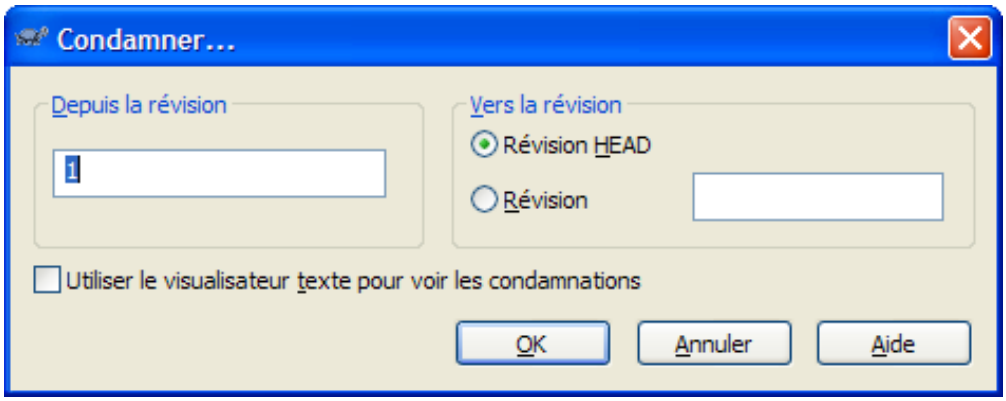


Figure 4.43. La boîte de dialogue Annoter

Si vous n'êtes pas intéressés par les changements des révisions précédentes, vous pouvez définir la révision d'où l'annotation devrait commencer. Mettez cela à `1`, si vous voulez l'annotation pour *toutes* les révisions.

Par défaut le fichier d'annotations est visualisé en utilisant *TortoiseBlame*, qui met en évidence les différentes révisions pour le rendre plus facile à lire. Si vous voulez imprimer ou éditer le fichier d'annotations, sélectionnez Utiliser le visualisateur texte pour voir les annotations

You can specify the way that line ending and whitespace changes are handled. These options are described in Section 4.10.2, « Options de fins de ligne et d'espacement ». The default behaviour is to treat all whitespace and line-end differences as real changes, but if you want to ignore an indentation change and find the original author, you can choose an appropriate option here.

Une fois que vous cliquez sur OK TortoiseSVN commence à récupérer les données pour créer le fichier d'annotation. Veuillez noter : Cela peut prendre plusieurs minutes, selon le nombre de fois que le fichier a été modifié et bien sûr, selon votre connexion réseau au référentiel. Une fois que le processus d'annotation est terminé, le résultat est écrit dans un fichier temporaire et vous pouvez voir les résultats.

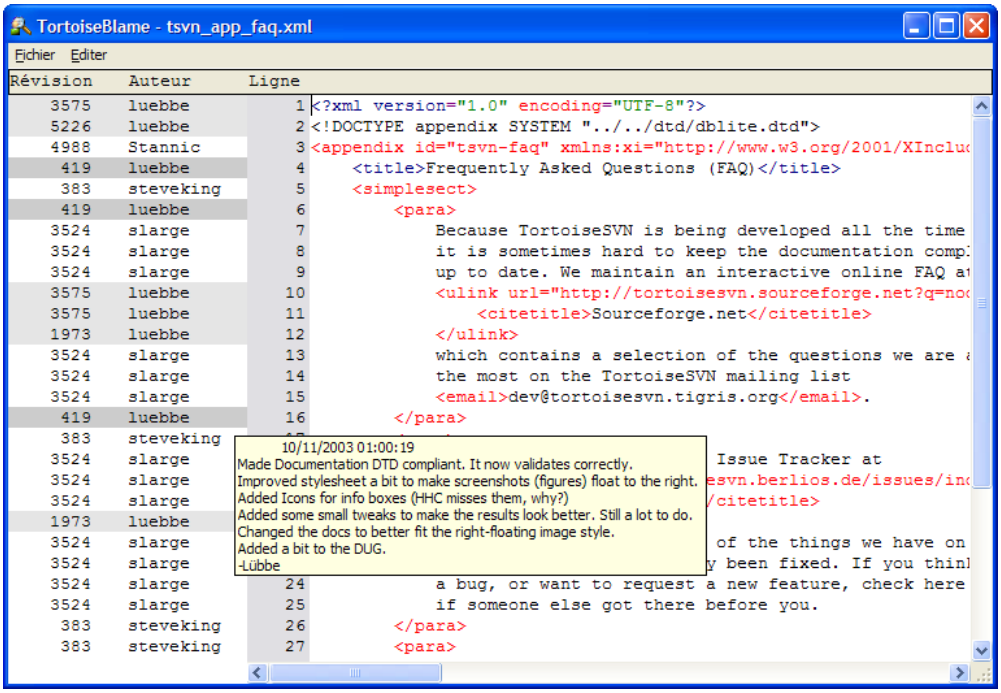


Figure 4.44. TortoiseBlame

TortoiseBlame, qui est inclus avec TortoiseSVN, rend le fichier d'annotation plus facile à lire. Quand vous survolez avec la souris une ligne dans la colonne de renseignements de l'annotation, toutes les lignes avec la même révision sont affichées avec un fond plus sombre. Les lignes d'autres révisions qui ont été changées par le même auteur sont affichées avec un fond clair. La coloration peut ne pas fonctionner aussi clairement si vous avez votre affichage en mode 256 couleurs.

Si vous faites un clic gauche sur une ligne, toutes les lignes avec la même révision sont mises en évidence et les lignes d'autres révisions du même auteur sont mises en évidence dans une couleur plus claire. Cette accentuation est figée, vous permettant de déplacer la souris sans perdre les points en évidence. Cliquez sur cette révision de nouveau pour arrêter l'accentuation.

Les commentaires de révision (commentaire du journal) sont affichés dans une info-bulle à chaque fois que la souris survole la colonne d'information de l'annotation. Si vous voulez copier le commentaire de cette révision, utilisez le menu contextuel qui apparaît quand vous faites un clic droit sur la colonne d'information de l'annotation.

Vous pouvez faire des recherches dans le rapport d'annotation en utilisant Éditer → Rechercher.... Cela vous permet de chercher des numéros de révision, des auteurs et le contenu du fichier lui-même. Les commentaires ne sont pas inclus dans la recherche - vous devriez utiliser la boîte de dialogue de Journal pour la recherche.

Vous pouvez aussi aller à une ligne particulière en utilisant Editer → Aller à la ligne....

When the mouse is over the blame info columns, a context menu is available which helps with comparing revisions and examining history, using the revision number of the line under the mouse as a reference. Context menu → Blame previous revision generates a blame report for the same file, but using the previous revision as the upper limit. This gives you the blame report for the state of the file just before the line you are looking at was last changed. Context menu → Show changes starts your diff viewer, showing you what changed in the referenced revision. Context menu → Show log displays the revision log dialog starting with the referenced revision.

If you need a better visual indicator of where the oldest and newest changes are, select View → Color age of lines. This will use a colour gradient to show newer lines in red and older lines in blue. The default colouring is quite light, but you can change it using the TortoiseBlame settings.

If you are using Merge Tracking, where lines have changed as a result of merging from another path, TortoiseBlame will show the revision and author of the last change in the original file rather than the revision where the merge took place. These lines are indicated by showing the revision and author in italics. If you do not want merged lines shown in this way, uncheck the Include merge info checkbox.

Si vous souhaitez voir les chemins concernés dans la fusion, sélectionnez Voir → Chemins Fusionnés.

Les paramètres de TortoiseBlame peuvent être affiché en sélectionnant TortoiseSVN → Paramètres... dans l'onglet TortoiseBlame. Voir Section 4.30.9, « Configuration de TortoiseBlame ».

4.23.2. Annoter les différences

L'une des limitations du rapport d'annotation est qu'il ne montre que le fichier tel qu'il était à une révision particulière et montre la dernière personne à avoir changé chaque ligne. Parfois vous voulez connaître quel changement a été fait, et aussi qui l'a fait. Ce dont vous avez ici besoin est une combinaison des rapports d'annotation et de comparaison.

La boîte de dialogue du journal de révision inclut plusieurs options vous permettant de faire cela.

Annoter les révisions
: Dans le panneau supérieur, sélectionnez 2 révisions, puis sélectionnez Menu contextuel → Annoter les révisions. Cela parcoura les annotations pour les 2 révisions puis utilisera le visualisateur de différences pour comparer les deux fichiers d'annotation.

Bannir les modifications
: Sélectionnez une révision dans le panneau supérieur, puis choisissez un fichier dans le panneau inférieur et sélectionnez Menu contextuel → Annoter les différences. Cela ira chercher les annotations pour la révision sélectionnée et la révision précédente, puis utilisera le visualisateur de différences pour comparer les deux fichiers d'annotations.

Comparez et annoter avec la BASE de travail
: Affichez le journal pour un seul fichier et, dans le panneau supérieur, choisissez une seule révision, puis sélectionnez Menu contextuel → Comparer et annoter avec la BASE de travail. Cela ira chercher les annotations pour la révision choisie et pour le fichier dans la BASE de travail, puis utilisera ensuite le visualisateur de différences pour comparer les deux fichiers d'annotations.

4.24. l'explorateur de référentiel

Parfois, vous devez travailler directement sur le référentiel, sans avoir de copie de travail. C'est à cela que sert le *Explorateur de référentiel*. Comme l'explorateur et les recouvrements d'icône qui vous permettent de voir votre copie de travail, l'explorateur de référentiel vous permet de voir la structure et le statut du référentiel.

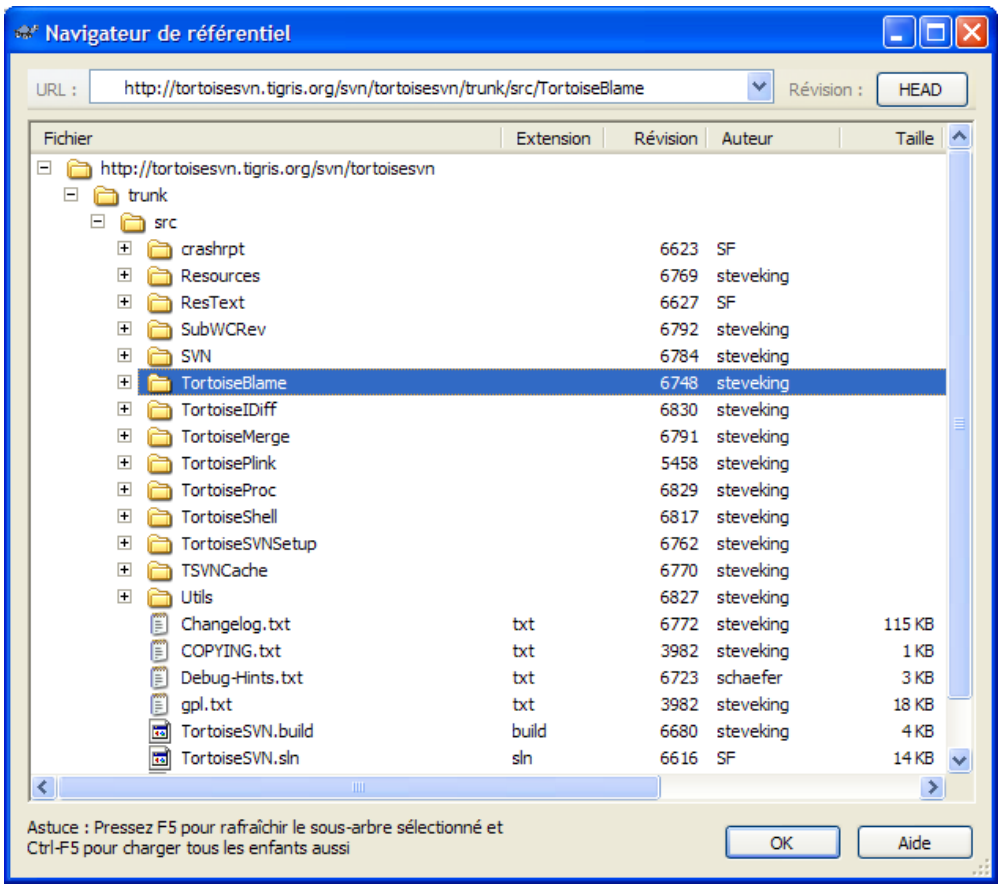


Figure 4.45. l'explorateur de référentiel

Avec l'explorateur de référentiel vous pouvez exécuter des commandes comme copier, déplacer, renommer... directement sur le référentiel.

The repository browser looks very similar to the Windows explorer, except that it is showing the content of the repository at a particular revision rather than files on your computer. In the left pane you can see a directory tree, and in the right pane are the contents of the selected directory. At the top of the Repository Browser Window you can enter the URL of the repository and the revision you want to browse.

Tout comme dans l'explorateur Windows, vous pouvez clicquer sur les en-têtes des colonnes dans le panneau de droite si vous voulez modifier le tri. Et comme dans l'explorateur il yu a des menus contextuels ans les deux panneaux.

Le menu contextuel pour un fichier vous permet de :

- Ouvrir le fichier sélectionné, avec le visualisateur par défaut pour ce type de fichier ou avec le programme de votre choix.
- Récupérer une copie non versionnée du fichier sur votre disque dur.
- Show the revision log for that file, or show a graph of all revisions so you can see where the file came from.
- Annoter le fichier pour voir qui a modifier quelle ligne et quand.
- Supprimez ou renommez le fichier.
- Make a copy of the file, either to a different part of the repository, or to a working copy rooted in the same repository.

- Voir/Modifier les propriétés du fichier

Le menu contextuel pour un dossier vous permet de :

- Voir l'historique des messages de log de ce répertoire, ou voir le graphique de toutes les révisions afin de pouvoir savoir d'où vient ce répertoire.
- Exporter le répertoire vers une copie non versionnée sur votre disque dur.
- Extraire le répertoire pour avoir une version de travail sur votre disque dur.
- Créez un nouveau répertoire dans le référentiel
- Ajouter directement des fichiers ou des répertoires dans le référentiel.
- Supprimez ou renommez le répertoire.
- Make a copy of the folder, either to a different part of the repository, or to a working copy rooted in the same repository.
- Voir/Modifier les propriétés du dossier.
- Marque le répertoire pour la comparaison. Un nom d'un répertoire marqué est en gras.
- Compare the folder with a previously marked folder, either as a unified diff, or as a list of changed files which can then be visually diffed using the default diff tool. This can be particularly useful for comparing two tags, or trunk and branch to see what changed.

If you select two folders in the right pane, you can view the differences either as a unified-diff, or as a list of files which can be visually diffed using the default diff tool.

If you select multiple folders in the right pane, you can checkout all of them at once into a common parent folder.

Si vous sélectionnez 2 étiquettes qui sont copiées depuis la même racine (typiquement `/trunk/`), vous pouvez utiliser Menu contextuel → Voir le journal pour voir la liste des révisions entre les deux points d'étiquettes.

You can use **F5** to refresh the view as usual. This will refresh everything which is currently displayed. If you want to pre-fetch or refresh the information for nodes which have not been opened yet, use **Ctrl-F5**. After that, expanding any node will happen instantly without a network delay while the information is fetched.

You can also use the repository browser for drag-and-drop operations. If you drag a folder from explorer into the repo-browser, it will be imported into the repository. Note that if you drag multiple items, they will be imported in separate commits.

Si vous voulez déplacer un élément au sein du référentiel, effectuez un glisser-déplacer avec le bouton gauche de la souris vers le nouvel emplacement. Si vous voulez créer une copie plutôt que de déplacer l'élément, effectuez un glisser-déplacer avec le bouton gauche de la souris, tout en maintenant la touche **Ctrl** enfoncée. Lors de la copie, le curseur affiche un symbole « plus », comme dans l'Explorateur.

If you want to copy/move a file or folder to another location and also give it a new name at the same time, you can right drag or **Ctrl**-right drag the item instead of using left drag. In that case, a rename dialog is shown where you can enter a new name for the file or folder.

Quand vous faites des changements dans le référentiel en utilisant l'une de ces méthodes, une boîte de dialogue de saisie de commentaire vous sera présentée. Si vous avez glissé quelque chose par erreur, c'est aussi une chance pour annuler l'action.

Sometimes when you try to open a path you will get an error message in place of the item details. This might happen if you specified an invalid URL, or if you don't have access permission, or if there is some other server problem. If you need to copy this message to include it in an email, just right click on it and use Context Menu → Copy error message to clipboard, or simply use **Ctrl+C**.

4.25. Graphiques de révision

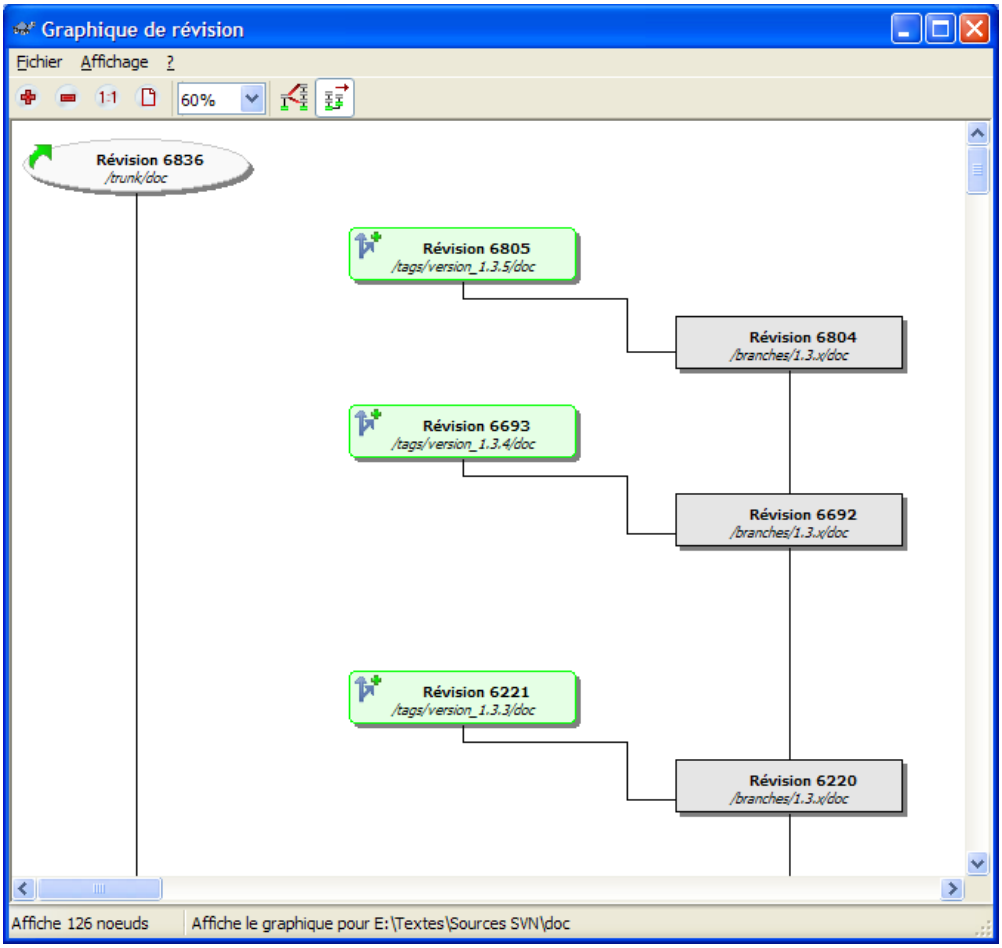


Figure 4.46. Un graphique de révision

Parfois vous avez besoin de savoir à quel moment les branches et les étiquettes ont été prises du tronc et la façon idéale de voir ce genre d'informations est comme un graphique ou une structure arborescente. C'est à ce moment que vous devez utiliser TortoiseSVN → Graphique de révision

Cette commande analyse l'historique des révisions et essaye de créer un arbre montrant les points auxquels les copies ont été prises et quand les branches/étiquettes ont été supprimées.

Important

In order to generate the graph, TortoiseSVN must fetch all log messages from the repository root. Needless to say this can take several minutes even with a repository of a few thousand revisions, depending on server speed, network bandwidth, etc. If you try this with something like the *Apache* project which currently has over 500,000 revisions you could be waiting for some time.

> The good news is that if you are using log caching, you only have to suffer this delay once. After that, log data is held locally. Log caching is enabled in TortoiseSVN's settings.

4.25.1. Graphiques des révisions

Each revision graph node represents a revision in the repository where something changed in the tree you are looking at. Different types of node can be distinguished by shape and colour. The shapes are fixed, but colours can be set using TortoiseSVN → Settings

Eléments ajoutés ou copiés
: Items which have been added, or created by copying another file/folder are shown using a rounded rectangle. The default colour is green. Tags and trunks are treated as a special case and use a different shade, depending on the TortoiseSVN → Settings

Elements supprimés
: Deleted items eg. a branch which is no longer required, are shown using an octagon (rectangle with corners cut off). The default colour is red.

Elements renommés
: Renamed items are also shown using an octagon, but the default colour is blue.

Révision de pointe de branche
: The graph is normally restricted to showing branch points, but it is often useful to be able to see the respective HEAD revision for each branch too. If you select Show HEAD revisions, each HEAD revision nodes will be shown as an ellipse. Note that HEAD here refers to the last revision committed on that path, not to the HEAD revision of the repository.

Révision de la Copie de Travail
: If you invoked the revision graph from a working copy, you can opt to show the BASE revision on the graph using Show WC revision, which marks the BASE node with a bold outline.

Copie de travail modifiée
: If you invoked the revision graph from a working copy, you can opt to show an additional node representing your modified working copy using Show WC modifications. This is an elliptical node with a bold outline in red by default.

Element normal
: Tous les autres éléments sont affichés en utilisant un rectangle plat.

Note that by default the graph only shows the points at which items were added, copied or deleted. Showing every revision of a project will generate a very large graph for non-trivial cases. If you really want to see *all* revisions where changes were made, there is an option to do this in the View menu and on the toolbar.

The default view (grouping off) places the nodes such that their vertical position is in strict revision order, so you have a visual cue for the order in which things were done. Where two nodes are in the same column the order is very obvious. When two nodes are in adjacent columns the offset is much smaller because there is no need to prevent the nodes from overlapping, and as a result the order is a little less obvious. Such optimisations are necessary to keep complex graphs to a reasonable size. Please note that this ordering uses the *edge* of the node on the *older* side as a reference, i.e. the bottom edge of the node when the graph is shown with oldest node at the bottom. The reference edge is significant because the node shapes are not all the same height.

4.25.2. Changer l'affichage

Because a revision graph is often quite complex, there are a number of features which can be used to tailor the view the way you want it. These are available in the View menu and from the toolbar.

Groupe les branches
: The default behavior (grouping off) has all rows sorted strictly by revision. As a result, long-living branches with sparse commits occupy a whole column for only a few changes and the graph becomes very broad.

  This mode groups changes by branch, so that there is no global revision ordering: Consecutive revisions on a branch will be shown in (often) consecutive lines. Sub-branches, however, are arranged in such a way that later branches will be shown in the same column above older branches to keep the graph slim. As a result, a given row may contain changes from different revisions.

Les plus anciens en haut
: Normally the graph shows the oldest revision at the bottom, and the tree grows upwards. Use this option to grow down from the top instead.

Aligner les arborescences en haut
: When a graph is broken into several smaller trees, the trees may appear either in natural revision order, or aligned at the bottom of the window, depending on whether you are using the Group Branches option. Use this option to grow all trees down from the top instead.

Réduire les intersections
: If the layout of the graph has produced a lot of crossing lines, use this option to clean it up. This may make the layout columns appear in less logical places, for example in a diagonal line rather than a column, and the graph may require a larger area to draw.

Differential path names
: Long path names can take a lot of space and make the node boxes very large. Use this option to show only the changed part of a path, replacing the common part with dots. E.g. if you create a branch `/branches/1.2.x/doc/html` from `/trunk/doc/html` the branch could be shown in compact form as `/branches/1.2.x/..` because the last two levels, `doc` and `html`, did not change.

Montrer toutes les révisions
: This does just what you expect and shows every revision where something (in the tree that you are graphing) has changed. For long histories this can produce a truly huge graph.

Montrer la version de tête (HEAD)
: This ensures that the latest revision on every branch is always shown on the graph.

Copie exacte des sources
: When a branch/tag is made, the default behaviour is to show the branch as taken from the last node where a change was made. Strictly speaking this is inaccurate since the branches are often made from the current HEAD rather than a specific revision. So it is possible to show the more correct (but less useful) revision that was used to create the copy. Note that this revision may be younger than the HEAD revision of the source branch.

Replier les étiquettes
: When a project has many tags, showing every tag as a separate node on the graph takes a lot of space and obscures the more interesting development branch structure. At the same time you may need to be able to access the tag content easily so that you can compare revisions. This option hides the nodes for tags and shows them instead in the tooltip for the node that they were copied from. A tag icon on the right side of the source node indicates that tags were made.

Cacher les répertoires supprimés
: Hides paths which are no longer present at the HEAD revision of the repository, e.g. deleted branches.

Cacher les branches non modifiées
: Hides branches where no changes were committed to the respective file or sub-folder. This does not necessarily indicate that the branch was not used, just that no changes were made to *this* part of it.

Montrer la version de la CdT
: Marks the revision on the graph which corresponds to the update revision of the item you fetched the graph for. If you have just updated, this will be HEAD, but if others have committed changes

since your last update your WC may be a few revisions lower down. The node is marked by giving it a bold outline.

Montrer les modifications de la CdT
: If your WC contains local changes, this option draws it as a separate elliptical node, linked back to the node that your WC was last updated to. The default outline colour is red. You may need to refresh the graph using **F5** to capture recent changes.

filtrer
: Sometimes the revision graph contains more revisions than you want to see. This option opens a dialog which allows you to restrict the range of revisions displayed, and to hide particular paths by name.

Tree stripes
: Where the graph contains several trees, it is sometimes useful to use alternating colours on the background to help distinguish between trees.

Montrer l'aperçu
: Shows a small picture of the entire graph, with the current view window as a rectangle which you can drag. This allows you to navigate the graph more easily. Note that for very large graphs the overview may become useless due to the extreme zoom factor and will therefore not be shown in such cases.

4.25.3. Utiliser le Graphique de révisions

To make it easier to navigate a large graph, use the overview window. This shows the entire graph in a small window, with the currently displayed portion highlighted. You can drag the highlighted area to change the displayed region.

La date de révision, l'auteur et les commentaires sont affichés dans une info-bulle chaque fois que la souris survole une boîte de révision.

If you select two revisions (Use **Ctrl**-left click), you can use the context menu to show the differences between these revisions. You can choose to show differences as at the branch creation points, but usually you will want to show the differences at the branch end points, i.e. at the HEAD revision.

Vous pouvez voir les différences avec un fichier de différences unifiées, qui montre toutes les différences dans un seul fichier avec un contexte minimal. Si vous optez pour Menu contextuel → Comparer les révisions, vous vous retrouvez avec une liste des fichiers modifiés. Double-cliquez sur un nom de fichier pour aller chercher les deux révisions du fichier et les comparer en utilisant l'outil de comparaison visuel.

Si vous faites un clic droit sur une révision, vous pouvez utiliser Menu contextuel → Voir le journal pour voir l'historique.

You can also merge changes in the selected revision(s) into a different working copy. A folder selection dialog allows you to choose the working copy to merge into, but after that there is no confirmation dialog, nor any opportunity to try a test merge. It is a good idea to merge into an unmodified working copy so that you can revert the changes if it doesn't work out! This is a useful feature if you want to merge selected revisions from one branch to another.

Apprendre à Lire le Graphe de Révisions

First-time users may be surprised by the fact that the revision graph shows something that does not match the user's mental model. If a revision changes multiple copies or branches of a file or folder, for instance, then there will be multiple nodes for that single revision. It is a good practice to start with the leftmost options in the toolbar and customize the graph step-by-step until it comes close to your mental model.

All filter options try lose as little information as possible. That may cause some nodes to change their color, for instance. Whenever the result is unexpected, undo the last filter

> operation and try to understand what is special about that particular revision or branch. In most cases, the initially expected outcome of the filter operation would either be inaccurate or misleading.

4.25.4. Refraîchissement de l'affichage

If you want to check the server again for newer information, you can simply refresh the view using **F5**. If you are using the log cache (enabled by default), this will check the repository for newer commits and fetch only the new ones. If the log cache was in offline mode, this will also attempt to go back online.

If you are using the log cache and you think the message content or author may have changed, you should use the log dialog to refresh the messages you need. Since the revision graph works from the repository root, we would have to invalidate the entire log cache, and refilling it could take a *very* long time.

4.25.5. Pruning Trees

A large tree can be difficult to navigate and sometimes you will want to hide parts of it, or break it down into a forest of smaller trees. If you hover the mouse over the point where a node link enters or leaves the node you will see one or more popup buttons which allow you to do this.

Cliquez sur le bouton "moins" pour réduire la sous arborescence.

Cliquez sur le bouton "plus" pour développer une sous arborescence. Quand une arborescence a été réduite, ce bouton reste pour montrer la sous arborescence cachée.

Cliquez sur le bouton "croix" pour séparer la sous arborescence et l'afficher comme une arborescence séparée.

Cliquez sur le bouton "rond" pour ré-attacher une sous arborescence préalablement séparée. Quand une arborescence a été séparée, ce bouton reste visible pouir indiquer qu'il y a une sous arborescence séparée.

Click on the graph background for the main context menu, which offers options to Expand all and Join all. If no branch has been collapsed or split, the context menu will not be shown.

4.26. Exporter une copie de travail Subversion

Sometimes you may want a copy of your working tree without any of those `.svn` directories, e.g. to create a zipped tarball of your source, or to export to a web server. Instead of making a copy and then deleting all those `.svn` directories manually, TortoiseSVN offers the command TortoiseSVN → Export.... Exporting from a URL and exporting from a working copy are treated slightly differently.

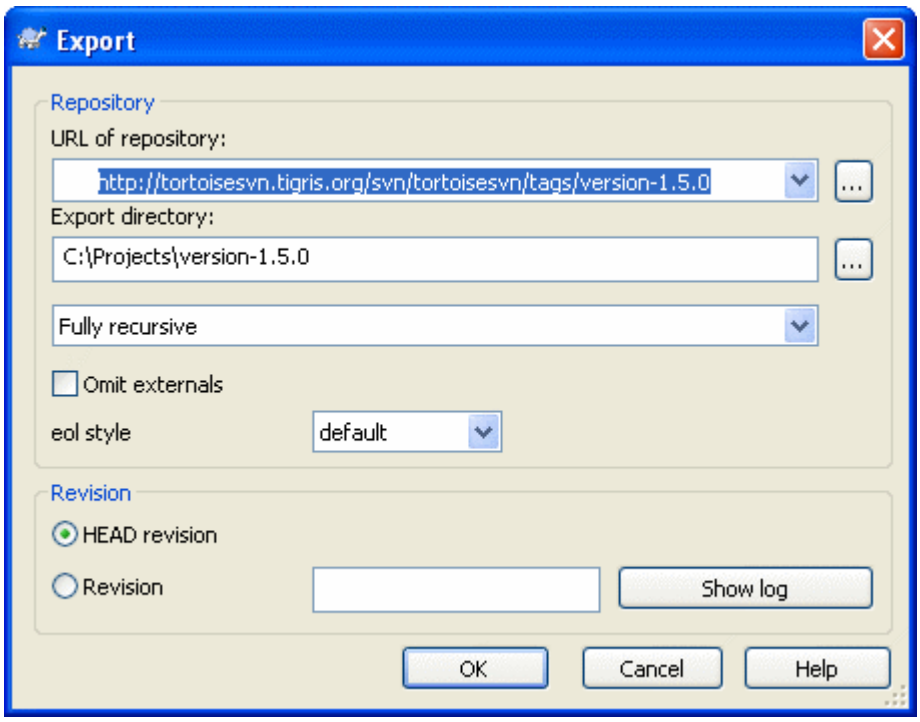


Figure 4.47. La fenêtre extraction-depuis-une-URL

If you execute this command on an unversioned folder, TortoiseSVN will assume that the selected folder is the target, and open a dialog for you to enter the URL and revision to export from. This dialog has options to export only the top level folder, to omit external references, and to override the line end style for files which have the `svn:eol-style` property set.

Bien sûr, vous pouvez aussi exporter directement depuis le référentiel. Utilisez l'explorateur de référentiel pour naviguer au sous-arbre pertinent dans votre référentiel, utilisez ensuite Menu contextuel → Exporter. Vous obtiendrez la boîte de dialogue Exporter depuis l'URL décrite ci-dessus.

If you execute this command on your working copy you'll be asked for a place to save the *clean* working copy without the `.svn` folders. By default, only the versioned files are exported, but you can use the Export unversioned files too checkbox to include any other unversioned files which exist in your WC and not in the repository. External references using `svn:externals` can be omitted if required.

Another way to export from a working copy is to right drag the working copy folder to another location and choose Context Menu → SVN Export here or Context Menu → SVN Export all here. The second option includes the unversioned files as well.

When exporting from a working copy, if the target folder already contains a folder of the same name as the one you are exporting, you will be given the option to overwrite the existing content, or to create a new folder with an automatically generated name, eg. `Target (1)`.

Extraire quelques fichiers

La boite de dialogue d'exportation ne permet pas d'extraire juste quelques fichiers, même si Subversion le permet.

To export single files with TortoiseSVN, you have to use the repository browser (Section 4.24, « l'explorateur de référentiel »). Simply drag the file(s) you want to export from the repository browser to where you want them in the explorer, or use the context menu in the repository browser to export the files.

Exporter une Arborescence des Modifications

If you want to export a copy of your project tree structure but containing only the files which have changed in a particular revision, or between any two revisions, use the compare revisions feature described in Section 4.10.3, « Comparer des répertoires ».

4.26.1. Retirer une copie de travail du contrôle de version

Sometimes you have a working copy which you want to convert back to a normal folder without the `.svn` directories. What you really need is an export-in-place command, that just removes the control directories rather than generating a new clean directory tree.

The answer is surprisingly simple - export the folder to itself! TortoiseSVN detects this special case and asks if you want to make the working copy unversioned. If you answer *yes* the control directories will be removed and you will have a plain, unversioned directory tree.

4.27. Relocaliser une copie de travail

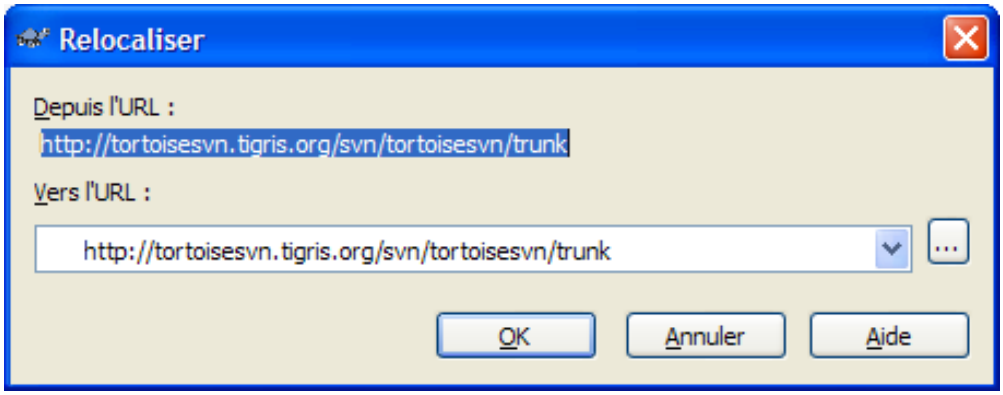


Figure 4.48. La boîte de dialogue Relocaliser

If your repository has for some reason changed it's location (IP/URL). Maybe you're even stuck and can't commit and you don't want to checkout your working copy again from the new location and to move all your changed data back into the new working copy, TortoiseSVN → Relocate is the command you are looking for. It basically does very little: it scans all `entries` files in the `.svn` folder and changes the URL of the entries to the new value.

You may be surprised to find that TortoiseSVN contacts the repository as part of this operation. All it is doing is performing some simple checks to make sure that the new URL really does refer to the same repository as the existing working copy.

Avertissement

C'est une opération très rare. La commande relocaliser est utilisée *seulement* si l'URL de la racine du référentiel a changé. Les raisons possibles sont :

- L'adresse IP du serveur a changé.
- Le protocole a changé (par exemple http:// en https: //).
- Le chemin de la racine du référentiel a changé dans le paramétrage du serveur.

> Autrement, vous devez relocaliser quand votre copie de travail se réfère au même emplacement dans le même référentiel, mais le référentiel lui-même a été déplacé.
>
> Elle ne s'applique pas si :
>
> - Vous voulez vous déplacer vers un référentiel Subversion différent. Dans ce cas, vous devriez exécuter une extraction fraîche à partir du nouvel emplacement du référentiel.
> - Vous voulez aller sur une branche différente ou sur un répertoire dans le même référentiel.
>
>   Pour ce faire, vous devriez utiliser TortoiseSVN → Aller sur.... Lisez Section 4.19.2, « Extraire ou aller sur... » pour plus d'informations.
>
> Si vous utilisez relocaliser dans n'importe lequel des cas ci-dessus, cela *corrompra votre copie de travail* et vous obtiendrez beaucoup de messages d'erreur inexplicables en mettant à jour, en livrant, etc. Une fois que cela s'est produit, la seule correction est une extraction fraiche.

4.28. Intégration avec des systèmes de bug tracking / traqueurs d'incidents

Il est très fréquent dans le développement logiciel que les changements soient liés à un bug spécifique ou à un ID d'incident. Les utilisateurs de systèmes de traque de bug (traqueurs d'incidents) voudraient associer les changements qu'ils font dans Subversion avec un ID spécifique dans leur traqueur d'incidents. La plupart des traqueurs fournissent donc un script hook de pre-commit qui analyse syntaxiquement le commentaire pour trouver l'ID du bug auquel la livraison est associée. C'est quelque peu sujet aux erreurs puisque cela compte sur l'utilisateur pour écrire le commentaire correctement pour que le script hook de pre-commit puisse en faire l'analyse syntaxique correctement.

TortoiseSVN peut aider l'utilisateur de deux manières :

1. Quand l'utilisateur entre un commentaire, une ligne bien définie incluant le numéro de l'incident associé à la livraison peut être ajoutée automatiquement. Cela réduit le risque que l'utilisateur entre le numéro de l'incident d'une façon que les outils de traque de bugs ne peuvent pas analyser correctement.

   Ou TortoiseSVN peut mettre en évidence la partie du commentaire saisi qui est reconnu par le traqueur d'incidents. De cette façon, l'utilisateur sait que le commentaire peut être correctement analysé syntaxiquement.

2. Quand l'utilisateur parcourt les commentaires, TortoiseSVN crée un lien à partir de chaque ID de bug dans le commentaire qui lance le navigateur à l'incident mentionné.

4.28.1. Ajouter des numéros de bugs aux messages de log

You can integrate a bug tracking tool of your choice in TortoiseSVN. To do this, you have to define some properties, which start with `bugtraq:`. They must be set on Folders: (Section 4.17, « Configuration des projets »)

There are two ways to integrate TortoiseSVN with issue trackers. One is based on simple strings, the other is based on *regular expressions*. The properties used by both approaches are:

bugtraq:url
: Set this property to the URL of your bug tracking tool. It must be properly URI encoded and it has to contain `%BUGID%`. `%BUGID%` is replaced with the Issue number you entered. This allows TortoiseSVN to display a link in the log dialog, so when you are looking at the revision log you can jump directly to your bug tracking tool. You do not have to provide this property, but then TortoiseSVN shows only the issue number and not the link to it. e.g the TortoiseSVN project is using `http://issues.tortoisesvn.net/?do=details&id=%BUGID%`

You can also use relative URLs instead of absolute ones. This is useful when your issue tracker is on the same domain/server as your source repository. In case the domain name ever changes, you don't have to adjust the `bugtraq:url` property. There are two ways to specify a relative URL:

If it begins with the string `^/` it is assumed to be relative to the repository root. For example, `^/../?do=details&id=%BUGID%` will resolve to `http://tortoisesvn.net/?do=details&id=%BUGID%` if your repository is located on `http://tortoisesvn.net/svn/trunk/`.

A URL beginning with the string `/` is assumed to be relative to the server's hostname. For example `/?do=details&id=%BUGID%` will resolve to `http://tortoisesvn.net/?do=details&id=%BUGID%` if your repository is located anywhere on `http://tortoisesvn.net`.

bugtraq:warnifnoissue
: Set this to `true`, if you want TortoiseSVN to warn you because of an empty issue-number text field. Valid values are `true/false`. *If not defined, `false` is assumed.*

4.28.1.1. Numéro de Bug dans un Champ Texte

Dans l'approche simple, TortoiseSVN montre à l'utilisateur un champ de saisie séparé où un ID de bug peut être entré. Une ligne séparée est alors ajoutée avant/après le commentaire que l'utilisateur a saisi.

bugtraq:message
: This property activates the bug tracking system in *Input field* mode. If this property is set, then TortoiseSVN will prompt you to enter an issue number when you commit your changes. It's used to add a line at the end of the log message. It must contain `%BUGID%`, which is replaced with the issue number on commit. This ensures that your commit log contains a reference to the issue number which is always in a consistent format and can be parsed by your bug tracking tool to associate the issue number with a particular commit. As an example you might use `Issue : %BUGID%`, but this depends on your Tool.

bugtraq:append
: Cette propriété définit si l'ID-bug est ajouté (true) à la fin du commentaire ou inséré (false) au début du commentaire. Les valeurs valables sont `true/false`. *Si elle n'est pas définie, `true` est par défaut, pour que les projets existants ne cassent pas.*

bugtraq:label
: This text is shown by TortoiseSVN on the commit dialog to label the edit box where you enter the issue number. If it's not set, `Bug-ID / Issue-Nr:` will be displayed. Keep in mind though that the window will not be resized to fit this label, so keep the size of the label below 20-25 characters.

bugtraq:number
: If set to `true` only numbers are allowed in the issue-number text field. An exception is the comma, so you can comma separate several numbers. Valid values are `true/false`. *If not defined, `true` is assumed.*

4.28.1.2. Numéros de Bug en Utilisant des Expressions Régulières

In the approach with *regular expressions*, TortoiseSVN doesn't show a separate input field but marks the part of the log message the user enters which is recognized by the issue tracker. This is done while the user writes the log message. This also means that the bug ID can be anywhere inside a log message! This method is much more flexible, and is the one used by the TortoiseSVN project itself.

bugtraq:logregex
: Cette propriété active le système de bug tracking en mode *Regex*. Elle contient soit une expression régulière, soit deux séparées par un retour à la ligne.

If two expressions are set, then the first expression is used as a pre-filter to find expressions which contain bug IDs. The second expression then extracts the bare bug IDs from the result of the first

regex. This allows you to use a list of bug IDs and natural language expressions if you wish. e.g. you might fix several bugs and include a string something like this: « This change resolves issues #23, #24 and #25 »

Si vous voulez isoler les ID des bugs tels qu'utilisés à l'intérieur des messages de log, vous pouvez utiliser utiliser les espressions régulières suivantes, qui sont celles utilisées par TortoiseSVN en personne : `[Ii]ssues?:?(\s*(,|and)?\s*#\d+)+` et `(\d+)`

The first expression picks out « issues #23, #24 and #25 » from the surrounding log message. The second regex extracts plain decimal numbers from the output of the first regex, so it will return « 23 », « 24 » and « 25 » to use as bug IDs.

Breaking the first regex down a little, it must start with the word « issue », possibly capitalised. This is optionally followed by an « s » (more than one issue) and optionally a colon. This is followed by one or more groups each having zero or more leading whitespace, an optional comma or « and » and more optional space. Finally there is a mandatory « # » and a mandatory decimal number.

If only one expression is set, then the bare bug IDs must be matched in the groups of the regex string. Example: `[Ii]ssue(?:s)? #?(\d+)` This method is required by a few issue trackers, e.g. trac, but it is harder to construct the regex. We recommend that you only use this method if your issue tracker documentation tells you to.

If you are unfamiliar with regular expressions, take a look at the introduction at *http://en.wikipedia.org/wiki/Regular_expression*, and the online documentation and tutorial at *http://www.regular-expressions.info/*.

Si les deux propriétés `bugtraq:message` et `bugtraq:logregex` sont définies, `logregex` a la priorité.

Astuce

Même si vous n'avez pas de traqueur d'incidents avec un hook pre-commit analysant syntaxiquement vos commentaires, vous pouvez toujours utiliser cela pour transformer les incidents mentionnés dans vos commentaires en liens !

And even if you don't need the links, the issue numbers show up as a separate column in the log dialog, making it easier to find the changes which relate to a particular issue.

Some `tsvn:` properties require a `true/false` value. TortoiseSVN also understands `yes` as a synonym for `true` and `no` as a synonym for `false`.

Mettre les propriétés sur les dossiers

These properties must be set on folders for the system to work. When you commit a file or folder the properties are read from that folder. If the properties are not found there, TortoiseSVN will search upwards through the folder tree to find them until it comes to an unversioned folder, or the tree root (eg. `C:\`) is found. If you can be sure that each user checks out only from e.g `trunk/` and not some sub-folder, then it's enough if you set the properties on `trunk/`. If you can't be sure, you should set the properties recursively on each sub-folder. A property setting deeper in the project hierarchy overrides settings on higher levels (closer to `trunk/`).

For `tsvn:` properties *only* you can use the Recursive checkbox to set the property to all sub-folders in the hierarchy, without also setting it on all files.

No Issue Tracker Information from Repository Browser

Because the issue tracker integration depends upon accessing subversion properties, you will only see the results when using a checked out working copy. Fetching properties remotely is a slow operation, so you will not see this feature in action from the repo browser.

This issue tracker integration is not restricted to TortoiseSVN; it can be used with any Subversion client. For more information, read the full *Issue Tracker Integration Specification* [http://tortoisesvn.tigris.org/svn/tortoisesvn/trunk/doc/issuetrackers.txt] in the TortoiseSVN source repository. (Section 3, « TortoiseSVN est gratuit ! » explains how to access the repository).

4.28.2. Récupérer des Informations depuis un Traqueur de Bug

The previous section deals with adding issue information to the log messages. But what if you need to get information from the issue tracker? The commit dialog has a COM interface which allows integration an external program that can talk to your tracker. Typically you might want to query the tracker to get a list of open issues assigned to you, so that you can pick the issues that are being addressed in this commit.

Any such interface is of course highly specific to your issue tracker system, so we cannot provide this part, and describing how to create such a program is beyond the scope of this manual. The interface definition and sample plugins in C# and C++/ATL can be obtained from the `contrib` folder in the *TortoiseSVN repository* [http://tortoisesvn.tigris.org/svn/tortoisesvn/trunk/contrib/issue-tracker-plugins]. (Section 3, « TortoiseSVN est gratuit ! » explains how to access the repository). A summary of the API is also given in Chapitre 6, *IBugtraqProvider interface* Another (working) example plugin in C# is *Gurtle* [http://code.google.com/p/gurtle/] which implements the required COM interface to interact with the *Google Code* [http://code.google.com/hosting/] issue tracker.

For illustration purposes, let's suppose that your system administrator has provided you with an issue tracker plugin which you have installed, and that you have set up some of your working copies to use the plugin in TortoiseSVN's settings dialog. When you open the commit dialog from a working copy to which the plugin has been assigned, you will see a new button at the top of the dialog.

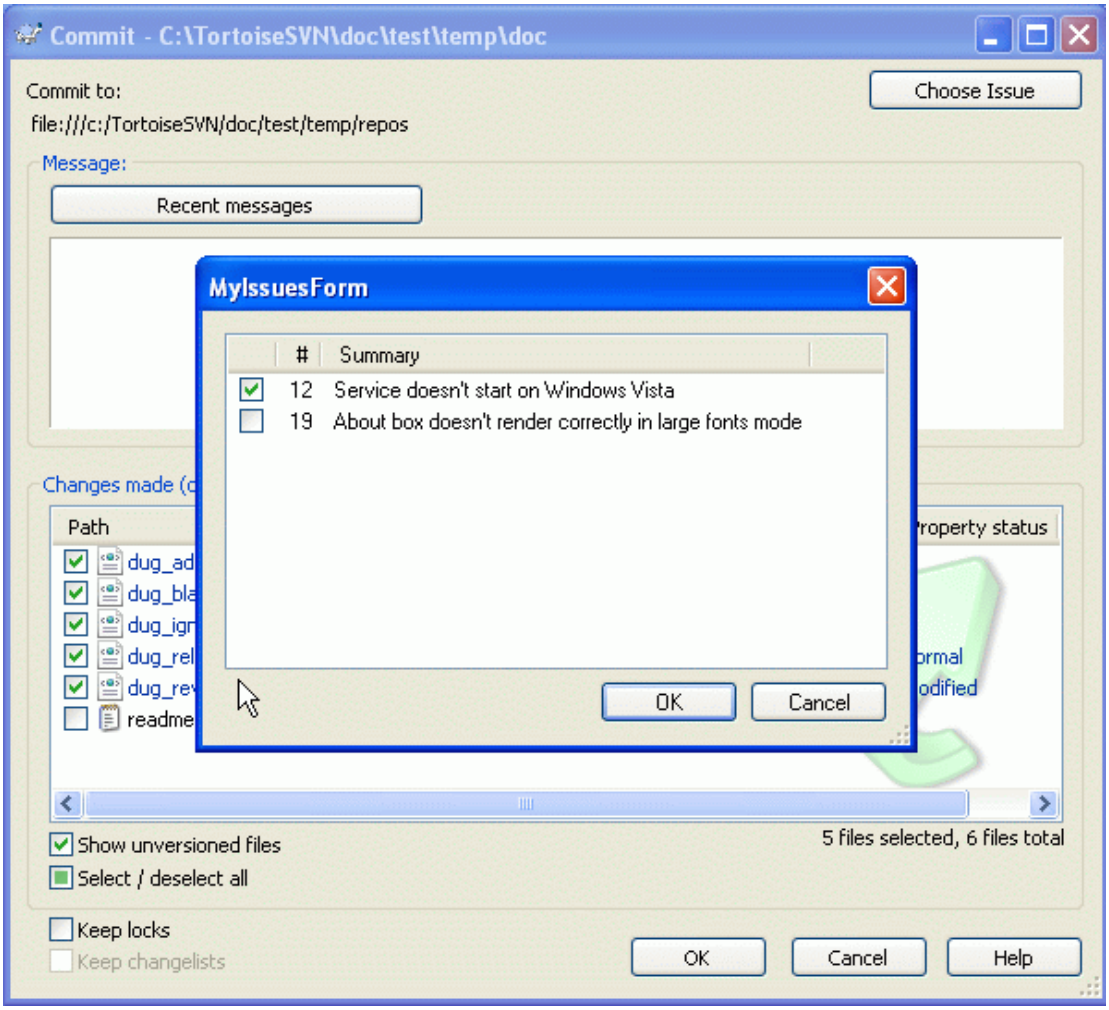


Figure 4.49. Exemple de fenêtre de bug tracker

In this example you can select one or more open issues. The plugin can then generate specially formatted text which it adds to your log message.

4.29. Intégration avec des explorateur de référentiel de type web.

There are several web-based repository viewers available for use with Subversion such as *ViewVC* [http://www.viewvc.org/] and *WebSVN* [http://websvn.tigris.org/]. TortoiseSVN provides a means to link with these viewers.

You can integrate a repo viewer of your choice in TortoiseSVN. To do this, you have to define some properties which define the linkage. They must be set on Folders: (Section 4.17, « Configuration des projets »)

webviewer:revision
: Set this property to the URL of your repo viewer to view all changes in a specific revision. It must be properly URI encoded and it has to contain `%REVISION%`. `%REVISION%` is replaced with the revision number in question. This allows TortoiseSVN to display a context menu entry in the log dialog Context Menu → View revision in webviewer

webviewer:pathrevision
: Set this property to the URL of your repo viewer to view changes to a specific file in a specific revision. It must be properly URI encoded and it has to contain `%REVISION%` and `%PATH%`. `%PATH`

% is replaced with the path relative to the repository root. This allows TortoiseSVN to display a context menu entry in the log dialog Context Menu → View revision and path in webviewer For example, if you right-click in the log dialog bottom pane on a file entry `/trunk/src/file` then the %PATH% in the URL will be replaced with `/trunk/src/file`.

You can also use relative URLs instead of absolute ones. This is useful in case your web viewer is on the same domain/server as your source repository. In case the domain name ever changes, you don't have to adjust the `webviewer:revision` and `webviewer:pathrevision` property. The format is the same as for the `bugtraq:url` property. See Section 4.28, « Intégration avec des systèmes de bug tracking / traqueurs d'incidents ».

Mettre les propriétés sur les dossiers

These properties must be set on folders for the system to work. When you commit a file or folder the properties are read from that folder. If the properties are not found there, TortoiseSVN will search upwards through the folder tree to find them until it comes to an unversioned folder, or the tree root (eg. `C:\`) is found. If you can be sure that each user checks out only from e.g `trunk/` and not some sub-folder, then it's enough if you set the properties on `trunk/`. If you can't be sure, you should set the properties recursively on each sub-folder. A property setting deeper in the project hierarchy overrides settings on higher levels (closer to `trunk/`).

For `tsvn:` properties *only* you can use the Recursive checkbox to set the property to all sub-folders in the hierarchy, without also setting it on all files.

Aucun visualiseur dans l'explorateurde référentiel

Because the repo viewer integration depends upon accessing subversion properties, you will only see the results when using a checked out working copy. Fetching properties remotely is a slow operation, so you will not see this feature in action from the repo browser.

4.30. Configuration de TortoiseSVN

Pour découvrir à quoi servent les réglages, laissez simplement votre pointeur de souris une seconde sur la saisie/coche... et une info-bulle utile apparaîtra.

4.30.1. Configuration générale

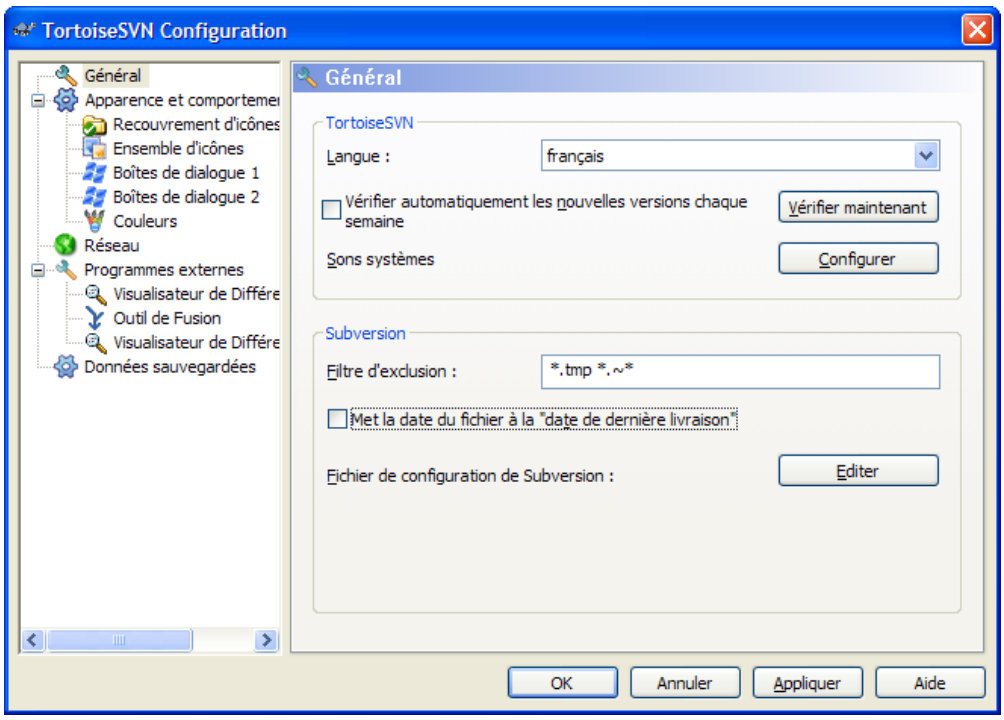


Figure 4.50. La boîte de dialogue Configuration, page Général

Cette boîte de dialogue vous permet de spécifier votre langue préférée et la configuration spécifique à Subversion.

Langue
: Sélectionne la langue de votre interface utilisateur. À quoi d'autre vous attendiez-vous ?

Vérifier automatiquement les nouvelles versions chaque semaine
: Si elle est cochée, TortoiseSVN entrera en contact avec son site de téléchargement une fois par semaine pour voir s'il y a une version plus récente du programme disponible. Utilisez Vérifier maintenant si vous voulez une réponse tout de suite. La nouvelle version ne sera pas téléchargée ; vous recevez simplement une boîte de dialogue d'information vous disant que la nouvelle version est disponible.

Sons système
: TortoiseSVN a trois sons personnalisés qui sont installés par défaut.

  - Erreur
  - Avis
  - Avertissement

  Vous pouvez choisir des sons différents (ou désactiver ces sons complètement) en utilisant le Panneau de Configuration Windows. Configurer est un raccourci vers le Panneau de Configuration.

Modèle d'exclusion global

: Global ignore patterns are used to prevent unversioned files from showing up e.g. in the commit dialog. Files matching the patterns are also ignored by an import. Ignore files or directories by typing in the names or extensions. Patterns are separated by spaces e.g. `bin obj *.bak *.~?? *.jar *.[Tt]mp`. These patterns should not include any path separators. Note also that there is no way to differentiate between files and directories. Read Section 4.13.1, « L'utilisation des pattern matching dans la liste des fichier à ignorer » for more information on the pattern-matching syntax.

Notez que les modèles d'exclusion que vous spécifiez ici affecteront aussi les autres clients Subversion fonctionnant sur votre PC, y compris le client en ligne de commande.

Attention

Si vous utilisez le fichier de configuration de Subversion pour définir un modèle `global-ignores`, il ignorera les réglages que vous faites ici. Le fichier de configuration de Subversion est accessible en utilisant Éditer comme décrit ci-dessous.

Ce modèle d'exclusion affectera tous vos projets. Ce n'est pas versionné, donc il n'affectera pas les autres utilisateurs. En comparaison, vous pouvez aussi utiliser la propriété versionnée `svn:ignore` pour exclure des fichiers ou des répertoires du contrôle de version. Lisez Section 4.13, « Ignorer des fichiers et des répertoires » pour plus d'informations.

Remplace la date des fichiers par la « date de dernière livraison »
: This option tells TortoiseSVN to set the file dates to the last commit time when doing a checkout or an update. Otherwise TortoiseSVN will use the current date. If you are developing software it is generally best to use the current date because build systems normally look at the date stamps to decide which files need compiling. If you use « last commit time » and revert to an older file revision, your project may not compile as you expect it to.

Fichier de configuration de Subversion
: Use Edit to edit the Subversion configuration file directly. Some settings cannot be modified directly by TortoiseSVN, and need to be set here instead. For more information about the Subversion `config` file see the *Runtime Configuration Area* [http://svnbook.red-bean.com/en/1.5/svn.advanced.confarea.html]. The section on *Automatic Property Setting* [http://svnbook.red-bean.com/en/1.5/svn.advanced.props.html#svn.advanced.props.auto] is of particular interest, and that is configured here. Note that Subversion can read configuration information from several places, and you need to know which one takes priority. Refer to *Configuration and the Windows Registry* [http://svnbook.red-bean.com/en/1.5/svn.advanced.confarea.html#svn.advanced.confarea.windows-registry] to find out more.

Utiliser les répertoires `_svn` au lieu de `.svn`
: VS.NET quand il est utilisé avec des projets web ne peut pas manipuler les répertoires `.svn` que Subversion utilise pour stocker son information interne. Ce n'est pas un bug de Subversion. Le bug est dans VS.NET et les extensions frontpage qu'il utilise. Lisez Section 4.30.11, « Dossiers de travail de Subversion » pour en savoir plus sur cet incident.

  Si vous voulez changer le comportement de Subversion et de TortoiseSVN, vous pouvez utiliser cette case à cocher pour définir la variable d'environnement qui contrôle cela.

  Vous devriez noter que changer cette option ne convertira pas automatiquement les copies de travail existantes pour utiliser le nouveau répertoire d'administration. Vous devrez le faire vous-même en utilisant un script (Voir notre FAQ) ou en extrayant simplement une copie de travail récente.

4.30.1.1. Paramètres du Menu Contextuel

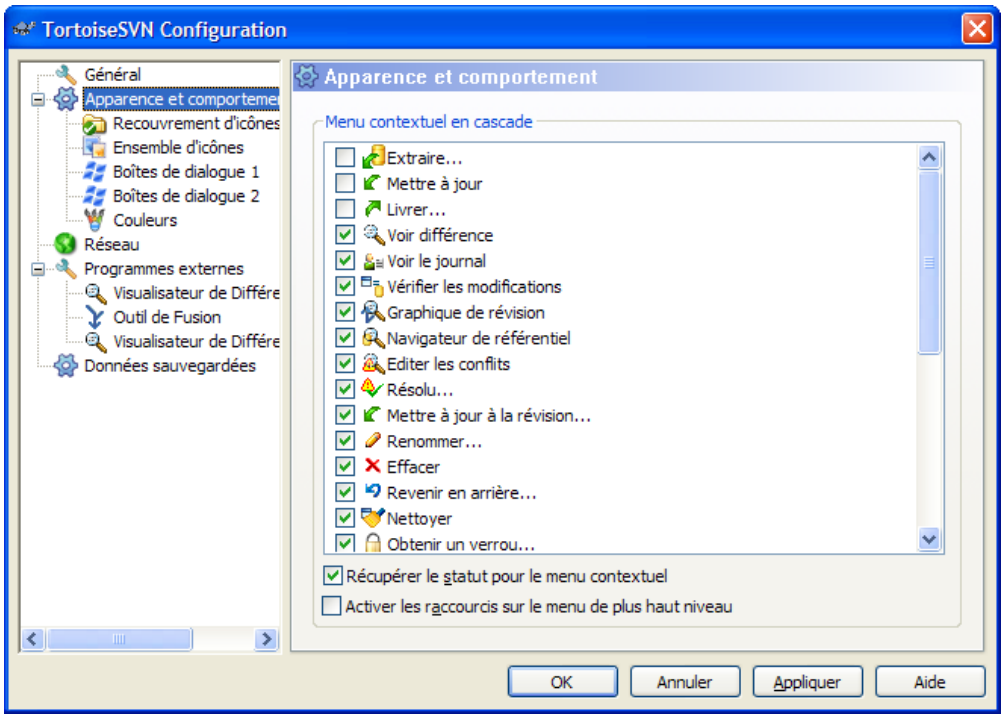


Figure 4.51. La boîte de dialogue de Configuration, Page Menu Contextuel

Cette page vous permet de spécifier quelles entrées du menu contextuel de TortoiseSVN s'afficheront dans le menu contextuel principal et lesquelles apparaîtront dans le sous-menu de TortoiseSVN. Par défaut, la plupart des éléments sont décochés et apparaissent dans le sous-menu.

Il existe un cas spécial pour Obtenir un verrou. Vous pouvez bien sûr le promouvoir au niveau supérieur en utilisant la liste ci-dessus mais puisque la plupart des fichiers n'ont pas besoin du verrouillage, cela ajoute juste du désordre. Cependant, un fichier avec la propriété `svn:needs-lock` nécessite cette action à chaque fois qu'il est modifié, donc dans ce cas, il est très utile de l'avoir au premier niveau. Ici, cocher la case signifie que lorsqu'un fichier ayant la propriété `svn:needs-lock` est sélectionné, Obtenir un verrou apparaitra toujours au premier niveau.

S'il y a quelques chemins sur votre ordinateur où vous ne voulez pas qu'apparaisse le menu contextuel de TortoiseSVN, vous pouvez les lister dans la zone en bas.

4.30.1.2. Réglages des boîtes de dialogues TortoiseSVN 1

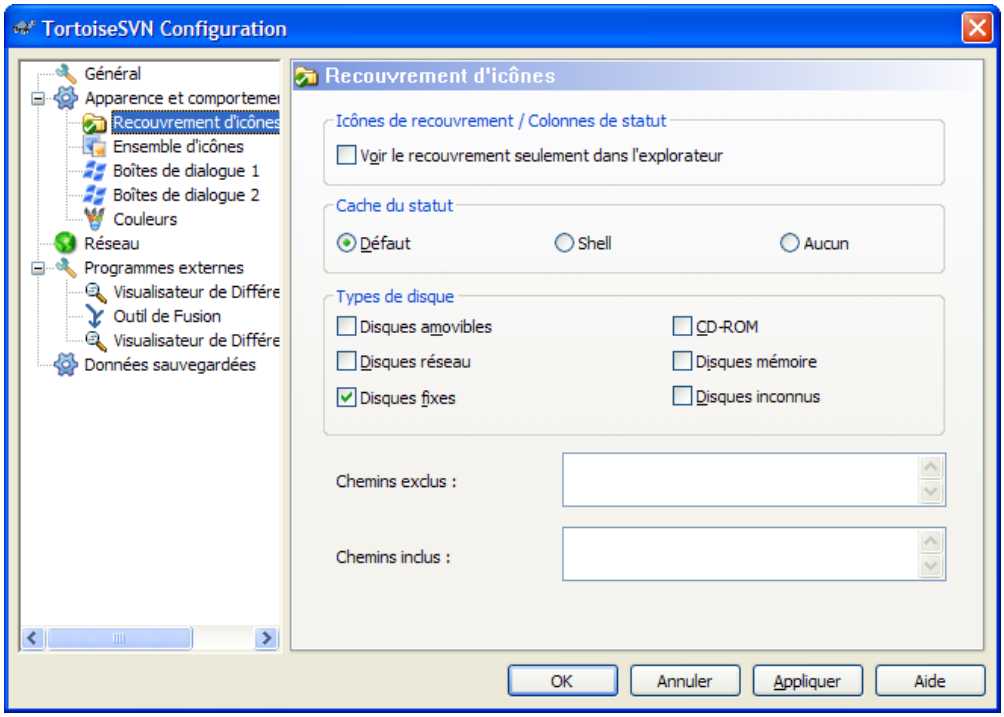


Figure 4.52. La boîte de dialogue Configuration, page Boîtes de dialogue 1

Cette boîte de dialogue vous permet de configurer certaines des boîtes de dialogue de TortoiseSVN de la façon que vous préferez.

Nombre par défaut de commentaires
: Limits the number of log messages that TortoiseSVN fetches when you first select TortoiseSVN → Show Log Useful for slow server connections. You can always use Show All or Next 100 to get more messages.

Police des commentaires
: Sélectionne le type et la taille de la police de caractères utilisée pour afficher le commentaire lui-même dans le panneau du milieu de la boîte de dialogue du Journal de révision et lors de la rédaction des commentaires dans la boîte de dialogue Livrer.

Format court des dates dans les commentaires
: Si les longs messages standards prennent trop place sur votre écran, utilisez le format court.

Vous pouvez double cliquer sur la liste des messages de log pour comparer avec la version précédente
: Si vous vous trouvez régulièrement à comparer les révisions dans le panneau supérieur de la fenêtre de commentaires, vous pouvez utiliser cette option pour exécuter cette action suite à un double-click. Elle n'est pas activée par défaut car consulter les différences est souvent un long processus et beaucoup de personnes préfèrent éviter l'attente suite à un double-click accidentel.

Boîte de dialogue de progression
: TortoiseSVN peut fermer automatiquement toutes les boîte de dialogues de progression quand l'action s'est terminée sans erreur. Ce réglage vous permet de choisir les conditions pour fermer les boîtes de dialogues. Le réglage par défaut est Fermeture manuelle ce qui vous permet de passer en revue tous les messages et de contrôler ce qui s'est passé. Cependant, vous pouvez décider que vous voulez ignorer quelques types de message et vouloir que la boîte de dialogue se ferme automatiquement s'il n'y a aucun changement critique.

  Fermeture automatique s'il n'a y pas eu de fusions, d'ajouts ou de suppression signifie que la boîte de dialogue de progression se fermera s'il y a que de simples mises à jour, mais si les

changements du référentiel ont été fusionnés avec les vôtres, ou si des fichiers ont été ajoutés ou supprimés, la boîte de dialogue restera ouverte. Elle restera aussi ouverte s'il n'y a pas eu de conflits ou d'erreurs pendant l'opération.

Fermeture automatique pour les opérations locales signifie que la boîte de dialogue de progression se fermera comme pour Fermeture automatique s'il n'a y pas eu de fusions, d'ajouts ou de suppression mais seulement pour les opérations locales comme ajouter des fichiers ou revenir en arrière. Pour les opérations distantes, la boîte de dialogue restera ouverte.

Fermeture automatique s'il n'y a pas de conflit assoupli les critères un peu plus et fermera la boîte de dialogue même s'il y a eu des fusions, des ajouts ou des suppressions. Cependant, s'il y a des conflits ou des erreurs, la boîte de dialogue reste ouverte.

Fermeture automatique s'il n'y a pas d'erreur ferme toujours la boîte de dialogue même s'il y a eu des conflits. La seule condition qui maintient la boîte de dialogue ouverte est un cas d'erreur, qui se produit quand Subversion est incapable d'achever la tâche. Par exemple, une mise à jour échoue parce que le serveur est inaccessible, ou une livraison échoue parce que la copie de travail est périmée.

Utiliser la poubelle lors d'un retour en arrière
: Lorsque vous annulez des modifications locales, vos changements sont oubliés. TortoiseSVN vous donne un filet de sécurité supplémentaire en envoyant le fichier modifié à la corbeille avant de rendre la copie primitive. Si vous préférez ne pas passer par la corbeille, décochez cette option.

Utiliser l'URL de la WC comme valeur par défaut pour l'URL « From: »
: Dans la boîte de dialogue de fusion, le comportement par défaut est de mémoriser l'URL De : entre les fusions. Cependant, certaines personnes aiment exécuter les fusions depuis différents endroits de leur hiérarchie et trouvent plus facile de partir avec l'URL de la copie de travail courante. Elle peut alors être éditée pour se référer à un chemin parallèle sur une autre branche.

Chemin d'extraction par défaut
: Vous pouvez spécifier le chemin par défaut pour les extractions. Si vous gardez toutes vos extractions à un seul endroit, il est utile d'avoir le disque et le dossier pré-rempli pour que vous n'ayez plus qu'à ajouter le nouveau nom du dossier à la fin.

URL d'extraction par défaut
: Vous pouvez aussi spécifier le chemin par défaut l'URL par défaut des extractions. Si vous extrayez souvent des sous-projets d'un très gros projet, il peut être utile d'avoir l'URL pré-remplie pour que vous n'ayez plus qu'à ajouter le nom du sous-projet à la fin.

4.30.1.3. Réglages des boîtes de dialogues TortoiseSVN 2

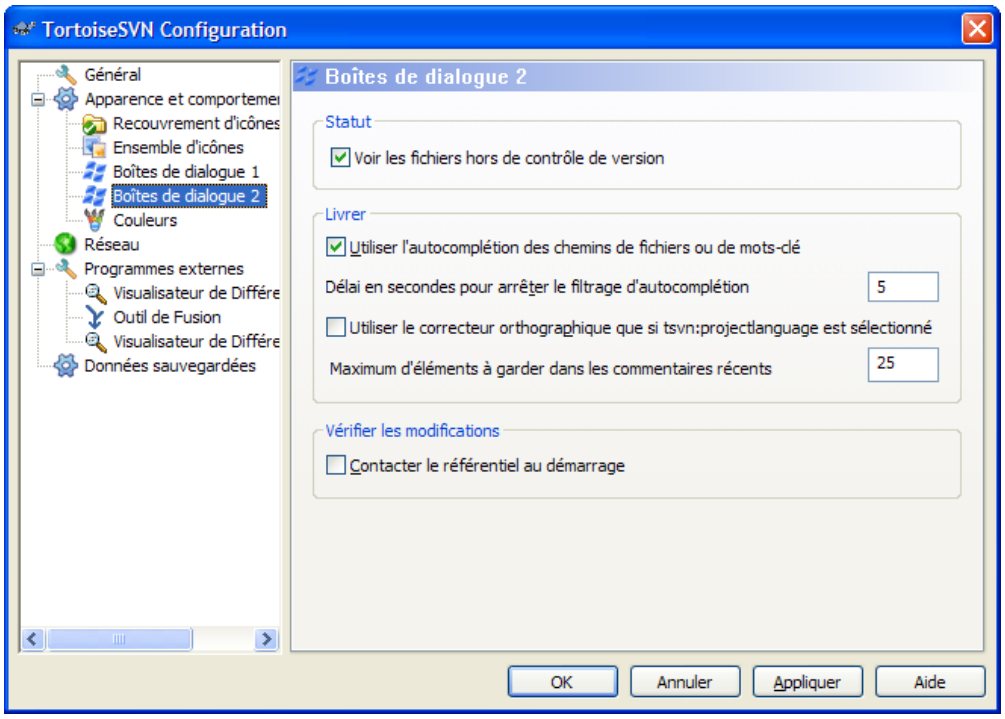


Figure 4.53. La boîte de dialogue Configuration, page Boîtes de dialogue 2

Parcourir récursivement les répertoires non versionnés
: Si cette case est cochée (l'état par défaut), alors à chaque fois que le statut d'un dossier non versionné est affiché dans les boîtes de dialogue Ajouter, Livrer ou Vérifier les modifications, tous les fichiers et tous les dossiers enfants sont aussi affichés. Si vous décochez cette case, seul le parent non versionné est affiché. Décocher réduit le désordre dans ces boîtes de dialogue. Dans ce cas, si vous sélectionnez un dossier non versionné à Ajouter, il est ajouté récursivement.

Utiliser la complétion automatique des chemins de fichiers et des mots-clé
: La boîte de dialogue de livraison inclut une fonction pour analyser syntaxiquement la liste de noms de fichier à livrer. Quand vous tapez les 3 premières lettres d'un élément dans la liste, la boîte de complétion automatique s'ouvre et vous pouvez appuyer sur Entrée pour compléter le nom du fichier. Cochez la case pour activer cette fonctionnalité.

Délai en secondes après lequel arrêter le parcours des données de complétion automatique
: The auto-completion parser can be quite slow if there are a lot of large files to check. This timeout stops the commit dialog being held up for too long. If you are missing important auto-completion information, you can extend the timeout.

Utiliser le correcteur orthographique que si `tsvn:projectlanguage` est défini
: Si vous ne voulez pas utiliser le vérificateur d'orthographe pour toutes les livraisons, cochez cette case. Le vérificateur d'orthographe sera toujours activé quand les propriétés du projet l'exigent.

Maximum d'éléments à garder dans les commentaires récents
: When you type in a log message in the commit dialog, TortoiseSVN stores it for possible re-use later. By default it will keep the last 25 log messages for each repository, but you can customize that number here. If you have many different repositories, you may wish to reduce this to avoid filling your registry.

  Notez que cette configuration s'appliques uniquement aux messages que vous créez sur cet ordinateur. Cela n'a aucune relation avec le cache des commentaires.

Ré-ouvrir la boîte de dialogue de livraison après qu'une livraison ait échoué
: Quand une livraison échoue pour une raison particulière (la copie de travail doit être mise à jour, les hooks pre-commit refusent la livraison, erreur réseau, etc), vous pouvez choisir cette option pour conserver la boîte de dialogue Livrer ouverte prête à recommencer. Cependant, vous devez prendre conscience que cela peut causer des problèmes. Si l'échec signifie que vous devez mettre à jour et que la mise à jour entraîne des conflits, vous devrez d'abord les résoudre.

Sélectionner les éléments automatiquement
: Le comportement normal dans la fenêtre de livraison est la sélection automatique de tous les éléments modifiés (et versionnés) en vue d'une livraison. Si vous préférez démarrer sans aucune sélection et choisir les éléments manuellement, décochez cette case.

Contacter le référentiel au démarrage
: La boîte de dialogue Vérifier les modifications vérifie la copie de travail par défaut et entre seulement en contact avec le référentiel quand vous cliquez sur Vérifier le référentiel. Si vous voulez toujours vérifier le référentiel, vous pouvez utiliser ce réglage pour que cette action se fasse automatiquement.

Afficher la boîte de dialogue de verrouillage avant de verrouiller des fichiers
: Lorsque vous sélectionnez u ou plusieurs fichiers, puis que vous utilisez TortoiseSVN → Verouillage pour retirer un verrou sur ces fichiers, sur certains projet il est demandé d'ajouter un message expliquant pourquoi vous avez verrouillé ces fichiers. Si vous n'utilisez pas de message de verrouillage, vous pouvez décocher la case afin de passer cette fenêtre et de verrouiller les fichiers immédiatement.

  Si vous utilisez la commande verrouiller sur un dossier, une fenêtre s'ouvrira vous permettant de sélectionner d'autres fichiers à verrouiller.

  If your project is using the `tsvn:lockmsgminsize` property, you will see the lock dialog regardless of this setting because the project *requires* lock messages.

4.30.1.4. Configuration des couleurs de TortoiseSVN

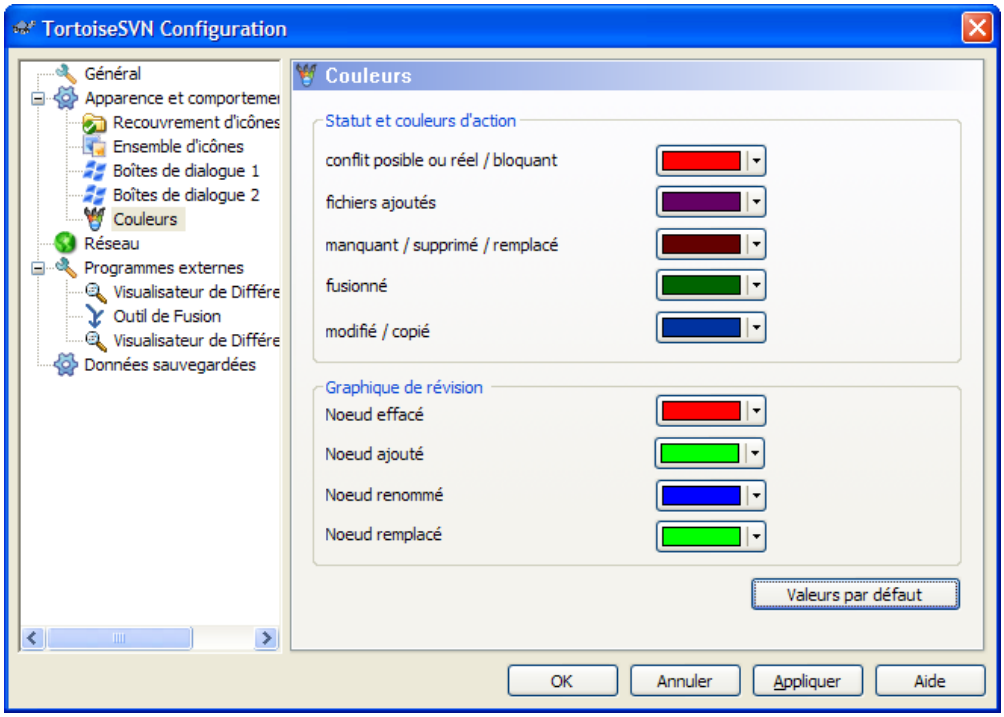


Figure 4.54. La boîte de dialogue Configuration, page Couleurs

Cette boîte de dialogue vous permet de configurer les couleurs de texte utilisées dans les boîtes de dialogue de TortoiseSVN de la façon que vous préferez.

Conflit possible ou réel / bloquant
: Un conflit s'est produit pendant la mise à jour, ou peut arriver pendant une fusion. La mise à jour est entravée par un fichier/dossier non versionné existant du même nom qu'un fichier versionné.

  Cette couleur est aussi utilisée pour des messages d'erreur dans les boîtes de dialogues de progression.

Fichiers ajoutés
: Éléments ajoutés au référentiel.

Manquant / supprimé / remplacé
: Éléments supprimés du référentiel, manquants de la copie de travail, ou supprimés de la copie de travail et remplacés par d'autres fichiers du même nom.

Fusionné
: Changements du référentiel fusionnés avec succès dans la CdT sans créer de conflits.

Modifié / copié
: Ajoutés avec l'historique, ou chemins copiés dans le référentiel. Aussi utilisé dans la boîte de dialogue de journal pour les entrées qui incluent des élements copiés.

Noeud Supprimé
: Un élément qui a été supprimé du référentiel.

Noeud ajouté
: Un élément ajouté au référentiel, par un ajout, une copie ou un déplacement.

Noeud renommé
: Un élément qui a été renommé dans le référentiel.

Noeud remplacé
: L'élément original a été supprimé et un nouvel élément avec le même nom le remplace.

4.30.2. Options du Graphe des Révisions

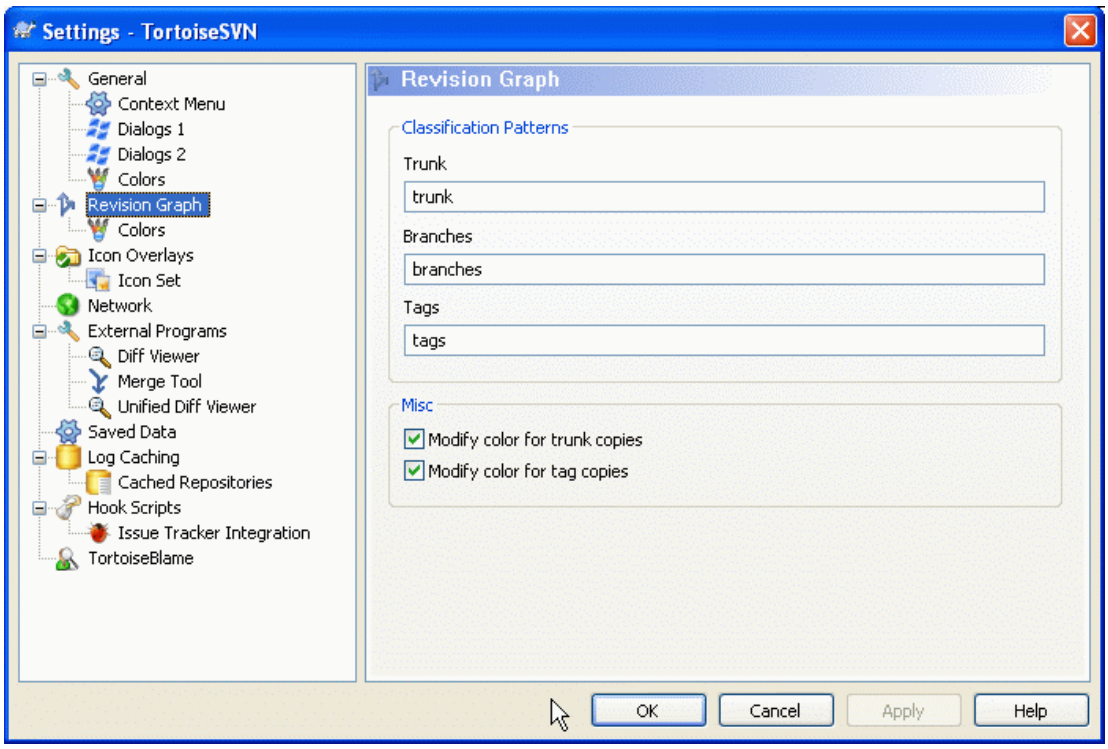


Figure 4.55. La boîte de dialogue Configuration, page graphique de révision

Filtres de classification
: Le graphe de révision tente de montrer une image plus claire de la structure de votre dépôt en distinguant le trunk, les branches et les tags. Puisqu'il n'existe pas de telle classidication construite dans Subversion, cette information est extraite à partir des chemins. La configuration par défaut suppose que vous utilisez les noms englais conventionnels comme il est suggéré dans la documentation Subversion, mais bien sûr, votre utilisation peut varier.

  Spécifiez les patrons utilisées pour reconnaître ces chemins dans les trois champs prévus à cette effet. Les patrons sont insensibles à la casse, mais vous devez les spécifier en minuscule. Les caractères spéciaux `*` et `?` fonctionneront comme d'habitude. Vous pouvez utiliser `;` pour séparer plusieurs patrons. N'incluez pas d'espaces supplémentaires comme ils seraient inclus dans la spécification correspondante.

Modifier les Couleurs
: Les couleurs sont utilisées dans le graphe de révision pour indiquer le type de noeud, c'est à dire si le noeud a été ajouté, supprimé ou supprimé. Afin de faciliter la classification des noeuds, vous pouvez permettre au graphe de mélanger les couleurs pour donner une identification à la fois au type et à la classification des noeuds. Si la case est cochée, le mélange sera utilisé. Dans le cas contraire, la couleur est utilisée pour indiquer uniquement le type du noeud. Utilisez cette fenêtre de sélection de couleurs pour allouer les couleurs spécifiques utilisées.

4.30.2.1. Couleurs du Graphes de Révision

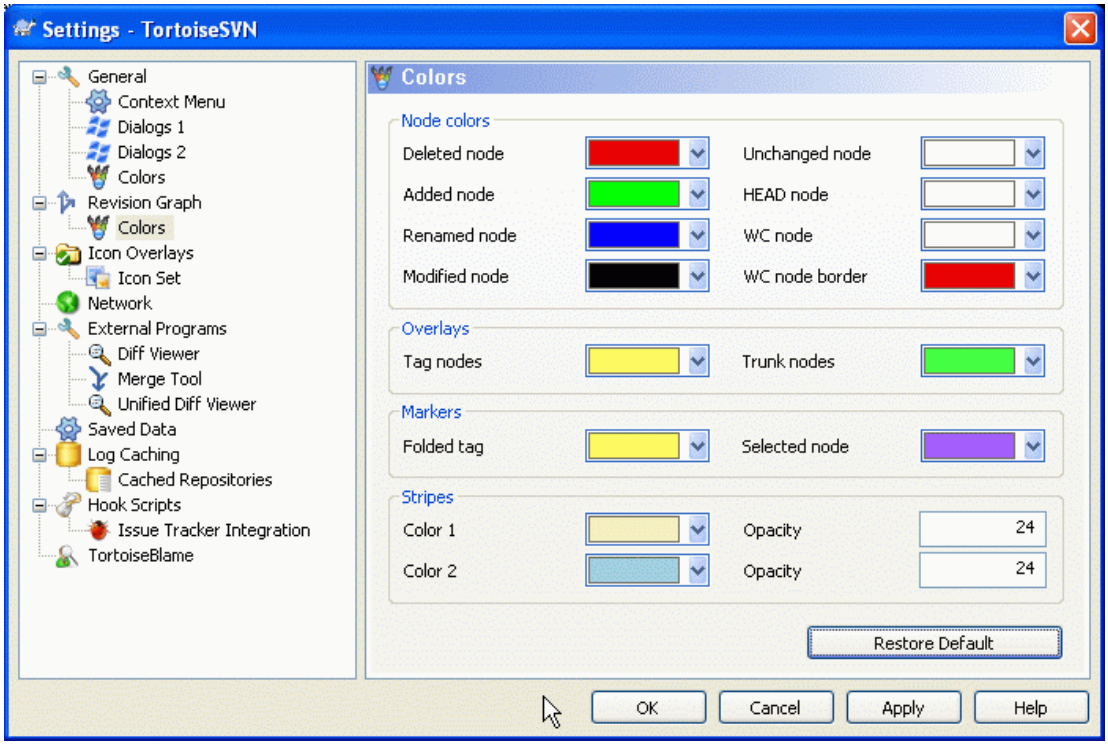


Figure 4.56. La boîte de dialogue Configuration, page des couleur du graphe de révision

Cette page vous permet de configurer les couleurs utilisées. Notez bien que les couleurs spécifiées ici sont des couleurs pleines. La plupart des noeuds sont colorés en utilisant un mélange de la couleur de leur type, de celle du fond d'écran et éventuellement, de la couleur de classification

Noeud Supprimé
: Eléments ayant été supprimés et non copiés ailleurs dans la même révision.

Noeud Ajouté
: Elements ajoutés récemment, ou copiés (ajout avec l'historique).

Noeud Renommé
: Eléments supprimés d'un endroit et ajoutés ailleurs dans une même révision.

Noeud Modifié
: Modification simple sans ajout ni suppression.

Noeud inchangé
: Peut être utilisée pour montrer la révision utilisée en tant que source d'une copie, même lorsqu'aucune modification (de l'élément représenté sur le graphe) n'a eut lieu lors de cette révision.

Noeud de tête
: Révision de tête courante dans le référentiel

Noeud de la CdT
: Si vous choisissez de montrer, sur le graphe, un noeud supplémentaire pour votre copie de travail modifiée, attachée à la révision de la dernière livraison, utilisez cette couleur.

Bordure du noeud CdT
: Si vous choisissez d'afficher si la copie de travail est modifiée, utilisez cette bordure de couleur sur le noeud WC lorsque des modifications sont trouvées.

Noeud tag
: Les noeuds classifiés comme étant des tags peuvent être nuancés avec cette couleur.

Noeud racine
: Les noeuds classifiés comme étant des trunk (racines) peuvent être nuancés avec cette couleur.

Marqueurs de Tags Fermés
: Si vous avez l'habitude de replier les étiquettes pour économiser de l'espace, elles sont marquées sur la copie source en utilisant un bloc de cette couleur

Marqueurs du Noeud Sélectionné
: Lorsque vous cliquez sur un noeud pour le sélectionner, le marqueur utilisée pour indiquer la sélection est un bloc de cette couleur.

Rayures
: Ces couleurs sont utilisées lorsque le graphe est divisé en sous-arbres et que le fond est coloré en rayures alternées pour aider à choisir les arbres séparés.

4.30.3. Configuration du recouvrement d'icônes

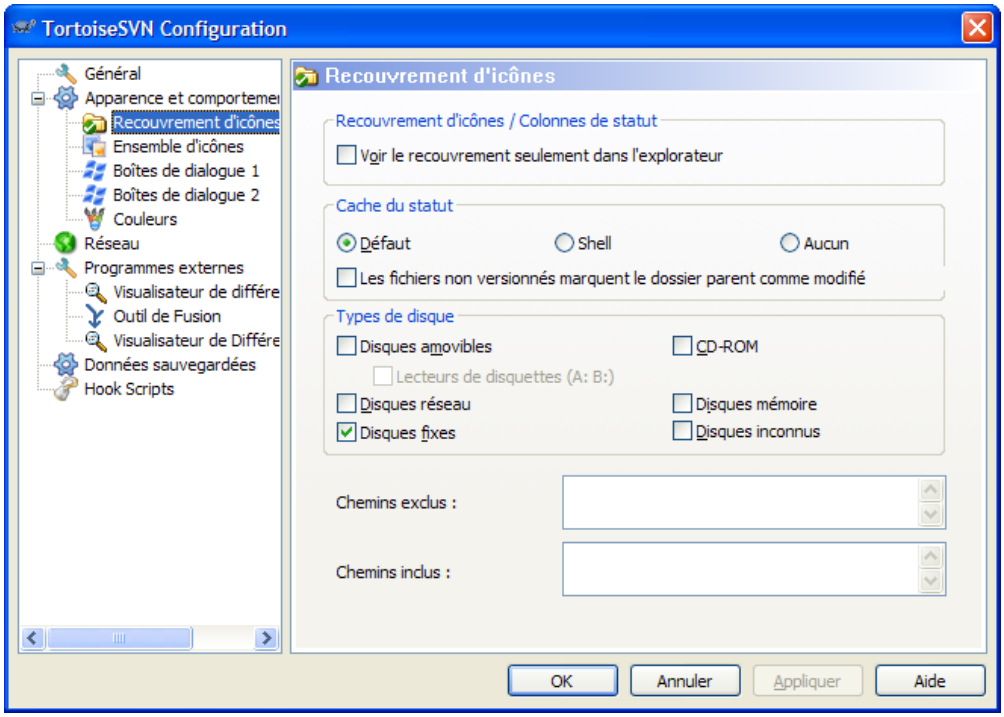


Figure 4.57. La Boîte de Dialogue Configuration, Page des Icônes de Recouvrement

Cette page vous permet de choisir à quels éléments TortoiseSVN doit associer des icônes de recouvrement

By default, overlay icons and context menus will appear in all open/save dialogs as well as in Windows Explorer. If you want them to appear *only* in Windows Explorer, check the Show overlays and context menu only in explorer box.

Les éléments ignorés ainsi que les éléments non versionnés ne donnent généralement généralement pas de recouvrement. Si vous désirez afficher un recouvrement dans ces cas-là, cochez les cases.

You can also choose to mark folders as modified if they contain unversioned items. This could be useful for reminding you that you have created new files which are not yet versioned. This option is only available when you use the *default* status cache option (see below).

Since it takes quite a while to fetch the status of a working copy, TortoiseSVN uses a cache to store the status so the explorer doesn't get hogged too much when showing the overlays. You can choose which type of cache TortoiseSVN should use according to your system and working copy size here:

Défaut
: Caches all status information in a separate process (`TSVNCache.exe`). That process watches all drives for changes and fetches the status again if files inside a working copy get modified. The process runs with the least possible priority so other programs don't get hogged because of it. That also means that the status information is not *real time* but it can take a few seconds for the overlays to change.

  Advantage: the overlays show the status recursively, i.e. if a file deep inside a working copy is modified, all folders up to the working copy root will also show the modified overlay. And since the process can send notifications to the shell, the overlays on the left tree view usually change too.

  Inconvénient : le processus fonctionne constamment, même si vous ne travaillez pas sur vos projets. Il utilise aussi environ 10-50 Mo de RAM selon le nombre et la taille de vos copies de travail.

Shell
: La mise en cache est faite directement à l'intérieur de la dll d'extension du shell, mais seulement pour le dossier actuellement visible. Chaque fois vous naviguez à un autre dossier, l'information de statut est parcourue de nouveau.

  Avantage : a seulement besoin de très peu de mémoire (autour de 1 MO de RAM) et peut montrer le statut *en temps réel*.

  Inconvénient : puisque un seul dossier est mis en cache, les recouvrements ne montrent pas le statut récursivement. Pour de grandes copies de travail, cela peut prendre plus de temps pour montrer un dossier dans l'explorateur qu'avec le cache par défaut. Aussi la colonne de type mime n'est pas disponible.

Aucun
: Avec ce réglage, TortoiseSVN ne va pas du tout chercher le statut dans l'Explorateur. De ce fait, les fichiers n'ont pas de recouvrement et les dossiers ont seulement un recouvrement 'normal' s'ils sont versionnés. Aucun autre recouvrement n'est affiché et aucune colonne supplémentaire n'est disponible non plus.

  Avantage : n'utilise absolument aucune mémoire supplémentaire et ne ralentit pas du tout l'Explorateur en parcourant.

  Disadvantage: Status information of files and folders is not shown in Explorer. To see if your working copies are modified, you have to use the « Check for modifications » dialog.

The next group allows you to select which classes of storage should show overlays. By default, only hard drives are selected. You can even disable all icon overlays, but where's the fun in that?

Les disques réseau peuvent être très lents, dont par défaut les icônes ne sont pas affichées pour les copies de travail situées dans des dossiers partagés.

Les disques USB Flash semblent être un cas particulier en cela que le type de disque est identifié par le périphérique lui-même. Certains apparaissent comme des disques fixes et d'autres comme des disques amovibles.

Les Chemins exclus sont utilisés pour indiquer à TortoiseSVN que ces chemins ne devraient *pas* montrer les recouvrements d'icône et les colonnes de statut. Cela est utile si vous avez de très grandes copies de travail contenant seulement des bibliothèques que vous ne changerez pas du tout et donc pour lesquelles vous n'aurez pas besoin des recouvrements. Par exemple :

`f:\dévelopement\SVN\Subversion` désactivera les recouvrements *seulement* sur ce dossier spécifique. Vous pouvez toujours voir les recouvrements sur tous les fichiers et tous les dossiers à l'intérieur de ce dossier.

`f:\development\SVN\Subversion*` désactivera les recouvrements sur *tous* les fichiers et dossiers dont le chemin commence par `f:\development\SVN\Subversion`. Cela signifie que vous ne verrez pas de recouvrement pour aucun des fichiers et des dossiers en-dessous de ce chemin.

La même chose s'applique aux Chemins inclus. Sauf que pour ces chemins, les recouvrements s'affichent même s'ils sont désactivés pour ce type de disque spécifique, ou par un chemin exclus indiqué au-dessus.

Users sometimes ask how these three settings interact, and the definitive answer is:

```
if (path is in include list)
  show overlays
if (path is allowed drive type) AND (path is not in exclude list)
  show overlays
```

The include list *always* makes the overlays show. Otherwise, overlays are shown for all marked drive types *unless* the path is excluded.

TSVNCache.exe utilise aussi ces chemins pour limiter son balayage. Si vous voulez qu'il ne regarde que dans des dossiers particuliers, désactivez tous les types de disque et incluez seulement les dossiers que vous voulez spécifiquement être parcourus.

Exclure les Lecteurs SUBST

It is often convenient to use a SUBST drive to access your working copies, e.g. using the command

```
subst T: C:\TortoiseSVN\trunk\doc
```

However this can cause the overlays not to update, as TSVNCache will only receive one notification when a file changes, and that is normally for the original path. This means that your overlays on the subst path may never be updated.

An easy way to work around this is to exclude the original path from showing overlays, so that the overlays show up on the subst path instead.

Sometimes you will exclude areas that contain working copies, which saves TSVNCache from scanning and monitoring for changes, but you still want a visual indication that such folders are versioned. The Show excluded folders as 'normal' checkbox allows you to do this. With this option, versioned folders in any excluded area (drive type not checked, or specifically excluded) will show up as normal and up-to-date, with a green check mark. This reminds you that you are looking at a working copy, even though the folder overlays may not be correct. Files do not get an overlay at all. Note that the context menus still work, even though the overlays are not shown.

As a special exception to this, drives A: and B: are never considered for the Show excluded folders as 'normal' option. This is because Windows is forced to look on the drive, which can result in a delay of several seconds when starting Explorer, even if your PC does have a floppy drive.

4.30.3.1. Sélection du jeu d'icônes

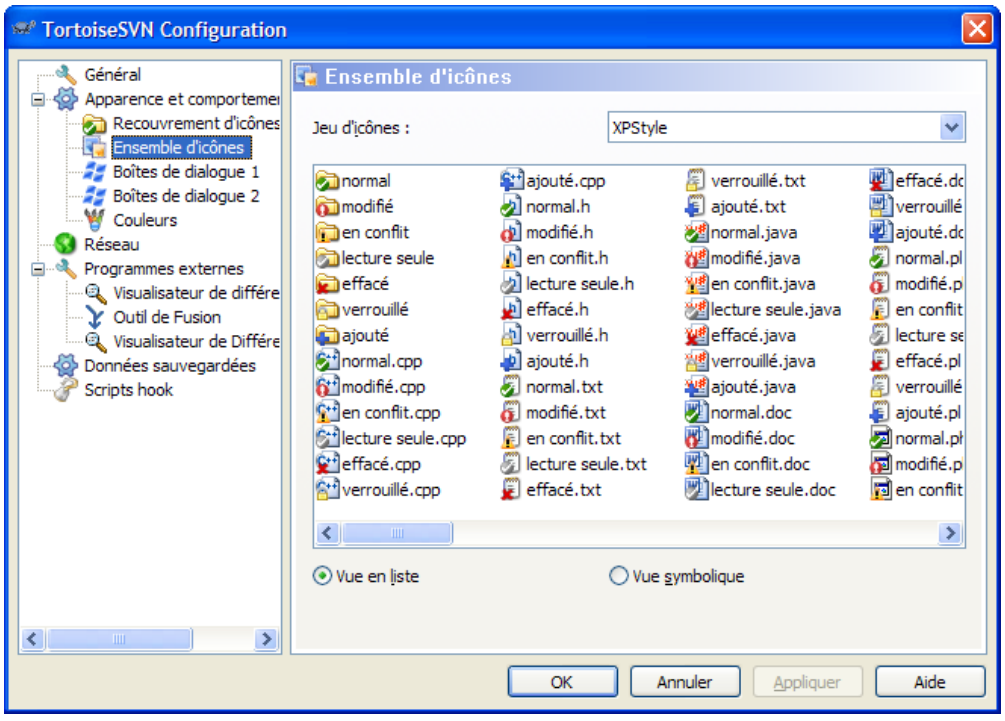


Figure 4.58. La boîte de dialogue Configuration, page Ensemble d'icônes

Vous pouvez changer le jeu d'icônes de recouvrement pour celui que vous aimez le plus. Notez que si vous changez le jeu de recouvrement, vous devriez redémarrer votre ordinateur pour que les changements prennent effet.

4.30.4. Configuration du réseau

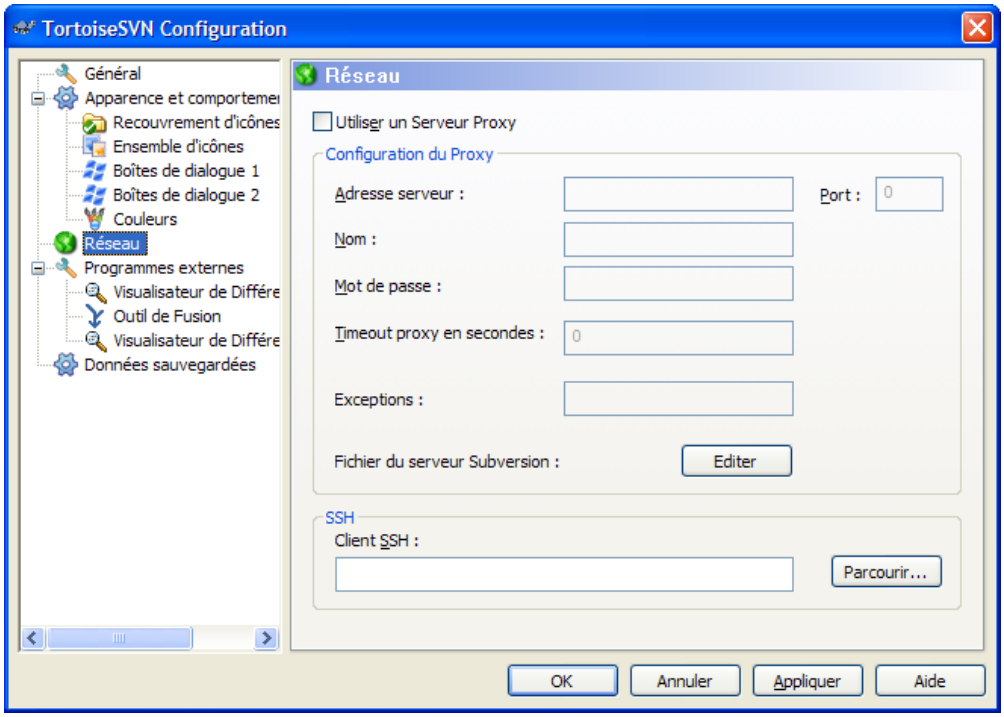


Figure 4.59. La boîte de dialogue Configuration, page Réseau

Vous pouvez ici configurer votre serveur proxy, si vous en avez besoin pour passer le pare-feu de votre société.

If you need to set up per-repository proxy settings, you will need to use the Subversion `servers` file to configure this. Use Edit to get there directly. Consult the *Runtime Configuration Area* [http://svnbook.red-bean.com/en/1.5/svn.advanced.confarea.html] for details on how to use this file.

Vous pouvez aussi spécifier quel programme TortoiseSVN devrait utiliser pour établir une connexion sécurisée à un référentiel svn+ssh. Nous vous recommandons d'utiliser TortoisePlink.exe. C'est une version du programme populaire Plink et elle est incluse avec TortoiseSVN, mais elle est compilée comme une application sans fenêtre, donc vous n'obtenez pas de boîte DOS surgissant chaque fois vous vous authentifiez.

You must specify the full path to the executable. For TortoisePlink.exe this is the standard TortoiseSVN bin directory. Use the Browse button to help locate it. Note that if the path contains spaces, you must enclose it in quotes, e.g.

```
"C:\Program Files\TortoiseSVN\bin\TortoisePlink.exe"
```

Un effet secondaire de ne pas avoir de fenêtre est qu'il n'y a nulle part où afficher les messages d'erreur, ainsi si l'authentification échoue, vous obtiendrez simplement un message disant quelque chose comme « Impossible d'écrire sur la sortie standard » Pour cette raison, nous vous recommandons de mettre d'abord en place en utilisant Plink standard. Quand tout fonctionne, vous pouvez utiliser TortoisePlink avec exactement les mêmes paramètres.

TortoisePlink does not have any documentation of its own because it is just a minor variant of Plink. Find out about command line parameters from the *PuTTY website* [http://www.chiark.greenend.org.uk/~sgtatham/putty/]

Pour que le mot de passe ne vous soit pas demandé à chaque fois, vous devriez utiliser un outil de mise en cache des mots de passe comme Pageant. Cet utilitaire est disponible sur le site internet de PuTTY.

Finally, setting up SSH on server and clients is a non-trivial process which is beyond the scope of this help file. However, you can find a guide in the TortoiseSVN FAQ listed under *Subversion/TortoiseSVN SSH How-To* [http://tortoisesvn.net/ssh_howto].

4.30.5. Réglages des programmes externes

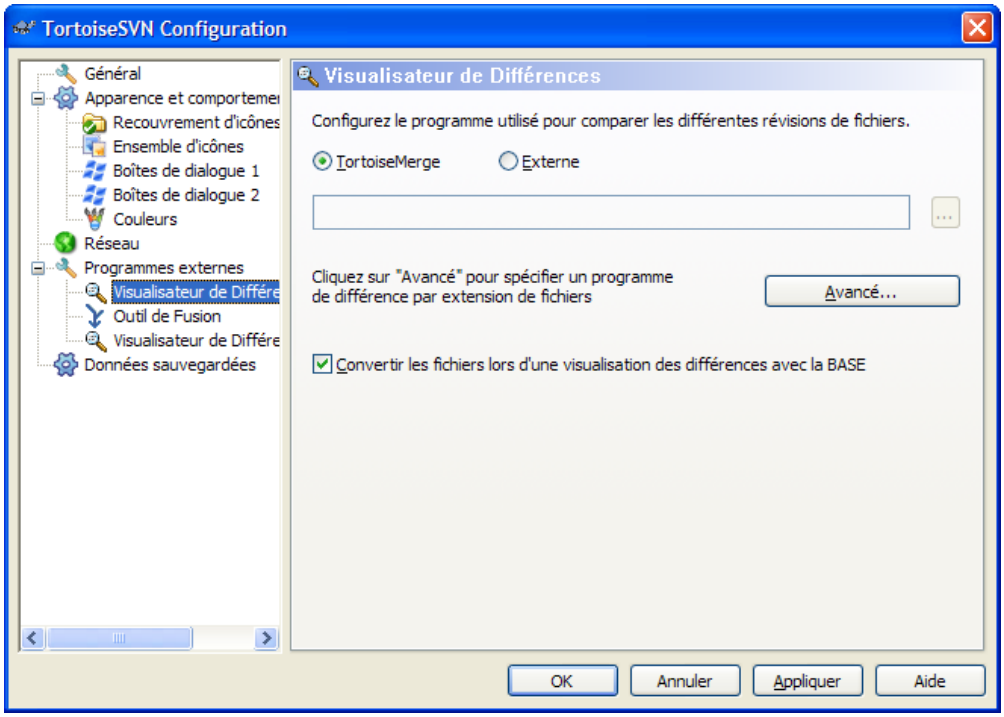


Figure 4.60. La boîte de dialogue Configuration, page Visualisateur de différence

Vous pouvez ici définir vos propres programmes de comparaison/fusion que TortoiseSVN devrait utiliser. Le réglage par défaut utilise TortoiseMerge qui est installé avec TortoiseSVN.

Lisez Section 4.10.5, « Outils de différenciation/fusion externes » pour une liste de quelques programmes externes de différenciation/fusion que les gens utilisent avec TortoiseSVN.

4.30.5.1. Visualisateur de différences

Un programme de comparaison externe peut être utilisé pour comparer des révisions différentes de fichiers. Le programme externe devra obtenir les noms de fichier depuis la ligne de commande, avec les autres options de ligne de commande. TortoiseSVN utilise des paramètres de substitution préfixés par %. Quand il rencontre l'un d'eux, il substituera la valeur appropriée. L'ordre des paramètres dépendra du programme de comparaison que vous utilisez.

%base
: Le fichier original sans vos changements

%bname
: Le titre de la fenêtre pour le fichier de base

%mine
: Votre propre fichier, avec vos changements

%yname
Le titre de la fenêtre pour votre fichier

Les titres de fenêtre ne sont pas des noms de fichier purs. TortoiseSVN traite cela comme un nom à afficher et crée les noms en conséquence. Si par exemple si vous faites une comparaison entre un fichier dans la révision 123 avec un fichier de votre copie de travail, les noms seront `nom de fichier : révision 123` et `nom de fichier : copie de travail`

For example, with ExamDiff Pro:

```
C:\Path-To\ExamDiff.exe %base %mine --left_display_name:%bname
    --right_display_name:%yname
```

or with KDiff3:

```
C:\Path-To\kdiff3.exe %base %mine --L1 %bname --L2 %yname
```

or with WinMerge:

```
C:\Path-To\WinMerge.exe -e -ub -dl %bname -dr %yname %base %mine
```

or with Araxis:

```
C:\Path-To\compare.exe /max /wait /title1:%bname /title2:%yname
    %base %mine
```

If you use the `svn:keywords` property to expand keywords, and in particular the *revision* of a file, then there may be a difference between files which is purely due to the current value of the keyword. Also if you use `svn:eol-style = native` the BASE file will have pure `LF` line endings whereas your file will have `CR-LF` line endings. TortoiseSVN will normally hide these differences automatically by first parsing the BASE file to expand keywords and line endings before doing the diff operation. However, this can take a long time with large files. If Convert files when diffing against BASE is unchecked then TortoiseSVN will skip pre-processing the files.

Vous pouvez aussi indiquer un autre outil de diff pour les propriétés Subversion. Dans la mesure où ce sont de courtes chaines de caractères, il est légitime de vouloir une visionneuse plus compacte.

If you have configured an alternate diff tool, you can access TortoiseMerge *and* the third party tool from the context menus. Context menu → Diff uses the primary diff tool, and **Shift**+ Context menu → Diff uses the secondary diff tool.

4.30.5.2. Outil de fusion

Un programme de fusion externe utilisé pour résoudre les fichiers en conflit. La substitution de paramètre est utilisée de la même manière qu'avec le programme de comparaison.

%base
le fichier original sans vos changements ou ceux des autres

%bname
Le titre de la fenêtre pour le fichier de base

%mine
votre propre fichier, avec vos changements

%yname
Le titre de la fenêtre pour votre fichier

%theirs
le fichier tel qu'il est dans le référentiel

%tname
Le titre de la fenêtre pour le fichier dans le référentiel

%merged
le fichier en conflit, le résultat de l'opération de fusion

%mname
Le titre de la fenêtre pour le fichier fusionné

Par exemple, avec Perforce Merge:

```
C:\Chemin-De\P4Merge.exe %base %theirs %mine %merged
```

ou avec KDiff3:

```
C:\Chemin-De\kdiff3.exe %base %mine %theirs -o %merged
    --L1 %bname --L2 %yname --L3 %tname
```

ou avec Araxis:

```
C:\Chemin-De\compare.exe /max /wait /3 /title1:%tname /title2:%bname
    /title3:%yname %theirs %base %mine %merged /a2
```

ou avec WinMerge (2.8 or later):

```
C:\Chemin-De\WinMerge.exe %merged
```

4.30.5.3. Réglages avancés de comparaison/fusion

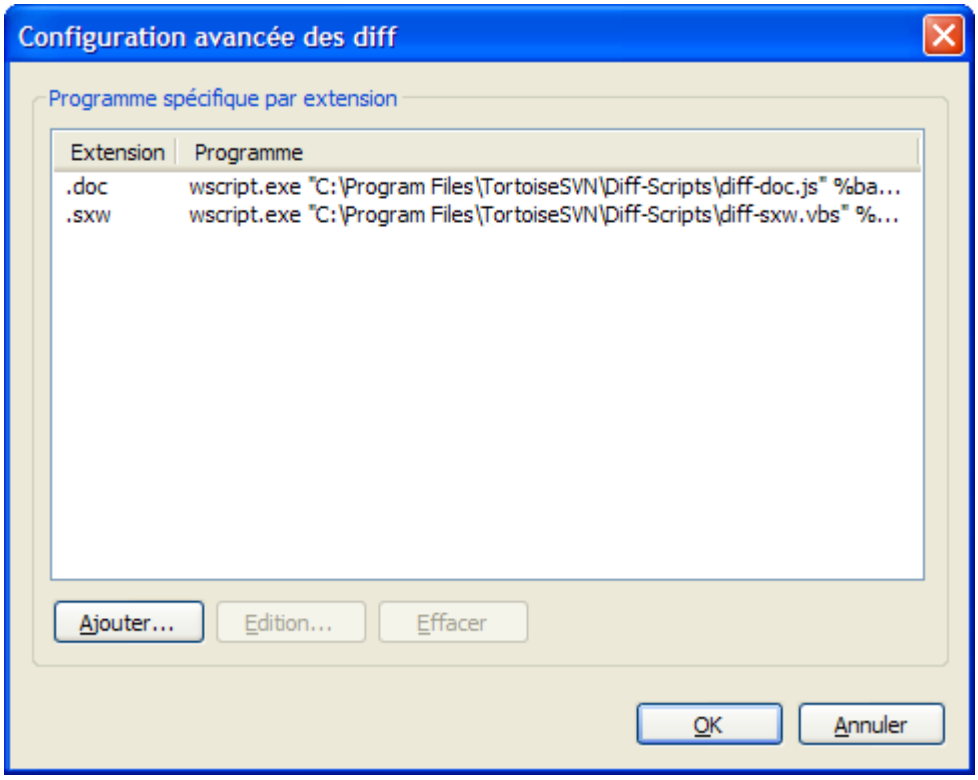


Figure 4.61. La boîte de dialogue Configuration, Boîte de dialogue Comparaison/fusion avancée

In the advanced settings, you can define a different diff and merge program for every file extension. For instance you could associate Photoshop as the « Diff » Program for `.jpg` files :-) You can also associate the `svn:mime-type` property with a diff or merge program.

Pour associer en utilisant une extension de fichier, vous devez spécifier l'extension. Utilisez `.bmp` pour décrire les fichiers bitmap de Windows. Pour associer en utilisant la propriété `svn:mime-type`, spécifier le type mime, en incluant un slash, par exemple `text/xml`.

4.30.5.4. Visualisateur de différences unifiées

Un programme de visualisation pour les fichiers de différences unifiées (fichiers patch). Aucun paramètre n'est exigé. L'option Défaut consiste à vérifier l'association des fichier pour les fichiers `.diff`, et ensuite pour les fichiers `.txt`. Si vous n'avez pas de visualisateur pour les fichiers `.diff`, vous arriverez très probablement sur le Bloc-notes.

Le programme Bloc-notes original de Windows ne se comporte pas bien sur les fichiers qui n'ont pas de fins de ligne standards CR-LF . Puisque la plupart des fichiers de différences unifiées ont des fins de ligne LF pures, ils n'apparaissent pas bien dans le Bloc-notes. Cependant, vous pouvez télécharger un remplaçant du Bloc-notes gratuit *Notepad2* [http://www.flos-freeware.ch/notepad2.html] qui montre non seulement les fins de ligne correctement, mais aussi met en couleurs les lignes ajoutées et supprimées.

4.30.6. Configuration des données sauvegardées

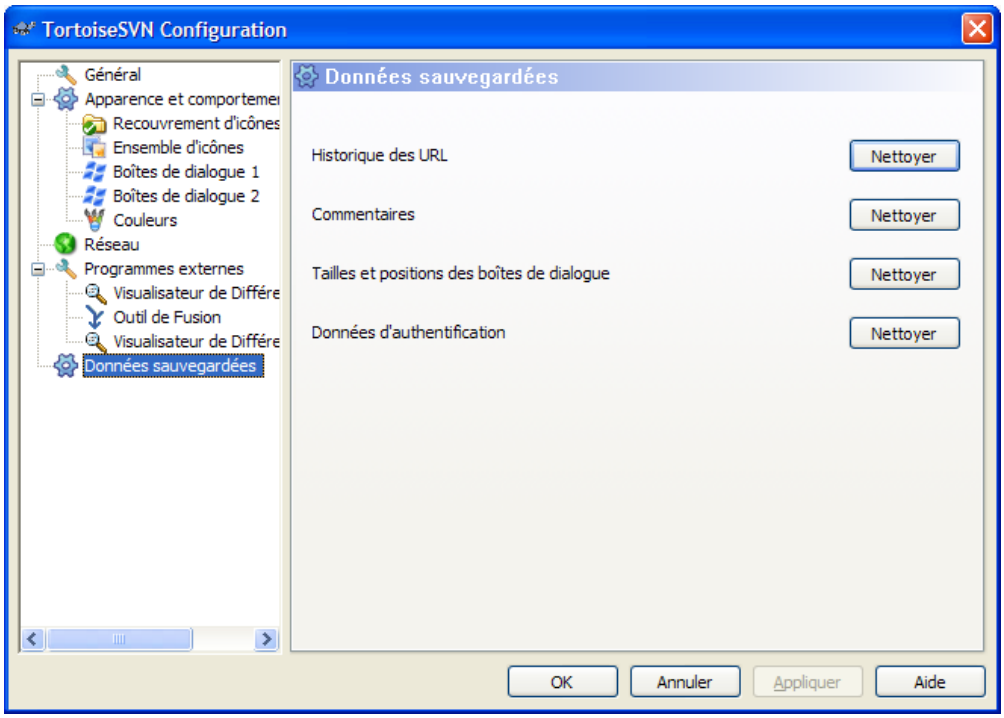


Figure 4.62. La boîte de dialogue Configuration, Page Données sauvegardées

Pour votre convenance, TortoiseSVN enregistre les réglages que vous utilisez et se souvient où vous avez été récemment. Si vous voulez nettoyer ce cache de données, vous pouvez le faire ici.

Historique des URL
: Chaque fois que vous extrayez une copie de travail, fusionnez des changements ou utilisez l'explorateur de référentiel, TortoiseSVN tient un rapport des URLS récemment utilisées et les propose dans une boîte déroulante. Cette liste est parfois encombrée par des URLs périmées donc il est utile de la nettoyer périodiquement.

  If you want to remove a single item from one of the combo boxes you can do that in-place. Just click on the arrow to drop the combo box down, move the mouse over the item you want to remove and type **Shift+Del**.

Messages de log (fenêtre d'édition)
: TortoiseSVN stocke les commentaires récents de livraison que vous saisissez. Ceux-ci sont stockés par référentiel, donc si vous avez accès à beaucoup de référentiels, cette liste peut devenir assez longue.

Messages de log (Montrer la fenêtre de log)
: TortoiseSVN caches log messages fetched by the Show Log dialog to save time when you next show the log. If someone else edits a log message and you already have that message cached, you will not see the change until you clear the cache. Log message caching is enabled on the Log Cache tab.

Tailles et positions des boîtes de dialogue
: Plusieurs boîtes de dialogue se souviennent de la taille et de la position de l'écran utilisées en dernier.

Données d'authentification
: Quand vous vous authentifiez avec un serveur Subversion, le nom de l'utilisateur et le mot de passe sont mis en cache localement pour que vous n'ayez pas à les entrer à nouveau. Vous pouvez vouloir effacer cela pour des raisons de sécurité ou parce que voulez accédez au référentiel sous un autre nom d'utilisateur ... est-ce que John sait que vous utilisez son PC ?

  If you want to clear authentication data for one particular server only, read Section 4.1.5, « Authentification » for instructions on how to find the cached data.

Log des Actions
: TortoiseSVN keeps a log of everything written to its progress dialogs. This can be useful when, for example, you want to check what happened in a recent update command.

  The log file is limited in length and when it grows too big the oldest content is discarded. By default 4000 lines are kept, but you can customize that number.

  Depuis cet endroit vous pouvez vois le contenu du fichier de commentaires, ainsi que le vider.

4.30.7. Mise en Cache des messages de log

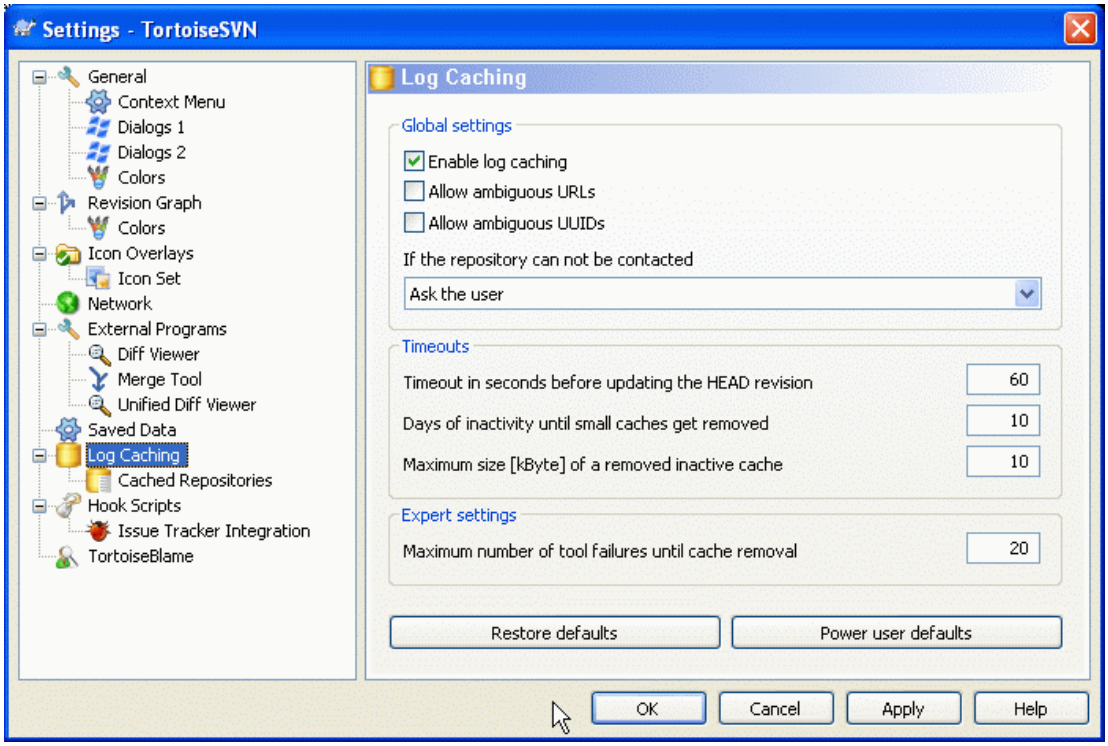


Figure 4.63. La boîte de dialogue de Configuration, Page de Mise en Cache des Logs

This dialog allows you to configure the log caching feature of TortoiseSVN, which retains a local copy of log messages and changed paths to avoid time-consuming downloads from the server. Using the log cache can dramatically speed up the log dialog and the revision graph. Another useful feature is that the log messages can still be accessed when offline.

Activer la mise en cache des messages de log
: Enables log caching whenever log data is requested. If checked, data will be retrieved from the cache when available, and any messages not in the cache will be retrieved from the server and added to the cache.

  If caching is disabled, data will always be retrieved directly from the server and not stored locally.

Permettre les URLs ambigues
: Occasionally you may have to connect to a server which uses the same URL for all repositories. Older versions of `svnbridge` would do this. If you need to access such repositories you will have to check this option. If you don't, leave it unchecked to improve performance.

Permettre les UUIDs ambigus
: Some hosting services give all their repositories the same UUID. You may even have done this yourself by copying a repository folder to create a new one. For all sorts of reasons this is a bad idea - a UUID should be *unique*. However, the log cache will still work in this situation if you check this box. If you don't need it, leave it unchecked to improve performance.

Si le référentiel n'est pas disponible
: If you are working offline, or if the repository server is down, the log cache can still be used to supply log messages already held in the cache. Of course the cache may not be up-to-date, so there are options to allow you to select whether this feature should be used.

  When log data is being taken from the cache without contacting the server, the dialog using those message will show the offline state in its title bar.

Temps d'attente maximum écoulé avant la mise à jour de la révision de tête (HEAD).
: When you invoke the log dialog you will normally want to contact the server to check for any newer log messages. If the timeout set here is non-zero then the server will only be contacted when the timeout has elapsed since the last time contact. This can reduce server round-trips if you open the log dialog frequently and the server is slow, but the data shown may not be completely up-to-date. If you want to use this feature we suggest using a value of 300 (5 minutes) as a compromise.

Nombre de jours d'inactivité avant que les petits caches soient supprimés
: If you browse around a lot of repositories you will accumulate a lot of log caches. If you're not actively using them, the cache will not grow very big, so TortoiseSVN purges them after a set time by default. Use this item to control cache purging.

Taille maximum des caches inactifs à supprimer
: Larger caches are more expensive to reacquire, so TortoiseSVN only purges small caches. Fine tune the threshold with this value.

Nombre maximum d'échecs de l'outil avant suppression du cache
: Occasionally something goes wrong with the caching and causes a crash. If this happens the cache is normally deleted automatically to prevent a recurrence of the problem. If you use the less stable nightly build you may opt to keep the cache anyway.

4.30.7.1. Référentiels mis en mémoire cache

On this page you can see a list of the repositories that are cached locally, and the space used for the cache. If you select one of the repositories you can then use the buttons underneath.

Cliquez sur le bouton Mise à jour pour rafraichir complètement la mémoire cache et remplir tous les trous. Pour les gros référentiels cette tâche peut être très coûteuse, mais utile si vous souhaitez vous déconnecter et avoir le meilleur cache possible.

Click on the Export button to export the entire cache as a set of CSV files. This could be useful if you want to process the log data using an external program, although it is mainly useful to the developers.

Click on Delete to remove all cached data for the selected repositories. This does not disable caching for the repository so the next time you request log data, a new cache will be created.

4.30.7.2. Statistiques d'Utilisation du Cache

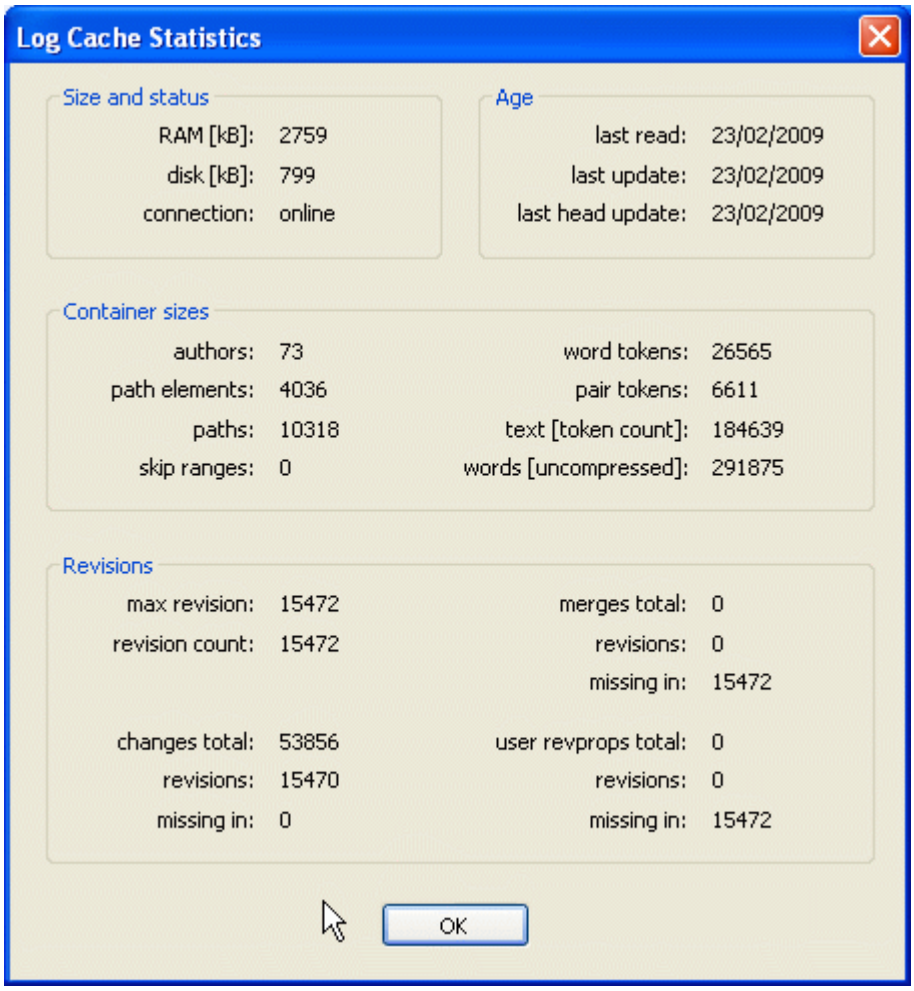


Figure 4.64. La Fenêtre de propriétés, Statistiques d'Utilisation de la Mémoire Cache

Cliquer sur le bouton Détails pour voir les statistiques détaillées d'une mémoire cache particulière. Beaucoup des champs montrés là ont surtout un intérêt pour les développeur de TortoiseSVN, ils ne sont donc pas tous expliqués en détail.

RAM
: La quantité de mémoire servant à ce cache

Disque
: The amount of disk space used for the cache. Data is compressed, so disk usage is generally fairly modest.

Connexion
: Montre si le référentiel était disponible la dernière fois que le cache a été utilisé.

Dernière Mise à Jour
: La dernière fois que le contenu de la mémoire cache a changé.

Dernière mise à jour de la version de tête
: La dernière fois que la révision HEAD a été demandée au serveur.

Auteurs
: Le nombre d'auteurs différents ayant enregistré des messages dans le cache.

Chemins
: Le nombre de chemins listés, comme vous pourriez le voir avec `svn log -v`.

Ignorer des plages de révisions
: The number of revision ranges which we have not fetched, simply because they haven't been requested. This is a measure of the number of holes in the cache.

Révision La plus Elevée
: Le numéro de version le plus élevé étant enregistré dans le cache.

Compteur de Révision
: Le nombre de révisions stockées en mémoire cache. C'est un autre système de mesure de la mémoire cache.

4.30.8. Scripts hook côté client

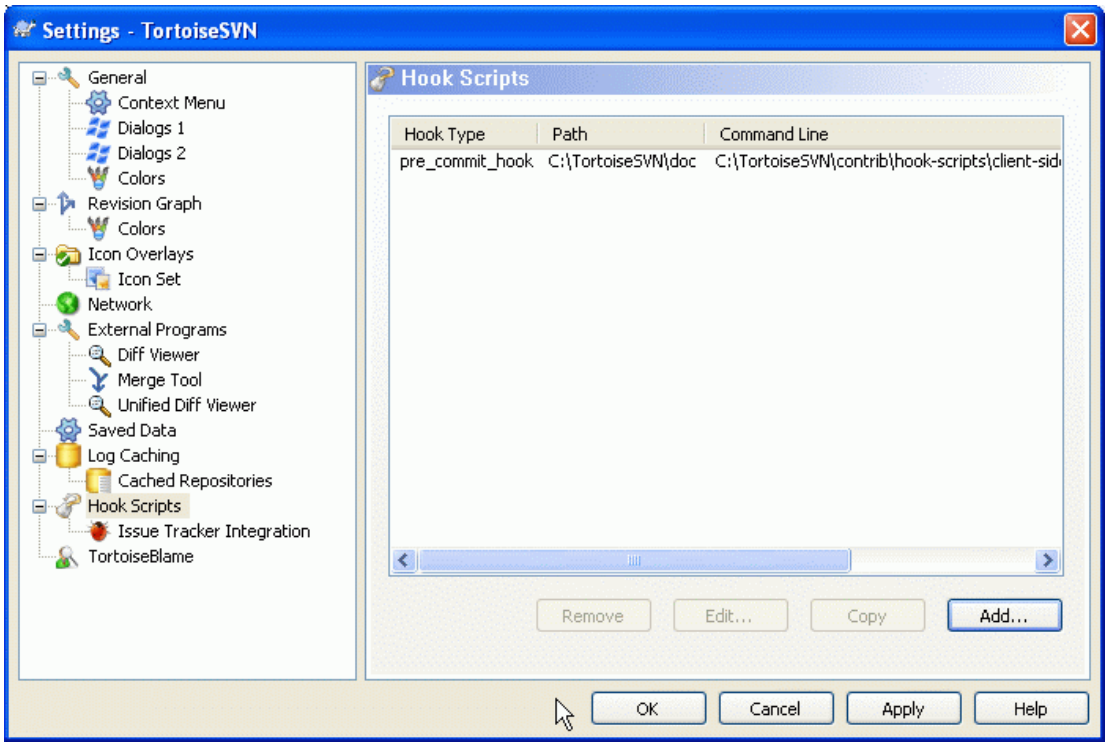


Figure 4.65. La boîte de dialogue Configuration, page Scripts hook

This dialog allows you to set up hook scripts which will be executed automatically when certain Subversion actions are performed. As opposed to the hook scripts explained in Section 3.3, « Scripts de hook côté serveur », these scripts are executed locally on the client.

One application for such hooks might be to call a program like `SubWCRev.exe` to update version numbers after a commit, and perhaps to trigger a rebuild.

Pour des raisons de sécurité et d'implémentation, les scripts de hook sont définis localement sur une machine plutôt que dans les propriétés du projet. Vous définissez ce qui se passe, en ne tenant pas compte

de ce que quelqu'un d'autre pourrait livrer dans le référentiel. Bien sûr vous pouvez toujours choisir d'appeler un script qui est lui même sous contrôle de version.

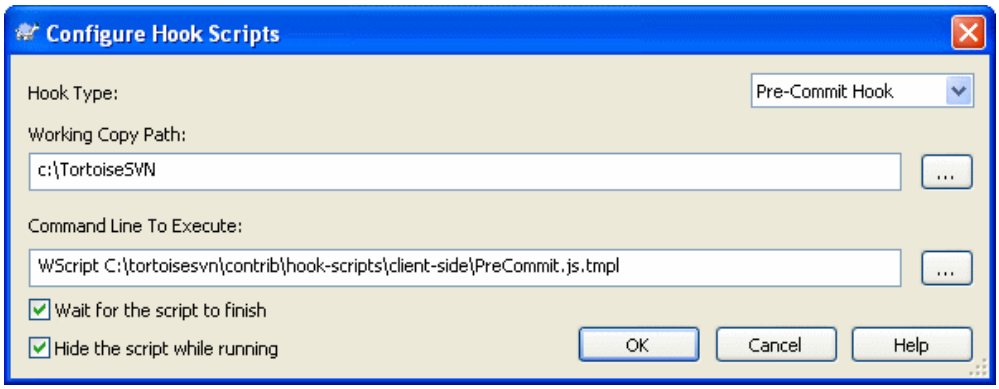


Figure 4.66. La fenêtre de paramétrage, configuration des scripts de hook

Pour ajouter un nouveau script hook, cliquez simplement sur Ajouter et saisissez les détails.

Il existe actuellement six types de script hook disponibles

Start-commit
: Appelé avant que la fenêtre de livraison ne s'affiche. Vous pouvez avoir besoin d'appeler ce hook pour ajouter à un fichier de version la liste des fichiers ayant été livrés et/ou le message de livraison. Néanmoins, il est important de noter qu'étant donné que le hook est appelé tôt, la liste complète des objects à livrer n'est pas disponible.

Pre-commit
: Appelé lorsque l'utilisateur clique sur le bouton OK dans la fenêtre de livraison, et avant que la livraison ne commence. Ce hook contient une liste de tout ce qui sera livré.

Post-commit
: Appellé après la fin de la livraison (qu'elle soit réussie ou non).

Start-update
: Appelé avant que la fenêtre mise à jour-à-la -révision ne soit affichée.

Pre-update
: Appelé avant que la mise à jour Subversion ne commence.

Post-update
: Appelé après la mise à jour (quelle soit réussie ou non)

Un hook est défini pour un répertoire précis d'une copie de travail. Vous n'avez besoin que de spécifier le répertoire de plus haut niveau ; si vous faites une opération sur un sous répertoire, TortoiseSVN recherchera automatiquement dans les dossiers parents un répertoire correspondant.

Ensuite vous devez spécifier la ligne de commande à exécuter, en commençant par le chemin du script de hook ou de l'exécutable. Ce peut être un script de batch, un exécutable ou quelqu'autre fichier que le système sache exécuter, par exemple un script perl.

The command line includes several parameters which get filled in by TortoiseSVN. The parameters passed depend upon which hook is called. Each hook has its own parameters which are passed in the following order:

Start-commit
: `PATHMESSAGEFILECWD`

Pre-commit
: `PATHDEPTHMESSAGEFILECWD`

Post-commit
: `PATHDEPTHMESSAGEFILEREVISIONERRORCWD`

Start-update
: `PATHCWD`

Pre-update
: `PATHDEPTHREVISIONCWD`

Post-update
: `PATHDEPTHREVISIONERRORCWD`

La signification de chacun des paramètres est décrite ici :

PATH
: Un chemin d'un fichier temporaire contenant tous les chemins d'où les opérations ont commencé. Il y a un chemin par ligne dans le fichier temporaire.

DEPTH
: Profondeur dans laquelle la livraison/mise à jour est faite.

  Les valeurs possibles sont :

  -2
  : `svn_depth_unknown`

  -1
  : `svn_depth_exclude`

  0
  : `svn_depth_empty`

  1
  : `svn_depth_files`

  2
  : `svn_depth_immediates`

  3
  : `svn_depth_infinity`

MESSAGEFILE
: Le chemin d'un fichier contenant les commentaires de livraison. Le fichier est encodé en UTF-8. Après l'exécution réussie d'un script de hook de start-commit, le commentaire est relu, permettant au script de hook de le modifier.

REVISION
: The repository revision to which the update should be done or after a commit completes.

ERROR
: Path to a file containing the error message. If there was no error, the file will be empty.

CWD
: The current working directory with which the script is run. This is set to the common root directory of all affected paths.

Note that although we have given these parameters names for convenience, you do not have to refer to those names in the hook settings. All parameters listed for a particular hook are always passed, whether you want them or not ;-)

If you want the Subversion operation to hold off until the hook has completed, check Wait for the script to finish.

Normally you will want to hide ugly DOS boxes when the script runs, so Hide the script while running is checked by default.

Sample client hook scripts can be found in the `contrib` folder in the *TortoiseSVN repository* [http://tortoisesvn.tigris.org/svn/tortoisesvn/trunk/contrib/hook-scripts]. (Section 3, « TortoiseSVN est gratuit ! » explains how to access the repository).

4.30.8.1. Intégration d'un Bug Tracker

TortoiseSVN can use a COM plugin to query issue trackers when in the commit dialog. The use of such plugins is described in Section 4.28.2, « Récupérer des Informations depuis un Traqueur de Bug ». If your system administrator has provided you with a plugin, which you have already installed and registered, this is the place to specify how it integrates with your working copy.

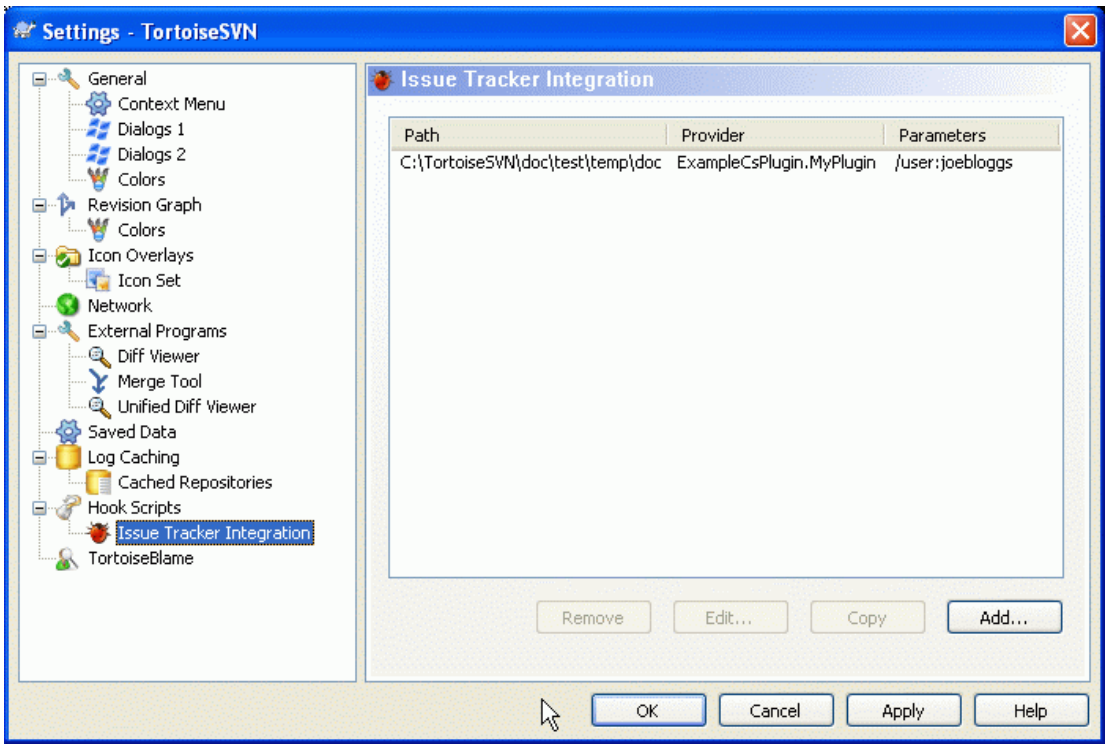


Figure 4.67. La Fenêtre de Propriétés, Page d'Intégration d'un Bug Tracker

Click on Add... to use the plugin with a particular working copy. Here you can specify the working copy path, choose which plugin to use from a drop down list of all registered issue tracker plugins, and any parameters to pass. The parameters will be specific to the plugin, but might include your user name on the issue tracker so that the plugin can query for issues which are assigned to you.

If you want all users to use the same COM plugin for your project, you can specify the plugin also with the properties `bugtraq:provideruuid` and `bugtraq:providerparams`.

bugtraq:provideruuid
: This property specifies the COM UUID of the IBugtraqProvider, for example `{91974081-2DC7-4FB1-B3BE-0DE1C8D6CE4E}`. (this example is the UUID of the *Gurtle bugtraq provider* [http://code.google.com/p/gurtle/], which is a provider for the *Google Code* [http://code.google.com/hosting/] issue tracker).

bugtraq:providerparams
: Cette propriété spécifie les paramètres envoyés à IBugTraqProvider.

Please check the documentation of your IBugtraqProvider plugin to find out what to specify in these two properties.

4.30.9. Configuration de TortoiseBlame

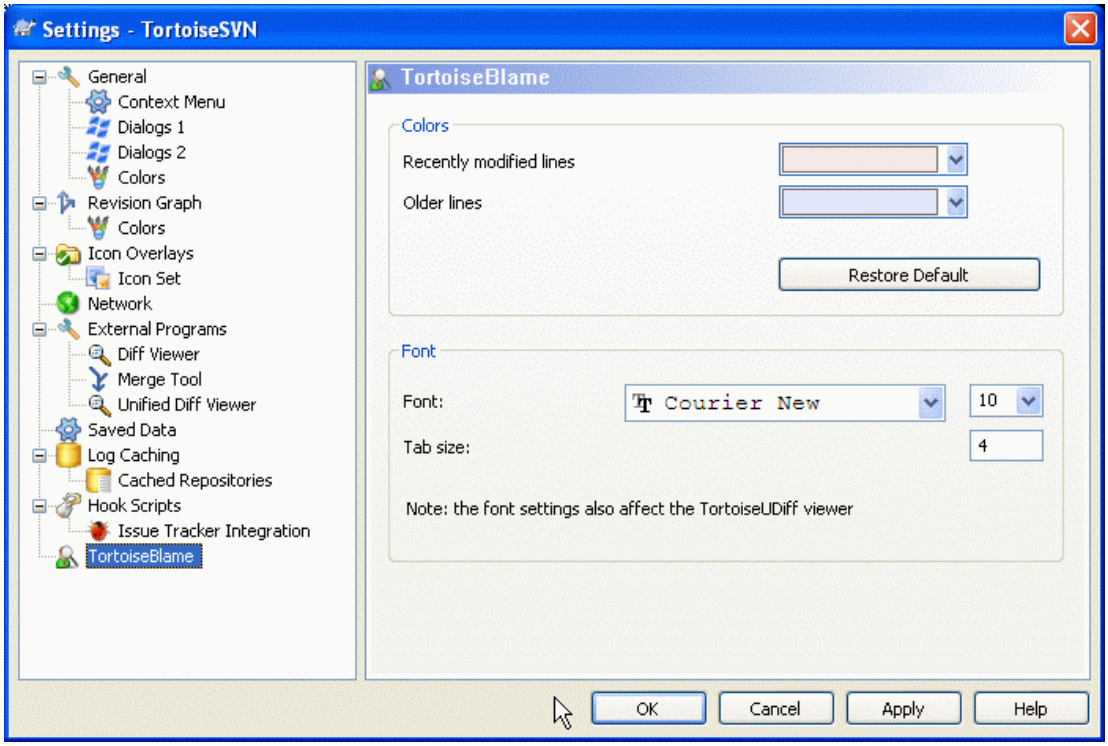


Figure 4.68. La boîte de dialogue ce configuration, page de bannissement.

The settings used by TortoiseBlame are controlled from the main context menu, not directly with TortoiseBlame itself.

Couleurs
: TortoiseBlame can use the background colour to indicate the age of lines in a file. You set the endpoints by specifying the colours for the newest and oldest revisions, and TortoiseBlame uses a linear interpolation between these colours according to the repository revision indicated for each line.

Police
: You can select the font used to display the text, and the point size to use. This applies both to the file content, and to the author and revision information shown in the left pane.

Tabulations
: Définit combien d'espaces utiliser à la place d'une tabulation dans le fichier.

4.30.10. Réglages dans le registre

Quelques-uns des réglages rarement utilisées ne sont disponibles qu'en éditant directement le registre. Il n'est nul besoin de préciser que vous ne devriez éditer le registre que si vous savez ce que vous faites.

Configuration
: Vous pouvez spécifier un emplacement différent pour le fichier de configuration Subversion en utilisant l'emplacement du registre `HKCU\Software\TortoiseSVN\ConfigDir`. Ceci affectera toutes les opérations de TortoiseSVN.

Mettre l'icône de la barre des tâche en cache
: Pour ajouter une icône de notification de cache pour le programme TSVNCache, créez une clé `DWORD` avec une valeur de 1 à `HKCU\Software\TortoiseSVN\CacheTrayIcon`. C'est vraiment seulement utile pour les développeurs puisqu'il vous permet de fermer le programme proprement.

Debug
: To show the command line parameters passed from the shell extension to TortoiseProc.exe create a `DWORD` key with a value of 1 at `HKCU\Software\TortoiseSVN\Debug`.

Icones du Menu contextuel
: This can be useful if you use something other than the windows explorer or if you get problems with the context menu displaying correctly. create a `DWORD` key with a value of 0 at `HKCU\Software\TortoiseSVN\ShowContextMenuIcons` if you don't want TortoiseSVN to not show icons for the shell context menu items. Set this value to 1 to show the icons again.

Bloquer l'état du recouvrement d'icônes
: If you don't want the explorer to update the status overlays while another TortoiseSVN command is running (e.g. Update, Commit, ...) then create a `DWORD` key with a value of 1 at `HKCU\Software\TortoiseSVN\BlockStatus`.

Update Check URL
: `HKCU\Software\TortoiseSVN\UpdateCheckURL` contains the URL from which TortoiseSVN tries to download a text file to find out if there are updates available. You can also set this under `HKLM` instead of `HKCU` if you want, but `HKCU` overwrites the setting in `HKLM`. This might be useful for company admins who don't want their users to update TortoiseSVN until they approve it.

Nom des fichiers sans extension dans la liste de complétion automatique
: The auto-completion list shown in the commit message editor displays the names of files listed for commit. To also include these names with extensions removed, create a `DWORD` key with a value of 1 at `HKCU\Software\TortoiseSVN\AutocompleteRemovesExtensions`.

Colonnes de l'explorateur partout
: The extra columns the TortoiseSVN adds to the details view in Windows Explorer are normally only active in a working copy. If you want those to be accessible everywhere, not just in working copies, create a `DWORD` key with a value of 1 at `HKCU\Software\TortoiseSVN\ColumnsEveryWhere`.

Séparateur des commentaires de fusion
: When you merge revisions from another branch, and merge tracking information is available, the log messages from the revisions you merge will be collected to make up a commit log message. A pre-defined string is used to separate the individual log messages of the merged revisions. If you prefer, you can create a `SZ` key at `HKCU\Software\TortoiseSVN\MergeLogSeparator` containing a separator string of your choice.

Toujours bannir les modifications faites avec TortoiseMerge
: TortoiseSVN allows you to assign external diff viewer. Most such viewers, however, are not suited for change blaming (Section 4.23.2, « Annoter les différences »), so you might wish to fall back to TortoiseMerge in this case. To do so, create a `DWORD` key with a value of 1 at `HKCU\Software\TortoiseSVN\DiffBlamesWithTortoiseMerge`.

Mise en valeur de la version courante des répertoires dans la fenêtre de log.
: The log dialog highlights the current working copy revision when the log is shown for a file. To do the same thing for a folder requires a working copy crawl, which is the default action, but it can be a slow operation for large working copies. If you want to change the operation of this feature you must create a `DWORD` registry key at `HKCU\Software\TortoiseSVN\RecursiveLogRev`. A value of 0 disables the feature (no highlighting for folders), a value of 1 (default) will fetch the status recursively (find the highest revision in the working copy tree), and a value of 2 will check the revision of the selected folder itself, but will not check any child items.

Faire échouer la récupération si un élément du même nom existe déjà
: By default, if you checkout a working copy over an existing unversioned folder structure, as you might do after import, then any existing which differ from the repository content will be left unchanged and marked as modified. When you come to commit, it is your local copy which will then be sent back to the repository. Some people would prefer the checkout to fail if the

existing content differs, so that if two people add the same file the second person's version does not overwrite the original version by mistake. If you want to force checkouts to fail in this instance you must create a `DWORD` registry key with value 0 at `HKCU\Software\TortoiseSVN\AllowUnversionedObstruction`.

4.30.11. Dossiers de travail de Subversion

VS.NET 2003 when used with web projects can't handle the `.svn` folders that Subversion uses to store its internal information. This is not a bug in Subversion. The bug is in VS.NET 2003 and the frontpage extensions it uses.

> Notez que le bug a été corrigé à partir de VS2005.

As of Version 1.3.0 of Subversion and TortoiseSVN, you can set the environment variable `SVN_ASP_DOT_NET_HACK`. If that variable is set, then Subversion will use `_svn` folders instead of `.svn` folders. You must restart your shell for that environment variable to take effect. Normally that means rebooting your PC. To make this easier, you can now do this from the general settings page using a simple checkbox - refer to Section 4.30.1, « Configuration générale ».

For more information, and other ways to avoid this problem in the first place, check out the article about this in our *FAQ* [http://tortoisesvn.net/aspdotnethack].

4.31. Étape Finale

Faites une donation!

Bien que TortoiseSVN et TortoiseMerge soient gratuits, vous pouvez supporter les développeurs en envoyant des patches et en jouant un rôle actif dans le développement. Vous pouvez aussi aider à nous réconforter pendant les heures interminables que nous passons devant nos ordinateurs.

While working on TortoiseSVN we love to listen to music. And since we spend many hours on the project we need a *lot* of music. Therefore we have set up some wish-lists with our favourite music CDs and DVDs: *http://tortoisesvn.tigris.org/donate.html* Please also have a look at the list of people who contributed to the project by sending in patches or translations.

Chapitre 5. Le programme SubWCRev

SubWCRev est un programme console Windows qui peut être utilisé pour lire le statut d'une copie de travail Subversion et exécuter facultativement la substitution de mots-clé dans un fichier modèle. C'est souvent utilisé comme une partie du processus de génération comme un moyen d'incorporer l'information de la copie de travail dans l'objet que vous générez. Typiquement cela pourrait être utilisé pour inclure le numéro de révision dans une boîte d'« À propos ».

5.1. La ligne de commande SubWCRev

SubWCRev lit le statut Subversion de tous les fichiers d'une copie de travail, en excluant par défaut les externes. Il enregistre le plus haut numéro de révision livrée trouvé et la date de cette révision, Il enregistre aussi s'il y a des modifications locales dans la copie de travail, ou des révisions mélangées. Le numéro de révision, la plage de révisions mises à jour et le statut de modification sont affichés sur stdout.

SubWCRev.exe est appelé de la ligne de commande ou d'un script et est contrôlé en utilisant les paramètres de ligne de commande.

```
SubWCRev CheminCopieTravail [VersionSrcFichier VersionDstFichier] [-nmdfe]
```

`CheminCopieTravail` est le chemin de la copie de travail étant vérifiée. Vous pouvez seulement utiliser SubWCRev sur les copies de travail, et pas directement sur le référentiel. Le chemin peut être absolu ou relatif au répertoire de travail courant.

Si vous voulez que SubWCRev exécute la substitution de mots-clé, pour que les champs comme la révision du référentiel et l'URL soient sauvegardées dans un fichier texte, vous devez fournir un fichier modèle `FichierVersionSrc` et un fichier de sortie `FichierVersionDst` qui contient la version substituée du modèle.

Il y a un certain nombre de commutateurs facultatifs qui affectent la façon dont fonctionne SubWCRev. Si vous en utilisez plus d'un, ils doivent être spécifiés comme un groupe simple, par exemple `-nm`, et non `-n -m`.

Aller sur...	Description
-n	Si ce commutateur est renseigné, SubWCRev se fermera avec `ERRORLEVEL 7` si la copie de travail contient des modifications locales. Cela peut être utile pour empêcher une compilation incluant des changements non livrés.
-m	Si ce commutateur est renseigné, SubWCRev se fermera avec `ERRORLEVEL 8` si la copie de travail contient des révisions mélangées. Cela peut être utile pour empêcher la compilation d'une version partiellement mise à jour.
-d	Si ce commutateur est donné, SubWCRev sortira avec `ERRORLEVEL 9` si le fichier de destination existe déjà .
-f	Si ce commutateur est donné, SubWCRev inclura la dernière révision changée des dossiers. Le comportement par défaut consiste à utiliser seulement les fichiers lors de l'obtenant des numéros de révision.
-e	Si ce commutateur est donné, SubWCRev examinera les répertoires qui sont inclus avec `svn:externals`, mais seulement s'ils sont du même référentiel. Le comportement par défaut est d'ignorer les éléments externes.
-x	Si ce commutateur est donné, SubWCRev renverra les numéro de révision en héxadécimal.
-X	Si ce commutateur est donné, SubWCRev renverra les numéros de révision en héxadécimal, préfixés de '0X'.

Tableau 5.1. Liste des commutateurs de ligne de commande disponibles

5.2. Substitution de mot-clés

Si un fichier source et un fichier de destination sont fournis, SubWCRev copie la source à la destination, en exécutant la substitution de mots-clé comme suit :

Mot-clé	Description
$WCREV$	Remplacé par la plus haute révision livrée dans la copie de travail.
$WCDATE$	Replaced with the commit date/time of the highest commit revision. By default, international format is used: `yyyy-mm-dd hh:mm:ss`. Alternatively, you can specify a custom format which will be used with `strftime()`, for example: `$WCDATE=%a %b %d %I:%M:%S %p $`. For a list of available formatting characters, look at the *Aide en ligne* [http://www.cppreference.com/stddate/strftime.html].
$WCNOW$	Replaced with the current system date/time. This can be used to indicate the build time. Time formatting can be used as described for `$WCDATE $`.
$WCRANGE$	Remplacé par la plage de révisions mises à jour dans la copie de travail. Si la copie de travail est dans un état cohérent, ce sera une seule révision. Si la copie de travail contient des révisions mélangées, soit parce qu'elle est périmée, soit à cause d'une mise à jour à la révision délibérée, donc la plage sera affichée dans la forme 100:200
$WCMIXED$	`$WCMIXED?VTexte:FTexte$` est remplacé par `TText` s'il y a des révisions avec des mises à jour mélangées, ou `FText` sinon.
$WCMODS$	`$WCMODS?VTexte:FTexte$` est remplacé par `TText` s'il y a des modifications locales, ou `FText` sinon.
$WCURL$	Remplacé par l'URL du référentiel du chemin de la copie de travail passée à SubWCRev.
$WCINSVN$	`$WCINSVN?TText:FText$` est remplacé par `TText` si l'élément est versionné, ou `FText` sinon.
$WCNEEDSLOCK$	`$WCNEEDSLOCK?TText:FText$` est remplacé par `TText` si l'élément à la propriété `svn:needs-lock` activée, ou `FText` sinon.
$WCISLOCKED$	`$WCISLOCKED?TText:FText$` est remplacé par `TText` si l'élément est verrouillé, ou `FText` sinon.
$WCLOCKDATE$	Replaced with the lock date. Time formatting can be used as described for `$WCDATE$`.
$WCLOCKOWNER$	Remplacer par le nom du propriétaire du verrou
$WCLOCKCOMMENT$	Remplacé par le commentaire du verrou

Tableau 5.2. Liste des commutateurs de ligne de commande disponibles

Astuce

Some of these keywords apply to single files rather than to an entire working copy, so it only makes sense to use these when SubWCRev is called to scan a single file. This applies to `$WCINSVN$`, `$WCNEEDSLOCK$`, `$WCISLOCKED$`, `$WCLOCKDATE$`, `$WCLOCKOWNER$` and `$WCLOCKCOMMENT$`.

5.3. Exemple de mot-clé

L'exemple ci-dessous montre comment les mots-clés dans un fichier modèle sont substitués dans le fichier de résultat.

```
// Fichier de test pour SubWCRev: testfile.tmpl

char *Revision = "$WCREV$";
char *Modified = "$WCMODS?Modified:Not modified$";
char *Date     = "$WCDATE$";
char *Range    = "$WCRANGE$";
char *Mixed    = "$WCMIXED?Mixed revision WC:Not mixed$";
char *URL      = "$WCURL$";

#if $WCMODS?1:0$
#error Source is modified
#endif

// Fin du fichier
```

Après l'exécution de `SubWCRev.exe path\to\workingcopy testfile.tmpl testfile.txt`, le fichier de sortie `testfile.txt` doit être de la forme :

```
// Fichier de test pour SubWCRev: testfile.txt

char *Revision = "3701";
char *Modified = "Modified";
char *Date     = "2005/06/15 11:15:12";
char *Range    = "3699:3701";
char *Mixed    = "Mixed revision WC";
char *URL      = "http://nom.de.domaine.du.projet.org/svn/trunk/src";

#if 1
#error Source is modified
#endif

// End of file
```

Astuce

A file like this will be included in the build so you would expect it to be versioned. Be sure to version the template file, not the generated file, otherwise each time you regenerate the version file you need to commit the change, which in turn means the version file needs to be updated.

5.4. Interface COM

Si vous avec besoin d'accéder à l'information sur les révisions depuis d'autres programmes que Subversion, vous pouvez utiliser l'objet COM SubWCRev comme interface. L'objet à créer est `SubWCRev.object`, et les méthodes suivantes sont supportées :

Méthode	Description
.GetWCInfo	This method traverses the working copy gathering the revision information. Naturally you must call this before you can access the information using the remaining methods. The first parameter is the path. The second parameter should be true if you want to include folder revisions. Equivalent to the `-f` command line switch. The third parameter should be true if you want to include svn:externals. Equivalent to the `-e` command line switch.

Méthode	Description
.Revision	Le numéro de version le pus élevé de la copie de travail. Equivalent à `$WCREV$`
.Date	Date/heure de la livraison de la dernière révision. Equivalent à `$WCDATE$`
.Author	L'auteur de la version de livraison la plus élevée, c'est à dire, la dernière personne à avoir fait une livraison.
.MinRev	Le numéro de version le moins élevé, comme montré dans `$WCRANGE$`.
.MaxRev	Le numéro de version le plus élevé, comme montré dans `$WCRANGE$`.
.HasModifications	Vrais s'il y a des modifications locales.
.Url	Remplacé par l'URL du référentiel du chemin de la copie de travail passée à `GetWCInfo`. Equivalent à `$WCURL$`.
.IsSvnItem	Vrai si l'élément est versionné
.NeedsLocking	Vrai si la propriété `svn:needs-lock` est activée pour l'élément.
.IsLocked	Vrai si l'élément est verrouillé
.LockCreationDate	String representing the date when the lock was created, or an empty string if the item is not locked.
.LockOwner	String representing the lock owner, or an empty string if the item is not locked.
.LockComment	Le message renseigné au moment du verrouillage.

Tableau 5.3. Les méthodes COM/automation sont supportées

Les exemples suivants montrent comment l'interface devrait être utilisée.

```
// testCOM.js - fichier javascript
// script de test pour l'objet COM SubWCRev

filesystem = new ActiveXObject("Scripting.FileSystemObject");

revObject1 = new ActiveXObject("SubWCRev.object");
revObject2 = new ActiveXObject("SubWCRev.object");
revObject3 = new ActiveXObject("SubWCRev.object");
revObject4 = new ActiveXObject("SubWCRev.object");

revObject1.GetWCInfo(
    filesystem.GetAbsolutePathName("."), 1, 1);
revObject2.GetWCInfo(
    filesystem.GetAbsolutePathName(".."), 1, 1);
revObject3.GetWCInfo(
    filesystem.GetAbsolutePathName("SubWCRev.cpp"), 1, 1);
revObject4.GetWCInfo(
    filesystem.GetAbsolutePathName("..\\.."), 1, 1);

wcInfoString1 = "Revision = " + revObject1.Revision +
    "\nMin Revision = " + revObject1.MinRev +
    "\nMax Revision = " + revObject1.MaxRev +
    "\nDate = " + revObject1.Date +
    "\nURL = " + revObject1.Url + "\nAuthor = " +
    revObject1.Author + "\nHasMods = " +
```

```
    revObject1.HasModifications + "\nIsSvnItem = " +
    revObject1.IsSvnItem + "\nNeedsLocking = " +
    revObject1.NeedsLocking + "\nIsLocked = " +
    revObject1.IsLocked + "\nLockCreationDate = " +
    revObject1.LockCreationDate + "\nLockOwner = " +
    revObject1.LockOwner + "\nLockComment = " +
    revObject1.LockComment;
wcInfoString2 = "Revision = " + revObject2.Revision +
    "\nMin Revision = " + revObject2.MinRev +
    "\nMax Revision = " + revObject2.MaxRev +
    "\nDate = " + revObject2.Date +
    "\nURL = " + revObject2.Url + "\nAuthor = " +
    revObject2.Author + "\nHasMods = " +
    revObject2.HasModifications + "\nIsSvnItem = " +
    revObject2.IsSvnItem + "\nNeedsLocking = " +
    revObject2.NeedsLocking + "\nIsLocked = " +
    revObject2.IsLocked + "\nLockCreationDate = " +
    revObject2.LockCreationDate + "\nLockOwner = " +
    revObject2.LockOwner + "\nLockComment = " +
    revObject2.LockComment;
wcInfoString3 = "Revision = " + revObject3.Revision +
    "\nMin Revision = " + revObject3.MinRev +
    "\nMax Revision = " + revObject3.MaxRev +
    "\nDate = " + revObject3.Date +
    "\nURL = " + revObject3.Url + "\nAuthor = " +
    revObject3.Author + "\nHasMods = " +
    revObject3.HasModifications + "\nIsSvnItem = " +
    revObject3.IsSvnItem + "\nNeedsLocking = " +
    revObject3.NeedsLocking + "\nIsLocked = " +
    revObject3.IsLocked + "\nLockCreationDate = " +
    revObject3.LockCreationDate + "\nLockOwner = " +
    revObject3.LockOwner + "\nLockComment = " +
    revObject3.LockComment;
wcInfoString4 = "Revision = " + revObject4.Revision +
    "\nMin Revision = " + revObject4.MinRev +
    "\nMax Revision = " + revObject4.MaxRev +
    "\nDate = " + revObject4.Date +
    "\nURL = " + revObject4.Url + "\nAuthor = " +
    revObject4.Author + "\nHasMods = " +
    revObject4.HasModifications + "\nIsSvnItem = " +
    revObject4.IsSvnItem + "\nNeedsLocking = " +
    revObject4.NeedsLocking + "\nIsLocked = " +
    revObject4.IsLocked + "\nLockCreationDate = " +
    revObject4.LockCreationDate + "\nLockOwner = " +
    revObject4.LockOwner + "\nLockComment = " +
    revObject4.LockComment;

WScript.Echo(wcInfoString1);
WScript.Echo(wcInfoString2);
WScript.Echo(wcInfoString3);
WScript.Echo(wcInfoString4);
```

Chapitre 6. IBugtraqProvider interface

To get a tighter integration with issue trackers than by simply using the `bugtraq:` properties, TortoiseSVN can make use of COM plugins. With such plugins it is possible to fetch information directly from the issue tracker, interact with the user and provide information back to TortoiseSVN about open issues, verify log messages entered by the user and even run actions after a successful commit to e.g, close an issue.

Nous ne pouvons pas vous indiquer comment développer un objet COM dans votre langage préféré, mais des exemples de greffons en C++/ATL et C# sont disponibles dans notre référentiel dans le répertoire `contrib/issue-tracker-plugins`. Vous pouvez également y trouver les fichiers à inlcure pour compiler votre greffon. (Section 3, « TortoiseSVN est gratuit ! » explique comment accéder au référentiel).

6.1. L'interface de IBugtraqProvider

TortoiseSVN 1.5 peut être agrémenté de greffons qui ajouteront une interface pour IBugtraqProvider. Cette dernière fournira quelques méthodes que les greffons pourront utiliser pour s'interfacer avec le système de suivi d'anomalies logicielles.

```
HRESULT ValidateParameters (
  // Fenêtre parent pour toutes les interfaces devant être
  // affichées pendant la validation.
  [in] HWND hParentWnd,

  // La chaine en paramètre doit être validée.
  [in] BSTR parameters,

  // La chaine est elle valide ?
  [out, retval] VARIANT_BOOL *valid
);
```

Cette méthode est appele depuis la fenêtre des paramètres là où l'utilisateur peut ajouter et configurer le greffon. La chaine `parametres` peut être utilisée par un greffon pour avoir plus d'informations, i.e., les identifiants, l'URL du bug tracker, etc. Le greffon doit vérifier la chaine `parametres` et montrer une fenêtre d'erreur si la chaine n'est pas valide. Le paramètre `hParentWnd` permet au plugin d'accéder à la fenêtre mère. Le greffon DOIT renvoyer TRUE si la chaine `parametres` est validée. Si le greffon renvoie FALSE, la fenêtre de paramètrage n'autorisera pas l'utilisateur à ajouter le greffon au chemin d'une copie de travail.

```
HRESULT GetLinkText (
  // Fenêtre parent pour toutes les interfaces (d'erreur) devant être affiché.
  [in] HWND hParentWnd,

  // La chaine en paramètre, au cas où vous devriez communiquer avec votre
  // service web (i.e.) pour savoir qu'est ce qu'un texte correct.
  [in] BSTR parameters,

  // Quel texte voulez vous afficher ?
  // Utilisez le thread local courant.
  [out, retval] BSTR *linkText
);
```

Le greffon peut fournir une chaine ici laquelle sera utilisée dans la fenêtre de livraison de TortoiseSVN par le bouton qui appelle le greffon, i.e., "Choisir le bug" ou "Selectionner le ticket". Assurez vous que la

chaine n'est pas trop longue, sinon elle ne rentrera pas dans le bouton. Si la méthode retourne une erreur (i.e., `E_NOTIMPL`), un texte par défaut est utilisé pour le bouton.

```
HRESULT GetCommitMessage (
  // Parent window for your provider's UI.
  [in] HWND hParentWnd,

  // Parameters for your provider.
  [in] BSTR parameters,
  [in] BSTR commonRoot,
  [in] SAFEARRAY(BSTR) pathList,

  // The text already present in the commit message.
  // Your provider should include this text in the new message,
  // where appropriate.
  [in] BSTR originalMessage,

  // The new text for the commit message.
  // This replaces the original message.
  [out, retval] BSTR *newMessage
);
```

This is the main method of the plugin. This method is called from the TortoiseSVN commit dialog when the user clicks on the plugin button. The `parameters` string is the string the user has to enter in the settings dialog when he configures the plugin. Usually a plugin would use this to find the URL of the issue tracker and/or login information or more. The `commonRoot` string contains the parent path of all items selected to bring up the commit dialog. Note that this is *not* the root path of all items which the user has selected in the commit dialog. The `pathList` parameter contains an array of paths (as strings) which the user has selected for the commit. The `originalMessage` parameter contains the text entered in the log message box in the commit dialog. If the user has not yet entered any text, this string will be empty. The `newMessage` return string is copied into the log message edit box in the commit dialog, replacing whatever is already there. If a plugin does not modify the `originalMessage` string, it must return the same string again here, otherwise any text the user has entered will be lost.

6.2. L'interface de IBugtraqProvider2

Dans TortoiseSVN 1.6 une nouvelle interface a été ajoutée, augmentant le potentiel des greffons. Cet interface IBugtraqProvider2 hérite de IBugtraqProvider.

```
HRESULT GetCommitMessage2 (
  // Parent window for your provider's UI.
  [in] HWND hParentWnd,

  // Parameters for your provider.
  [in] BSTR parameters,
  // The common URL of the commit
  [in] BSTR commonURL,
  [in] BSTR commonRoot,
  [in] SAFEARRAY(BSTR) pathList,

  // The text already present in the commit message.
  // Your provider should include this text in the new message,
  // where appropriate.
  [in] BSTR originalMessage,

  // You can assign custom revision properties to a commit
  // by setting the next two params.
```

```
  // note: Both safearrays must be of the same length.
  //       For every property name there must be a property value!

  // The content of the bugID field (if shown)
  [in] BSTR bugID,

  // Modified content of the bugID field
  [out] BSTR * bugIDOut,

  // The list of revision property names.
  [out] SAFEARRAY(BSTR) * revPropNames,

  // The list of revision property values.
  [out] SAFEARRAY(BSTR) * revPropValues,

  // The new text for the commit message.
  // This replaces the original message
  [out, retval] BSTR * newMessage
);
```

This method is called from the TortoiseSVN commit dialog when the user clicks on the plugin button. This method is called instead of `GetCommitMessage()`. Please refer to the documentation for `GetCommitMessage` for the parameters that are also used there. The parameter `commonURL` is the parent URL of all items selected to bring up the commit dialog. This is basically the URL of the `commonRoot` path. The parameter `bugID` contains the content of the bug-ID field (if it is shown, configured with the property `bugtraq:message`). The return parameter `bugIDOut` is used to fill the bug-ID field when the method returns. The `revPropNames` and `revPropValues` return parameters can contain name/value pairs for revision properties that the commit should set. A plugin must make sure that both arrays have the same size on return! Each property name in `revPropNames` must also have a corresponding value in `revPropValues`. If no revision properties are to be set, the plugin must return empty arrays.

```
HRESULT CheckCommit (
  [in] HWND hParentWnd,
  [in] BSTR parameters,
  [in] BSTR commonURL,
  [in] BSTR commonRoot,
  [in] SAFEARRAY(BSTR) pathList,
  [in] BSTR commitMessage,
  [out, retval] BSTR * errorMessage
);
```

This method is called right before the commit dialog is closed and the commit begins. A plugin can use this method to validate the selected files/folders for the commit and/or the commit message entered by the user. The parameters are the same as for `GetCommitMessage2()`, with the difference that `commonURL` is now the common URL of all *checked* items, and `commonRoot` the root path of all checked items. The return parameter `errorMessage` must either contain an error message which TortoiseSVN shows to the user or be empty for the commit to start. If an error message is returned, TortoiseSVN shows the error string in a dialog and keeps the commit dialog open so the user can correct whatever is wrong. A plugin should therefore return an error string which informs the user *what* is wrong and how to correct it.

```
HRESULT  OnCommitFinished (
  // Parent window for any (error) UI that needs to be displayed.
  [in] HWND hParentWnd,

  // The common root of all paths that got committed.
```

```
  [in] BSTR commonRoot,

  // All the paths that got committed.
  [in] SAFEARRAY(BSTR) pathList,


  // The text already present in the commit message.
  [in] BSTR logMessage,

  // The revision of the commit.
  [in] ULONG revision,


  // An error to show to the user if this function
  // returns something else than S_OK
  [out, retval] BSTR * error
);
```

This method is called after a successful commit. A plugin can use this method to e.g., close the selected issue or add information about the commit to the issue. The parameters are the same as for `GetCommitMessage2`.

```
HRESULT HasOptions(
  // Si le fournisseur fournit des options
  [out, retval] VARIANT_BOOL *ret
);
```

This method is called from the settings dialog where the user can configure the plugins. If a plugin provides its own configuration dialog with `ShowOptionsDialog`, it must return TRUE here, otherwise it must return FALSE.

```
HRESULT ShowOptionsDialog(
  // Parent window for the options dialog
  [in] HWND hParentWnd,

  // Parameters for your provider.
  [in] BSTR parameters,

  // The parameters string
  [out, retval] BSTR * newparameters
);
```

Cette méthode est appellée depuis la fenêtre de configuration lorque l'utilisateur clique sur le bouton "Options" qui s'affiche quand `HasOptions` renvoie TRUE. Un plugin peut afficher une fenêtre d'options pour faciliter sa configuration par l'utilisateur. La chaîne de charactères `parameters` contient les informations déjà entrées dans les paramètres du plugin. La chaîne de charactère `newparameters` renvoyée doit contenir la chaîne de charactères construite par le plugin via les informations collectées dans la fenêtre d'options. Cette chaîne, `parameters`, est envoyée à toutes les autres méthodes de type IBugtraqProvider et IBugtraqProvider2.

Annexe A. Foire aux questions (FAQ)

Because TortoiseSVN is being developed all the time it is sometimes hard to keep the documentation completely up to date. We maintain an *online FAQ* [http://tortoisesvn.tigris.org/faq.html] which contains a selection of the questions we are asked the most on the TortoiseSVN mailing lists <dev@tortoisesvn.tigris.org> and <users@tortoisesvn.tigris.org>.

We also maintain a project *Issue Tracker* [http://issues.tortoisesvn.net] which tells you about some of the things we have on our To-Do list, and bugs which have already been fixed. If you think you have found a bug, or want to request a new feature, check here first to see if someone else got there before you.

Le meilleur endroit pour poser une question à laquelle vous n'avez pas trouvé de réponse est l'une des listes de diffusion. <users@tortoisesvn.tigris.org> est celle à utiliser pour les questions relatives à TortoiseSVN. Si vous voulez aider au développement de TortoiseSVN, vous devriez alors participer aux débats sur <dev@tortoisesvn.tigris.org>.

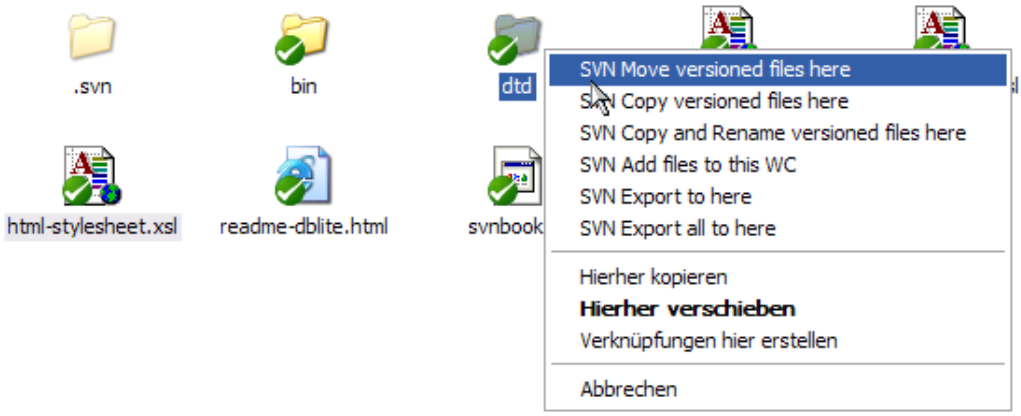
Annexe B. Comment faire pour...

Cette annexe contient les solutions aux problèmes/questions que vous pourriez avoir en utilisant TortoiseSVN.

B.1. Déplacer/copier beaucoup de fichiers en une fois

Déplacer/copier de simples fichiers peut être fait en utilisant TortoiseSVN → Renommer.... Mais si vous voulez déplacer/copier beaucoup de fichiers, cette façon est bien trop lente et demande trop de travail.

The recommended way is by right-dragging the files to the new location. Simply right-click on the files you want to move/copy without releasing the mouse button. Then drag the files to the new location and release the mouse button. A context menu will appear where you can either choose Context Menu → SVN Copy versioned files here. or Context Menu → SVN Move versioned files here.



B.2. Forcer les utilisateurs à entrer un commentaire

Il y a deux façons d'empêcher les utilisateurs de livrer avec un commentaire vide. L'une est spécifique à TortoiseSVN, l'autre fonctionne pour tous les clients Subversion, mais exige l'accès au serveur directement.

B.2.1. Script hook sur le serveur

Si vous avez un accès direct au serveur du référentiel, vous pouvez installer un script hook pre-commit qui rejette toutes les livraisons avec des commentaires vides ou trop courts.

In the repository folder on the server, there's a sub-folder `hooks` which contains some example hook scripts you can use. The file `pre-commit.tmpl` contains a sample script which will reject commits if no log message is supplied, or the message is too short. The file also contains comments on how to install/use this script. Just follow the instructions in that file.

Cette méthode est celle recommandée si vos utilisateurs utilisent aussi d'autres clients Subversion que TortoiseSVN. L'inconvénient réside dans le fait que la livraison est rejetée par le serveur et donc les utilisateurs obtiendront un message d'erreur. Le client ne peut pas savoir avant la livraison qu'elle sera rejetée. Si vous voulez que TortoiseSVN ait le bouton OK désactivé jusqu'à ce que le commentaire soit assez long alors veuillez utiliser la méthode décrite ci-dessous.

B.2.2. Propriétés de projet

TortoiseSVN utilise des propriétés pour contrôler certaines de ses fonctionnalités. Une de ces propriétés est la propriété `tsvn:minlogsize`.

Si vous définissez cette propriété sur un dossier, alors TortoiseSVN désactivera le bouton OK dans toutes les boîtes de dialogues de livraison jusqu'à ce que l'utilisateur ait entré un commentaire avec au moins la longueur indiquée dans la propriété.

Pour des informations détaillées sur ces propriétés de projet, veuillez vous référer à Section 4.17, « Configuration des projets »

B.3. Mettre à jour les fichiers sélectionnés à partir du référentiel

Normalement, vous mettez à jour votre copie de travail en utilisant TortoiseSVN → Mettre à jour. Mais si vous voulez seulement récupérer les nouveaux fichiers qu'un collègue a ajoutés sans fusionner les changements d'autres fichiers dans même temps, vous avez besoin d'une approche différente.

Utilisez TortoiseSVN → Vérifier les modifications. Et cliquez sur Vérifier le référentiel pour voir ce qui a changé dans le référentiel. Sélectionnez les fichiers que vous voulez mettre à jour localement, utilisez ensuite le menu contextuel pour ne mettre à jour que ces fichiers.

B.4. Annuler des révisions dans le référentiel

B.4.1. Utiliser la boîte de dialogue du journal de révision

La façon la plus facile d'annuler les changements d'une seule révision, ou d'une plage de révisions, est d'utiliser la boîte de dialogue du journal de révision. C'est aussi la méthode à utiliser si vous voulez renoncer aux changements récents et faire d'une révision précédente la nouvelle révision HEAD.

1. Sélectionnez le fichier ou le dossier pour lequel vous voulez annuler les changements. Si vous voulez annuler tous les changements, cela devrait être le dossier au niveau supérieur.
2. Sélectionnez TortoiseSVN → Voir le journal pour afficher une liste des révisions. Vous pouvez avoir à utiliser Afficher tout ou 100 suivants pour afficher les révisions qui vous intéressent.
3. Select the revision you wish to revert. If you want to undo a range of revisions, select the first one and hold the **Shift** key while selecting the last one. Note that for multiple revisions, the range must be unbroken with no gaps. Right click on the selected revision(s), then select Context Menu → Revert changes from this revision.
4. Or if you want to make an earlier revision the new HEAD revision, right click on the selected revision, then select Context Menu → Revert to this revision. This will discard *all* changes after the selected revision.

Vous avez annuler les changements dans votre copie de travail. Vérifiez les résultats, puis livrez les changements.

B.4.2. Utiliser la boîte de dialogue fusionner

Pour annuler une plus grande plage de révisions, vous pouvez utiliser la boîte de dialogue Fusionner. La méthode précédente utilise la fusion en coulisses ; cette méthode l'utilise explicitement.

1. Dans votre copie de travail, sélectionnez TortoiseSVN → Fusionner.
2. In the From: field enter the full folder URL of the branch or tag containing the changes you want to revert in your working copy. This should come up as the default URL.

3. Dans le champ De la révision entrez le numéro de la révision à laquelle vous êtes actuellement. Si vous êtes sûr qu'il n'y a personne d'autre faisant des changements, vous pouvez utiliser la révision HEAD.

4. assurez-vous que la case Utiliser "Depuis :" URL est cochée.

5. In the To Revision field enter the revision number that you want to revert to, namely the one *before* the first revision to be reverted.

6. Cliquez sur OK pour terminer la fusion.

Vous avez annuler les changements dans votre copie de travail. Vérifiez les résultats, puis livrez les changements.

B.4.3. Utiliser `svndumpfilter`

Puisque TortoiseSVN ne perd jamais de données, vos révisions « annulées » existent toujours comme révisions intermédiaires dans le référentiel. Seule la révision HEAD a été changée à un état précédent. Si vous voulez faire que les révisions disparaissent complètement de votre référentiel, en effaçant toute trace de leur existence, vous devez utiliser des mesures plus extrêmes. À moins d'avoir une bonne raison pour le faire, ce n'est *pas recommandé*. Une raison possible serait que quelqu'un a livré un document confidentiel à un référentiel public.

The only way to remove data from the repository is to use the Subversion command line tool `svnadmin`. You can find a description of how this works in the *Repository Maintenance* [http://svnbook.red-bean.com/en/1.5/svn.reposadmin.maint.html].

B.5. Compare deux révisions d'un fichier ou d'un répertoire

If you want to compare two revisions in an item's history, for example revisions 100 and 200 of the same file, just use TortoiseSVN → Show Log to list the revision history for that file. Pick the two revisions you want to compare then use Context Menu → Compare Revisions.

If you want to compare the same item in two different trees, for example the trunk and a branch, you can use the repository browser to open up both trees, select the file in both places, then use Context Menu → Compare Revisions.

If you want to compare two trees to see what has changed, for example the trunk and a tagged release, you can use TortoiseSVN → Revision Graph Select the two nodes to compare, then use Context Menu → Compare HEAD Revisions. This will show a list of changed files, and you can then select individual files to view the changes in detail. You can also export a tree structure containing all the changed files, or simply a list of all changed files. Read Section 4.10.3, « Comparer des répertoires » for more information. Alternatively use Context Menu → Unified Diff of HEAD Revisions to see a summary of all differences, with minimal context.

B.6. Inclure un sous-projet commun

Parfois vous voudrez inclure un autre projet dans votre copie de travail, peut-être du code de bibliothèque. Vous ne voulez pas faire un duplicata de ce code dans votre référentiel parce qu'alors vous perdriez la connexion avec le code original (et maintenu). Ou peut-être vous avez plusieurs projets qui partagent le code fondamental. Il y a au moins 3 façons de gérer cela.

B.6.1. Utiliser svn:externals

Set the `svn:externals` property for a folder in your project. This property consists of one or more lines; each line has the name of a sub-folder which you want to use as the checkout folder for common

code, and the repository URL that you want to be checked out there. For full details refer to Section 4.18, « Eléments externes ».

Commit the new folder. Now when you update, Subversion will pull a copy of that project from its repository into your working copy. The sub-folders will be created automatically if required. Each time you update your main working copy, you will also receive the latest version of all external projects.

Si le projet externe est dans le même référentiel, les changements que vous y faites seront inclus dans la liste de livraisons quand vous livrerez votre projet principal.

Si le projet externe est dans un référentiel différent, les changements que vous faites au projet externe seront signalés quand vous livrerez le projet principal, mais vous devez livrer ces changements externes séparément.

Of the three methods described, this is the only one which needs no setup on the client side. Once externals are specified in the folder properties, all clients will get populated folders when they update.

B.6.2. Utiliser une copie de travail nichée

Créez un nouveau dossier dans votre projet pour contenir le code commun, mais ne l'ajoutez pas à Subversion

Sélectionnez TortoiseSVN → Extraire pour le nouveau dossier et extrayez une copie du code commun. Vous avez maintenant une copie de travail séparée emboîtée dans votre copie de travail principale.

Les deux copies de travail sont indépendantes. Quand vous livrez des changements au parent, les changements à la CdT emboîtée sont ignorés. De même quand vous mettez à jour le parent, la CdT emboîtée n'est pas mise à jour.

B.6.3. Utiliser un emplacement relatif

Si vous utilisez le même code fondamental commun dans plusieurs projets et vous ne voulez pas en garder plusieurs copies de travail pour chaque projet qui l'utilise, vous pouvez juste l'extraire à un emplacement séparé qui est lié à tous les autres projets qui l'utilisent. Par exemple :

```
C:\Projets\Proj1
C:\Projets\Proj2
C:\Projets\Proj3
C:\Projets\Commun
```

et faites référence au code commun en utilisant un chemin relatif, par exemple `..\..\Commun \DSPcore`.

Si vos projets sont dispersés dans des emplacements sans rapport, vous pouvez utiliser une variante, qui consiste à mettre le code commun dans un emplacement et utilisez la substitution de lettre de disque pour affecter cet emplacement à quelque chose que vous pouvez coder en dur dans vos projets, par exemple Extrayez le code commun dans `D:\Documents\Framework` ou `C:\Documents and Settings \{login}\My Documents\framework` puis utilisez

```
SUBST X: "D:\Documents\framework"
```

pour créer l'affectation de disque utilisé dans votre code source. Votre code peut alors utiliser des emplacements absolus.

```
#include "X:\superio\superio.h"
```

Cette méthode ne fonctionnera que dans un environnement tout PC et vous devrez documenter les affectations de disque requises pour que votre équipe sache où se trouvent ces fichiers mystérieux.

Cette méthode est strictement utilisée dans des environnements de développement fermés et n'est pas recommandée pour une utilisation générale.

B.7. Créer un raccourci vers un référentiel

If you frequently need to open the repository browser at a particular location, you can create a desktop shortcut using the automation interface to TortoiseProc. Just create a new shortcut and set the target to:

```
TortoiseProc.exe /command:repobrowser /path:"url/to/repository"
```

Of course you need to include the real repository URL.

B.8. Ignorer les fichiers déjà versionnés

Si vous avez accidentellement ajouté des fichiers qui devraient avoir été ignorés, comment les retirez-vous du contrôle de version sans les perdre ? Peut-être que vous avez votre propre fichier de configuration d'IDE qui ne fait pas partie du projet, mais qui vous a pris beaucoup de temps à configurer juste comme vous l'aimez.

Si vous n'avez pas encore livré l'ajout, tout que vous avez alors à faire est d'utiliser TortoiseSVN → Revenir en arrière... pour annuler l'ajout. Vous devriez alors ajouter les fichiers à la liste des ignorés pour qu'ils ne soient plus ajoutés plus tard par erreur à nouveau.

Si les fichiers sont déjà dans le référentiel, vous devez faire un peu plus de travail.

1. Gardez la touche **Shift** appuyée pour afficher le menu contextuel étendu et utilisez TortoiseSVN → Supprimer (garder en local) pour marquer le fichier/répertoire comme étant destiné à être supprimé du référentiel sans perdre la copie locale.
2. TortoiseSVN → Livraison le répertoire parent.
3. Ajoutez le fichier/dossier à la liste des ignorés pour que vous n'ayez plus le problème.

B.9. Retirer une copie de travail du contrôle de version

Si vous souhaitez retirer tous les répertoires `.svn` de votre version de travail vous pouvez tout simplement utiliser la commande export. Lisez Section 4.26.1, « Retirer une copie de travail du contrôle de version » pour plus d'informations sur cette commande.

B.10. Retirer une copie de travail

If you have a working copy which you no longer need, how do you get rid of it cleanly? Easy - just delete it in Windows Explorer! Working copies are private local entities, and they are self-contained.

Annexe C. Trucs Utiles Pour Les Administrateurs

Cette annexe contient les solutions aux problèmes/questions que vous pourriez avoir quand vous êtes responsable du déploiement de TortoiseSVN sur plusieurs ordinateurs client.

C.1. Déployer TortoiseSVN via les stratégies de groupe

L'installeur TortoiseSVN est un fichier msi, ce qui signifie que vous ne devriez avoir aucun problème pour ajouter ce fichier msi à la stratégie de groupe de votre contrôleur de domaine.

A good walk-through on how to do that can be found in the knowledge base article 314934 from Microsoft: *http://support.microsoft.com/?kbid=314934*.

Versions 1.3.0 and later of TortoiseSVN must be installed under *Computer Configuration* and not under *User Configuration*. This is because those versions need the new CRT and MFC DLLs, which can only be deployed *per computer* and not *per user*. If you really must install TortoiseSVN on a per user basis, then you must first install the MFC and CRT package version 8 from Microsoft on each computer you want to install TortoiseSVN as per user.

C.2. Rediriger la vérification de mise à niveau

TortoiseSVN vérifie s'il y a une nouvelle version disponible tous les deux ou trois jours. S'il y a une version plus récente disponible, une boîte de dialogue s'affiche pour en informer l'utilisateur.

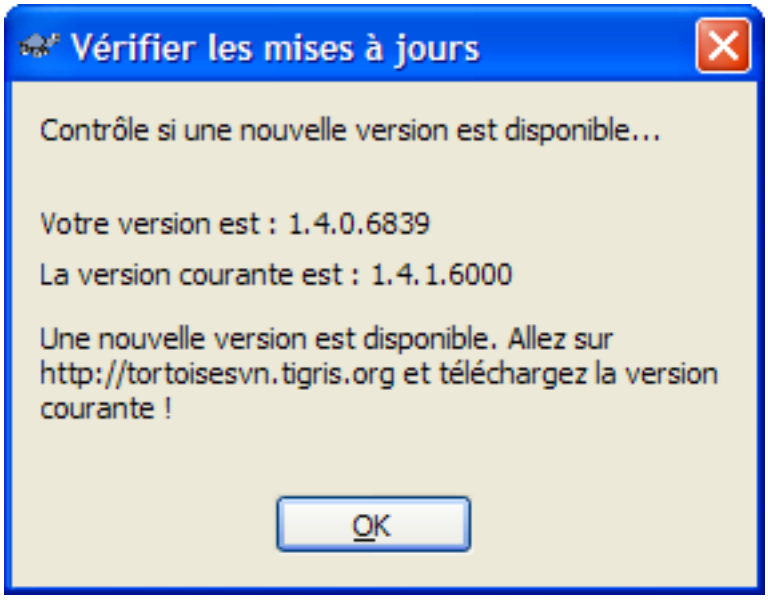


Figure C.1. La boîte de dialogue Mettre à jour

Si vous êtes responsable de beaucoup d'utilisateurs dans votre domaine, vous pourriez vouloir que vos utilisateurs n'utilisent que les versions que vous avez approuvées et qu'ils n'installent pas toujours la dernière version. Vous ne voulez pas probablement pas que la boîte de dialogue de mise à niveau s'affiche pour que vos utilisateurs n'aillent pas mettre à niveau immédiatement.

Versions 1.4.0 and later of TortoiseSVN allow you to redirect that upgrade check to your intranet server. You can set the registry key `HKCU\Software\TortoiseSVN\UpdateCheckURL` (string value) to an URL pointing to a text file in your intranet. That text file must have the following format:

```
1.4.1.6000
A new version of TortoiseSVN is available for you to download!
http://192.168.2.1/downloads/TortoiseSVN-1.4.1.6000-svn-1.4.0.msi
```

The first line in that file is the version string. You must make sure that it matches the exact version string of the TortoiseSVN installation package. The second line is a custom text, shown in the upgrade dialog. You can write there whatever you want. Just note that the space in the upgrade dialog is limited. Too long messages will get truncated! The third line is the URL to the new installation package. This URL is opened when the user clicks on the custom message label in the upgrade dialog. You can also just point the user to a web page instead of the MSI file directly. The URL is opened with the default web browser, so if you specify a web page, that page is opened and shown to the user. If you specify the MSI package, the browser will ask the user to save the MSI file locally.

C.3. Mettre la variable d'environnement SVN_ASP_DOT_NET_HACK

À partir des versions 1.4.0 et supérieures, l'installeur de TortoiseSVN ne fournit plus à l'utilisateur l'option pour mettre la variable d'environnement

Mais cette option est seulement cachée pour l'utilisateur. Vous pouvez encore forcer l'installeur de TortoiseSVN à mettre cette variable d'environnement en mettant la propriété `ASPDOTNETHACK` à TRUE. Par exemple, vous pouvez démarrer l'installeur comme ceci :

```
msiexec /i TortoiseSVN-1.4.0.msi ASPDOTNETHACK=TRUE
```

C.4. Désactiver les entrées du menu contextuel

As of version 1.5.0 and later, TortoiseSVN allows you to disable (actually, hide) context menu entries. Since this is a feature which should not be used lightly but only if there is a compelling reason, there is no GUI for this and it has to be done directly in the registry. This can be used to disable certain commands for users who should not use them. But please note that only the context menu entries in the *explorer* are hidden, and the commands are still available through other means, e.g. the command line or even other dialogs in TortoiseSVN itself!

The registry keys which hold the information on which context menus to show are `HKEY_CURRENT_USER\Software\TortoiseSVN\ContextMenuEntriesMaskLow` and `HKEY_CURRENT_USER\Software\TortoiseSVN\ContextMenuEntriesMaskHigh`.

Each of these registry entries is a `DWORD` value, with each bit corresponding to a specific menu entry. A set bit means the corresponding menu entry is deactivated.

Valeur	Entrée du menu
0x0000000000000001	Extraire
0x0000000000000002	Mettre à jour
0x0000000000000004	Livrer
0x0000000000000008	Ajouter
0x0000000000000010	Revenir en arrière
0x0000000000000020	Nettoyer
0x0000000000000040	Résoudre
0x0000000000000080	Aller sur...
0x0000000000000100	Importer
0x0000000000000200	Exporter

Valeur	Entrée du menu
0x0000000000000400	Créer un référentiel ici
0x0000000000000800	Branche/Etiquette
0x0000000000001000	Fusionner
0x0000000000002000	Supprimer
0x0000000000004000	Renommer
0x0000000000008000	Mettre à jour à la révision
0x0000000000010000	Voir les différences
0x0000000000020000	Voir le journal
0x0000000000040000	Éditer les conflits
0x0000000000080000	Relocaliser
0x0000000000100000	Vérifier les modifications
0x0000000000200000	Ignorer
0x0000000000400000	Explorateur de référentiel
0x0000000000800000	Annoter
0x0000000001000000	Créer un patch
0x0000000002000000	Appliquer un patch
0x0000000004000000	Graphique de révision
0x0000000008000000	Verrouiller
0x0000000010000000	Relâcher un verrou
0x0000000020000000	Propriétés
0x0000000040000000	Comparer avec l'URL
0x0000000080000000	Supprimer les éléments non versionnés
0x2000000000000000	Réglages
0x4000000000000000	Aide
0x8000000000000000	À propos

Tableau C.1. Entrées du menu et leurs valeurs

Example: to disable the « Relocate » the « Delete unversioned items » and the « Settings » menu entries, add the values assigned to the entries like this:

```
  0x0000000000080000
+ 0x0000000080000000
+ 0x2000000000000000
= 0x2000000080080000
```

The lower `DWORD` value (`0x80080000`) must then be stored in `HKEY_CURRENT_USER\Software\TortoiseSVN\ContextMenuEntriesMaskLow`, the higher `DWORD` value (`0x20000000`) in `HKEY_CURRENT_USER\Software\TortoiseSVN\ContextMenuEntriesMaskHigh`.

Pour réactiver les entrées du menu, supprimer simplement les deux clés de registre.

Annexe D. Automatiser TortoiseSVN

Puisque toutes les commandes pour TortoiseSVN sont contrôlées par des paramètres de ligne de commande, vous pouvez l'automatiser avec des scripts batch ou démarrer des commandes spécifiques et des boîtes de dialogues depuis d'autres programmes (par exemple votre éditeur de texte favori).

Important

Souvenez-vous que TortoiseSVN est un client GUI et ce guide d'automatisation vous montre comment faire les boîtes de dialogues de TortoiseSVN apparaissent pour collecter les entrées utilisateur. Si vous voulez écrire un script qui n'exige aucune entrée, vous devriez utiliser le client en ligne de commande Subversion officiel à la place.

D.1. Commandes de TortoiseSVN

Le programme GUI de TortoiseSVN s'appelle `TortoiseProc.exe`. Toutes les commandes sont spécifiées avec le paramètre `/command:abcd` où `abcd` est le nom de la commande requise. La plupart de ces commandes ont besoin d'au moins un argument de chemin, que l'on donne avec `/path:"un\chemin"`. Dans la table suivante, la commande fait référence au paramètre `/command:abcd` et le chemin se réfère au paramètre `/path:"un\chemin"`.

Puisque certaines des commandes peuvent prendre une liste de chemins cibles (par exemple livrer plusieurs fichiers spécifiques) le paramètre `/path` peut prendre plusieurs chemins, séparés par un caractère `*`.

TortoiseSVN utilise des fichiers temporaires pour passer des arguments multiples entre l'extension du shell et le programme principal. À partir de la version 1.5.0 et supérieures, le paramètre `/notempfile` est obsolète et il est inutile de l'ajouter désormais.

D'habitude, la barre de progression utilisée pour les livraisons, les mises à jour et beaucoup d'autres commandes reste ouverte après que la commande ait fini et jusqu'à ce que l'utilisateur appuie sur le bouton OK. Ce comportement peut être modifié en cochant l'option correspondante dans la boîte de dialogue de configuration. Mais ainsi la barre de progression se fermera à la fin de l'opération, que vous ayez lancé la commande depuis un fichier batch ou depuis le menu contextuel TortoiseSVN.

Pour spécifier un emplacement différent du fichier de configuration, utilisez le paramètre `/configdir:"chemin\vers\répertoire\de\conf"`. Cela remplacera le chemin par défaut, y compris tous les paramètres de la base de registre.

To close the progress dialog at the end of a command automatically without using the permanent setting you can pass the `/closeonend` parameter.

- `/closeonend:0` ne ferme pas la boîte de dialogue automatiquement
- `/closeonend:1` ferme automatiquement s'il n'y a pas d'erreurs
- `/closeonend:2` ferme automatiquement s'il n'y a pas d'erreurs ni de conflits
- `/closeonend:2` ferme automatiquement s'il n'y a pas d'erreurs, de conflits ni de fusions
- `/closeonend:4` ferme automatiquement s'il n'y a pas d'erreurs, de conflits ni de fusions pour les opérations locales

Le tableau ci-dessous liste toutes les commandes qui peuvent être accessibles en utilisant la ligne de commande TortoiseProc.exe. Comme décrit ci-dessus, celles-ci devraient être de la forme `/command:abcd`. Dans le tableau, le préfixe `/command` est omis pour économiser de la place.

Commande	Description
:about	Affiche la boîte de dialogue d'À propos. Elle s'affiche aussi si aucune commande n'est fournie.
:log	Opens the log dialog. The `/path` specifies the file or folder for which the log should be shown. Three additional options can be set: `/startrev:xxx`, `/endrev:xxx` and `/strict`
:checkout	Ouvre la boîte de dialogue d'extraction. Le `/path` spécifie le répertoire cible et le `/url` spécifie l'URL à extraire.
:import	Ouvre la boîte de dialogue d'Importation. Le `/path` spécifie le répertoire contenant les données à importer.
:update	Met à jour la copie de travail dans `/path` vers HEAD. Si l'option `/rev` est fournie alors une boîte de dialogue est affichée pour demander à l'utilisateur à quelle révision la mise à jour devrait se faire. Pour éviter la boîte de dialogue, spécifiez un numéro de révision `/rev:1234`. Les autres options sont `/nonrecursive` et `/ignoreexternals`.
:commit	Ouvre la boîte de dialogue de livraison. Le `/path` spécifie le répertoire cible ou la liste des fichiers à livrer. Vous pouvez aussi spécifier le commutateur `/logmsg` pour passer un commentaire prédéterminé à la boîte de dialogue de livraision. Ou, si vous ne voulez pas passer le commentaire via la ligne de commande, utilisez `/logmsgfile:chemin`, où `chemin` pointe sur un fichier contenant le commentaire. Pour préremplir le champ d'ID de bug (dans le cas où vous avez défini correctement la propriété d'intégration aux traqueurs de bug), vous pouvez utiliser le `/bugid: "l'id du bug ici"`.
:add	Ajoute les fichiers de `/path` au contrôle de version.
:revert	Annule les modifications locales d'une copie de travail. Le `/path` indique quels éléments annuler.
:cleanup	Nettoie les opérations interrompues ou annulées et déverrouille la copie de travail dans `/path`.
:resolve	Marque un fichier en conflit indiqué dans `/path` comme résolu. Si `/noquestion` est donné, alors la résolution est faite sans demander d'abord à l'utilisateur si cela doit être vraiment fait.
:repocreate	Crée un référentiel dans `/path`
:switch	Ouvre la fenêtre Aller sur. Le `/path` spécifie le répertoire cible.
:export	Exporte dans un autre répertoire la copie de travail située dans `/path`. Si le `/path` pointe vers un répertoire non versionné, une boîte de dialogue demandera une URL à laquelle exporter le dossier dans `/path`.
:merge	Ouvre la fenêtre de fusion. Le `/path` spécifie le répertoire cible. Pour fusionner une plage de révisions, les options disponibles sont les suivantes :`/fromurl:URL`, `/revrange:string`. Pour fusionner deux arborescences du référentieln les options sont les suivantes : `/fromurl:URL`, `/tourl:URL`, `/fromrev:xxx` et `/torev:xxx`. Elles pré-remplissent les champs correspondants dans la fenêtre de fusion.
:mergeall	Ouvre la fenêtre fusionner tout. Le `/path` spécifie le répertoire cible.
:copy	Ouvre la fenêtre des branches/tags. Le `/path` est la copie de travail de laquelle faire une branche ou un tag. Et le `/url` est l'URL de destination. Vous pouvez également spécifier le `/logmsg` pour passer un message prédéfini à la fenêtre des branches/tags. Ou si vous voulez passer le message en ligne de commande, utilisez `/logmsgfile:chemin`, où `chemin` pointe sur un fichier contenant le message.

Commande	Description
:settings	Ouvre la boîte de dialogue de configuration.
:remove	Supprime les fichiers dans `/path` du contrôle de version.
:rename	Renomme le fichier dans `/path`. Le nouveau nom pour le fichier est demandé par une boîte de dialogue. Pour éviter la question concernant le renommage de fichiers similaires en une étape, passez `/noquestion`.
:diff	Starts the external diff program specified in the TortoiseSVN settings. The `/path` specifies the first file. If the option `/path2` is set, then the diff program is started with those two files. If `/path2` is omitted, then the diff is done between the file in `/path` and its BASE. To explicitly set the revision numbers use `/startrev:xxx` and `/endrev:xxx`. If `/blame` is set and `/path2` is not set, then the diff is done by first blaming the files with the given revisions.
:showcompare	Depending on the URLs and revisions to compare, this either shows a unified diff (if the option `unified` is set), a dialog with a list of files that have changed or if the URLs point to files starts the diff viewer for those two files. Les options `url1`, `url2`, `revision1` et `revision2` doivent être précisées. Les options `pegrevision`, `ignoreancestry`, `blame` et `unified` sont facultatives.
:conflicteditor	Démarre l'éditeur de conflit indiqué dans la configuration de TortoiseSVN avec les fichiers corrects pour le fichier en conflit dans `/path`.
:relocate	Ouvre la boîte de dialogue Relocaliser. Le `/path` spécifie le chemin de la copie de travail à relocaliser.
:help	Ouvre le fichier d'aide.
:repostatus	Ouvre la fenêtre vérifier les modifications. Le `/path` spécifie le répertoire de la copie de travail.
:repobrowser	Ouvre la fenêtre de l'explorateur de référentiel, pointant sur l'URL de la copie de travail donnée dans `/path` ou `/path` pointe directement sur une URL. Une option supplémentaire `/rev:xxx` peut être utilisée pour spécifier la révision devant être affichée. Si le `/rev:xxx` n'est pas renseigné, la version de tête est utilisée par défaut. Si `/path` pointe sur une URL, le `/projectpropertiespath:path/to/wc` spécifie le chemin à partir duquel il faut lire et utiliser les propriétés du projet.
:ignore	Ajoute toutes les cibles dans `/path` à la liste des éléments ignorés, c'est-à-dire ajoute le `svn:ignore` à ces fichiers.
:blame	Ouvre la fenêtre de bannissement pour le fichier spécifier dans `/path`. Si les options `/startrev` et `/endrev` sont précisées, la fenêtre permettant de spécifier la plage de révisions à bannir n'est pas affichée, ces valeurs des révisions sont utilisées à la place. Si cette option est renseignée `/line:nnn`, TortoiseBlame ouvrira en affichant la nième ligne. The options `/ignoreeol`, `/ignorespaces` and `/ignoreallspaces` are also supported.
:cat	Enregistre un fichier depuis une URL ou depuis un chemin de la copie de travail donné dans `/path` à l'emplacement donné dans `/savepath:chemin`. La révision est donnée dans `/revision:xxx`. Cela peut être utilisé pour obtenir un fichier avec une révision spécifique.
:createpatch	Crée un patch pour le chemin donné dans `/path`.

Commande	Description
:revisiongraph	Montre le graphique de révision pour le chemin donné dans `/path`.
:lock	Verrouille un ou tous les fichiers dans un répertoire donné dans `/path`. La boîte de dialogue 'Verrouiller' s'affiche afin de permettre à l'utilisateur d'entrer un commentaire pour le verrou.
:unlock	Déverrouille un fichier ou tous les fichiers d'un répertoire donné dans `/path`.
:rebuildiconcache	Reconstruit le cache d'icône Windows. Utilisez-le seulement dans le cas où les icônes Windows sont corrompues. Un effet secondaire à cela (qui ne peut être évité) est que les icônes sur le bureau sont réarrangées. Pour supprimer la fenêtre d'information, passez `/noquestion`.
:properties	Affiche la boîte de dialogue de propriétés pour le chemin donné dans `/path`.

Tableau D.1. Liste des commandes et des options disponibles

Exemples (qui devraient être saisis sur une ligne):

```
TortoiseProc.exe /command:commit                /path:"c:\svn_ct\fichier1.txt*c:
                 /logmsg:"message de log de test" /closeonend

TortoiseProc.exe /command:update /path:"c:\svn_ct\" /closeonend

TortoiseProc.exe /command:log /path:"c:\svn_ct\fichier1.txt"
                 /startrev:50 /endrev:60 /closeonend
```

D.2. Commandes de TortoiseIDiff

The image diff tool has a few command line options which you can use to control how the tool is started. The program is called `TortoiseIDiff.exe`.

Le tableau suivant fait la liste des options pouvant être passées en ligne de commande à l'outil de comparaison d'images.

Option	Description
:left	Chemin du fichier affiché à gauche.
:lefttitle	Une chaine de titre. Cette chaine est utilisée dans le titre de la vue image au lieu du chemin complet du fichier image.
:right	Chemin du fichier affiché à droite.
:righttitle	Une chaine de titre. Cette chaine est utilisée dans le titre de la vue image au lieu du chemin complet du fichier image.
:overlay	If specified, the image diff tool switches to the overlay mode (alpha blend).
:fit	If specified, the image diff tool fits both images together.
:showinfo	Affiche la boîte d'informations sur l'image

Tableau D.2. Liste des options disponibles

Exemple (qui doit tenir sur une seule ligne):

```
TortoiseIDiff.exe /left:"c:\images\img1.jpg" /lefttitle:"image 1"
                  /right:"c:\images\img2.jpg" /righttitle:"image 2"
```

```
/fit /overlay
```

Annexe E. Référence croisée de l'interface en ligne de commande

Parfois ce manuel se réfère à la documentation principale de Subversion, qui décrit Subversion en termes d'Interface de Ligne de Commande (ILC). Pour vous aider à comprendre ce que fait TortoiseSVN en coulisses, nous avons compilé une liste montrant les commandes ILC équivalentes pour chacune des opérations GUI de TortoiseSVN.

> **Note**
>
> Bien qu'il y ait des équivalents ILC à ce que fait TortoiseSVN, souvenez-vous que TortoiseSVN ne fait *pas* appel à l'ILC mais utilise la bibliothèque de Subversion directement.

Si vous pensez avoir trouvé un bug dans TortoiseSVN, nous pouvons vous demander d'essayer de le reproduire en utilisant l'ILC, pour que nous puissions distinguer les problèmes de TortoiseSVN des problèmes de Subversion. Cette référence vous dit quelle commande essayer.

E.1. Conventions et règles de base

Dans les descriptions qui suivent, l'URL d'un référentiel est simplement affichée `URL` et un exemple pourrait être `http://tortoisesvn.tigris.org/svn/tortoisesvn/trunk`. Le chemin de la copie de travail est simplement affiché `PATH`, et un exemple pourrait être `C:\TortoiseSVN\trunk`.

> **Important**
>
> Parce que TortoiseSVN est une extension du shell Windows, il n'est pas capable d'utiliser la notion d'un répertoire de travail courant. Tous les chemins de la copie de travail doivent être donnés en utilisant le chemin absolu, pas le chemin relatif.

Certains éléments sont facultatifs et ceux-ci sont souvent contrôlés par des cases à cocher ou des boutons radio dans TortoiseSVN. Ces options sont affichées dans des [crochets] dans les définitions de ligne de commande.

E.2. Commandes de TortoiseSVN

E.2.1. Extraire

```
svn checkout [-N] [--ignore-externals] [-r rev] URL CHEMIN
```

Si Extraire seulement le répertoire de tête est coché, utilisez le commutateur `-N`.

Si Omettre les externes est coché, utilisez le commutateur `--ignore-externals`.

Si vous extrayez une révision spécifique, spécifiez cela après l'URL en utilisant le commutateur `-r`.

E.2.2. Mettre à jour

```
svn info URL_de_la_CdT
svn update [-r rev] CHEMIN
```

Mettre à jour plusieurs éléments n'est pas actuellement une opération atomique dans Subversion. Donc TortoiseSVN trouve d'abord la révision HEAD du référentiel et met ensuite à jour tous les éléments à ce numéro de révision particulier pour éviter de créer une copie de travail avec des révisions mélangées.

Si un seul élément est sélectionné à mettre à jour ou si les éléments choisis ne sont pas tous du même référentiel, TortoiseSVN met simplement à jour à HEAD.

Aucune option de ligne de commande n'est utilisée ici. Mettre à jour à la révision met aussi en oeuvre la commande de mise à jour, mais offre plus d'options.

E.2.3. Mettre à jour à la révision

```
svn info URL_de_la_CdT
svn update [-r rev] [-N] [--ignore-externals] CHEMIN
```

Si Mettre à jour seulement le répertoire de tête est coché, utilisez le commutateur `-N`.

Si Omettre les externes est coché, utilisez le commutateur `--ignore-externals`.

E.2.4. Livrer

Dans TortoiseSVN, la boîte de dialogue livrer utilise plusieurs commandes Subversion. La première étape est une vérification de statut qui détermine les éléments de votre copie de travail qui peuvent potentiellement être livrés. Vous pouvez passer en revue la liste, comparer les fichiers avec la BASE et les éléments que vous voulez inclure dans la livraison.

```
svn status -v CHEMIN
```

Si Afficher les fichiers non versionnés est coché, TortoiseSVN affichera aussi tous les fichiers et tous les dossiers non versionnés dans la hiérarchie de la copie de travail, en prenant en compte les règles d'exclusion. Cette fonctionnalité particulière n'a aucun équivalent direct dans Subversion, puisque la commande `svn status` ne parcourt pas les dossiers non versionnés.

Si vous sélectionnez des fichiers ou des dossiers non versionnés, ces éléments seront d'abord ajoutés à votre copie de travail.

```
svn add CHEMIN...
```

Quand vous cliquez sur OK, la livraison Subversion se produit. Si vous avez laissé toutes les cases de sélection de fichier dans leur état par défaut, TortoiseSVN utilise une seule livraison récursive de la copie de travail. Si vous désélectionnez quelques fichiers, alors une livraison non récursive (`-N`) doit être utilisée et chaque chemin doit être spécifié individuellement sur la ligne de commande de livraison.

```
svn commit -m "Commentaire" [-N] [--no-unlock] CHEMIN...
```

`Commentaire` représente ici le contenu de la boîte de saisie du commentaire. Cela peut être vide.

Si Garder les verrous est coché, utilisez le commutateur `--no-unlock`.

E.2.5. Voir les différences

```
svn diff CHEMIN
```

Si vous utilisez Voir les différences depuis le menu contextuel principal, vous comparez un fichier modifié avec sa version de BASE. La sortie de la commande de l'ILC ci-dessus fait la même chose et génère

une sortie au format unified-diff. Cependant, ce n'est pas ce qu'utilise TortoiseSVN. TortoiseSVN utilise TortoiseMerge (ou le programme de comparaison de votre choix) pour afficher les différences entre des fichiers purement texte, donc il n'y a aucun équivalent dans l'ILC.

Vous pouvez aussi comparer 2 fichiers en utilisant TortoiseSVN, qu'ils soient sous contrôle de version ou non. TortoiseSVN alimente simplement les deux fichiers dans le programme de comparaison choisi et laisse rechercher où se trouvent les différences.

E.2.6. Voir le journal

```
svn log -v -r 0:N --limit 100 [--stop-on-copy] CHEMIN
  ou
svn log -v -r M:N [--stop-on-copy] CHEMIN
```

Par défaut, TortoiseSVN essaye de récupérer 100 commentaires en utilisant la méthode --limit. Si les réglages indiquent d'utiliser de vieilles API, alors la deuxième forme est utilisée pour aller chercher les commentaires de 100 révisions du référentiel.

Si Arrêt à la copie/renommage est coché, utilisez le commutateur `--stop-on-copy`.

E.2.7. Vérifier les modifications

```
svn status -v CHEMIN
  ou
svn status -u -v CHEMIN
```

La vérification initiale du statut ne regarde que votre copie de travail. Si vous cliquez sur Vérifier le référentiel alors le référentiel est aussi vérifié pour voir quels fichiers seraient changés par une mise à jour, ce qui exige le commutateur `-u`.

Si Afficher les fichiers non versionnés est coché, TortoiseSVN affichera aussi tous les fichiers et tous les dossiers non versionnés dans la hiérarchie de la copie de travail, en prenant en compte les règles d'exclusion. Cette fonctionnalité particulière n'a aucun équivalent direct dans Subversion, puisque la commande `svn status` ne parcourt pas les dossiers non versionnés.

E.2.8. Graphique de révision

Le graphique de révision est une fonctionnalité de TortoiseSVN uniquement. Il n'y a pas d'équivalent pour le client en ligne de commande.

Ce que fait TortoiseSVN est

```
svn info URL_de_la_CdT
svn log -v URL
```

où l'URL est la *racine* du référentiel et analyse ensuite les données renvoyées.

E.2.9. Explorateur de référentiel

```
svn info URL_de_la_CdT
svn list [-r rev] -v URL
```

Vous pouvez utiliser `svn info` pour déterminer la racine du référentiel, qui est le niveau supérieur affiché dans l'explorateur de référentiel. Vous ne pouvez pas naviguer vers le `haut` au-dessus de ce niveau. Aussi, cette commande renvoie toute l'information de verrouillage affichée dans l'explorateur de référentiel.

L'appel `svn list` listera le contenu d'un répertoire, à l'URL et la révision données.

E.2.10. Éditer les conflits

Cette commande n'a aucun équivalent en ILC. Elle appelle TortoiseMerge ou un outil externe de comparaison/fusion à 3 vues pour regarder les fichiers impliqués dans le conflit et déterminer quelles lignes utiliser.

E.2.11. Résolu

```
svn resolved CHEMIN
```

E.2.12. Renommer

```
svn rename CHEMIN_COURANT NOUVEAU_CHEMIN
```

E.2.13. Supprimer

```
svn delete CHEMIN
```

E.2.14. Revenir en arrière

```
svn status -v CHEMIN
```

La première étape est un contrôle de statut qui détermine les éléments de votre copie de travail qui peuvent être potentiellement annulés. Vous pouvez examiner la liste, comparer les fichiers contre la BASE et choisir les éléments que vous voulez inclure dans le retour en arrière.

Quand vous cliquez sur OK, le retour en arrière de Subversion se produit. Si vous avez laissé toutes les cases de sélection de fichier dans leur état par défaut, TortoiseSVN utilise un seul retour en arrière récursif (`-R`) de la copie de travail. Si vous désélectionnez quelques fichiers, alors chaque chemin doit être spécifié individuellement sur la ligne de commande de retour en arrière.

```
svn revert [-R] CHEMIN...
```

E.2.15. Nettoyer

```
svn cleanup CHEMIN
```

E.2.16. Obtenir un verrou

```
svn status -v CHEMIN
```

La première étape est une vérification de statut qui détermine les fichiers de votre copie de travail qui peuvent être potentiellement verrouillés. Vous pouvez choisir les éléments que vous voulez verrouiller.

```
svn lock -m "Commentaire du verrou" [--force] CHEMIN...
```

`Commentaire du verrou` représente ici le contenu de la boîte de saisie de commentaire de verrou. Il peut être vide.

Si Voler les verrous est coché, utilisez le commutateur `--force`.

E.2.17. Relâcher un verrou

```
svn unlock CHEMIN
```

E.2.18. Branche/Etiquette

```
svn copy -m "Commentaire" URL URL
  ou
svn copy -m "Commentaire" URL@rev URL@rev
  ou
svn copy -m "Commentaire" CHEMIN URL
```

La boîte de dialogue de Branche/Étiquette exécute une copie vers le référentiel. Il y a 3 boutons radio d'options :

- Révision HEAD dans le référentiel
- Révision spécifique dans le référentiel
- Copie de travail

qui correspondent aux 3 variantes de ligne de commande ci-dessus.

`Commentaire` représente ici le contenu de la boîte de saisie du commentaire. Cela peut être vide.

E.2.19. Aller sur...

```
svn info URL_de_la_CdT
svn switch [-r rev] URL CHEMIN
```

E.2.20. Fusionner

```
svn merge [--dry-run] URL_Depuis@revN URL_Vers@revM CHEMIN
```

Le bouton Fusion de test exécute la même fusion mais avec l'option `--dry-run`.

```
svn diff URL_Depuis@revN URL_Vers@revM
```

Le Diff unifiée affiche l'opération de comparaison qui sera utilisée pour faire la fusion.

E.2.21. Exporter

```
svn export [-r rev] [--ignore-externals] URL CHEMIN_Export
```

Cette forme est utilisée lors d'un accès depuis un dossier non versionné et le dossier est utilisé comme destination.

L'exportation d'une copie de travail dans un emplacement différent est fait sans utiliser la bibliothèque de Subversion, donc il n'existe aucun équivalent en ligne de commande correspondant.

Ce que fait TortoiseSVN est une copie de tous les fichiers vers le nouvel emplacement lors de l'affichage de la progression de l'opération. Les fichiers/dossiers non versionnés peuvent être aussi exportés facultativement .

Dans les deux cas, si Omettre les externes est coché, utilisez le commutateur `--ignore-externals`.

E.2.22. Relocaliser

```
svn switch --relocate URL_Depuis URL_Vers
```

E.2.23. Créer un référentiel ici

```
svnadmin create --fs-type fsfs PATH
```

E.2.24. Ajouter

```
svn add CHEMIN...
```

Si vous avez choisi un dossier, TortoiseSVN le parcourt d'abord récursivement pour les éléments qui peuvent être ajoutés.

E.2.25. Importer

```
svn import -m Commentaire CHEMIN URL
```

`Commentaire` représente ici le contenu de la boîte de saisie du commentaire. Cela peut être vide.

E.2.26. Annoter

```
svn blame -r N:M -v CHEMIN
svn log -r N:M CHEMIN
```

Si vous utilisez TortoiseBlame pour voir les informations de bannissement, le fichier de log est également requis pour afficher les messages de log dans une info-bulle. Si vous voyez le bannissement comme un fichier texte, cette information n'est pas exigée.

E.2.27. Ajouter à la liste des ignorés

```
svn propget svn:ignore CHEMIN > FichierTemp
{éditez le nouvel élément à ignorer dans FichierTemp}
svn propset svn:ignore -F FichierTemp CHEMIN
```

Parce que la propriété `svn:ignore` est souvent composée de plusieurs lignes, elle est montrée ici comme étant modifiée via un fichier texte plutôt que directement en ligne de commande.

E.2.28. Créer un patch

```
svn diff CHEMIN > fichier_patch
```

TortoiseSVN crée un patch dans le format de différences unifiées en comparant la copie de travail avec sa version de BASE.

E.2.29. Appliquer un patch

Appliquer un patch est une activité difficile à moins que le patch et la copie de travail ne soient à la même révision. Heureusement pour vous, vous pouvez utiliser TortoiseMerge, qui n'a aucun équivalent direct dans Subversion.

Annexe F. Détails de l'implémentation

Cette annexe contient plus de détails concernant l'implémentation de quelques fonctionnalités de TortoiseSVN.

F.1. Recouvrement d'icônes

Every file and folder has a Subversion status value as reported by the Subversion library. In the command line client, these are represented by single letter codes, but in TortoiseSVN they are shown graphically using the icon overlays. Because the number of overlays is very limited, each overlay may represent one of several status values.

L'icone de recouvrement *En Conflit* est utilisée pour représenter un état `en conflit`, là ou une mise à jour génère des conflits entre la version locale et la version du référentiel. Elle est aussi utilisée pour un état `bloqué`, qui peut se produire quand une opération ne se termine pas correctement.

L'icone de recouvrement *Modifié* représente un état `modifié`, i.e. lorsque vous avez fait des modifications, l'état `fusionné` se produit lorsque les versions du référentiel ont changé et qu'elles ont été intégrées à la version locale, et l'état `remplacé` se produit lorsqu'un fichier a été supprimé et remplacé par un autre ayant le même nom mais dont le contenu est différent.

L'icone de recouvrement *Supprimé* représente un état `supprimé`, i.e. lorsqu'un élément a été marqué comme étant à supprimer, ou un état `manquant`, i.e. lorsqu'un élément n'est pas présent en local. Naturellement un élément qui manque ne peut avoir lui même d'icone de recouvrement, mais le répertoire le contenant le peut.

L'overlay indique juste qu'un fichier ou un dossier a été `ajouté` au contrôle de version.

L'icone de recouvrement *Dans Subversion* est utilisé pour représenter un élément qui est dans un état `normal`, ou un élément sous contrôle de version dont l'état n'est pas encore connu. TortoiseSVN fonctionne avec un système de mise en cache en arrière plan pour récupérer les états, les mises à jours des icones de recouvrement peuvent donc prendre quelques secondes.

L'icone de recouvrement *A Besoin d'être Verrouilé* est utilisé pour indiquer que la propriété `svn:needs-lock` est activée pour un fichier. Pour les copies de travail créées avec les versions 1.4.0 et ultérieures de Subversion, l'état de la propriété `svn:needs-lock` est mis en cache localement par Subversion et cette valeur est utilisée pour activer ou non l'icone de recouvrement. Pour les copies de travail créées avec des versions antérieures, TortoiseSVN utilise cette icone lorsque le fichier est en lecture seule. Notez que Subversion met à jour automatiquement le format des copies de travail lorsque vous faites une mise à jour, de ce fait la mise en cache de l'état de la propriété `svn:needs-lock` ne devrait pas se produite à moins que le fichier lui même ne soit pas à jour.

L'icone de recouvrement *Verrouillé* est utilisée lorsque le fichier est verrouillé dans la copie de travail.

L'icône de recouvrement *Ignoré* indique qu'un élément est dans un état ignoré, soit car il satisfait une condition globale (global pattern) soit car il satisfait une condition du dossier parent. Cette icone de recouvrement est optionnelle.

L'icône de recouvrement *non versionné* est utilisé pour représenté un élément étant dans l'état `non versionné`. C'est à dire un élément situé dans un répertoire sous contrôle de version, mais qui n'est pas lui même sous contrôle de version. Cette icone de recouvrement est optionelle.

If an item has subversion status `none` (the item is not within a working copy) then no overlay is shown. If you have chosen to disable the *Ignored* and *Unversioned* overlays then no overlay will be shown for those files either.

An item can only have one Subversion status value. For example a file could be locally modified and it could be marked for deletion at the same time. Subversion returns a single status value - in this case `deleted`. Those priorities are defined within Subversion itself.

When TortoiseSVN displays the status recursively (the default setting), each folder displays an overlay reflecting its own status and the status of all its children. In order to display a single *summary* overlay, we use the priority order shown above to determine which overlay to use, with the *Conflicted* overlay taking highest priority.

En fait, vous pouvez constater que toutes ces icônes ne sont pas utilisées sur votre système. C'est dû au fait que le nombre de recouvrements permis par Windows est limité à 15. Windows en utilise 4 et les 11 restantes peuvent être utilisés par d'autres applications. S'il n'a a pas assez d'emplacements libres, TortoiseSVN essaie d'être un « bon citoyen (TM) » et limite son utilisation des recouvrements pour laisser une chance aux autres applications.

- `Normal`, `Modifié` et `En conflit` sont toujours chargés et visibles.
- `Supprimé` est chargé si possible, mais dedevient `Modifié` s'il n'y a pas assez de connecteurs.
- `Lecture seule` est chargé si possible, mais redevient `Normal` s'il n'y a pas assez de connecteurs.
- *Verrouillé* est seulement chargé s'il y a moins de 13 recouvrements déjà chargés. Il devient *Normal* s'il n'y a pas assez d'emplacements.
- *Ajouté* est seulement chargé s'il y a moins de 14 recouvrements déjà chargés. Il devient *Modifié* s'il n'y a pas assez d'emplacements.

Annexe G. Sécuriser Svnserve grâce à SSH

Cette section fournit un guide pas-à-pas pour configurer Subversion et TortoiseSVN pour utiliser le protocole `svn+ssh`. Si vous utilisez déjà une authentification SSH pour vous connecter à votre serveur, alors vous pouvez passer cette section et trouver plus de détails dans le livre Subversion. Si vous n'utilisez pas encore SSH mais aimeriez le faire pour protéger votre installation Subversion, ce guide donne une méthode simple qui n'implique pas la création d'un compte utilisateur SSH pour chaque utilisateur subversion présent sur le serveur.

Dans cette implémentation nous créons un unique compte utilisateur SSH pour tous les utilisateurs subversion et utilisons différentes clés d'authentification pour les différencier des véritables utilisateurs Subversion.

Dans cette annexe, nous supposons que vous avez déjà les outils subversion installés et que vous avez créé un dépôt comme expliqué ailleurs dans ce manuel. Notez bien que vous ne devriez *pas* démarrer svnserve en tant que service ou démon lorsqu'il est utilisé avec SSH.

La plupart de ces informations est associé à un tutoriel fourni par Marc Logemann et qui peut être trouvé sur *www.logemann.org* [http://www.logemann.org/2007/03/13/subversion-tortoisesvn-ssh-howto/]. Des informations supplémentaires sur la configuration d'un serveur Windows sont fournies par Thorsten Müller. Merci les gars!

G.1. Configurer un Serveur Liunx

Vous devez avoir SSH activé sur votre serveur, et dans ce cas nous supposons que vous utilisez OpenSSH. Dans la plupart des distributions, il sera déjà installé. Pour le savoir, tapez:

```
ps xa | grep sshd
```

et cherchez les processus ssh.

Un détail à relever est que si vous compilez Subversion depuis le code source sans préciser de paramètres au `./configure`, Subversion créé un répertoire `bin` dabs `/usr/local` et y place les binaires. Si vous désirez utiliser le mode de tunnelage avec SSH, vous devez avoir conscience que l'utilisateur se connectant via SSH a besoin d'exécuter le programme svnserve et quelques autres binaires. Pour cette raison, placez `/usr/local/bin` dans la variable `PATH` ou créez les liens symboliques de vos binaires dans le répertoire `/usr/sbin`, ou dans tout autre répertoire communément inclus dans le `PATH`.

Vérifiez que tout est en ordre, connectez vous avec l'identifiant de l'utilisateur cible avec SSH et tapez :

```
which svnserve
```

Cette commande devrait vous indiquer si svnserve est accessible.

Créé un nouvel utilisateur que nous utiliserons pour accéder au dépôt svn:

```
useradd -m svnuser
```

Assurez-vous de donner à cette utilisateur tous les droits d'accès au dépôt.

G.2. Configurer un Serveur Windows

Install Cygwin SSH daemon as described here: *http://pigtail.net/LRP/printsrv/cygwin-sshd.html*

Create a new Windows user account `svnuser` which we will use to access the repository. Be sure to give this user full access rights to the repository.

If there is no password file yet then create one from the Cygwin console using:

```
mkpasswd -l > /etc/passwd
```

G.3. Client SSH à utiliser avec TortoiseSVN

Grab the tools we need for using SSH on the windows client from this site: *http://www.chiark.greenend.org.uk/~sgtatham/putty/* Just go to the download section and get `Putty`, `Plink`, `Pageant` and `Puttygen`.

G.4. Création des certificats OpenSSH

The next step is to create a key pair for authentication. There are two possible ways to create keys. The first is to create the keys with PuTTYgen on the client, upload the public key to your server and use the private key with PuTTY. The other is to create the key pair with the OpenSSH tool ssh-keygen, download the private key to your client and convert the private key to a PuTTY-style private key.

G.4.1. Créer des clés en utilisant ssh-keygen

Login to the server as `root` or `svnuser` and type:

```
ssh-keygen -b 1024 -t dsa -N passphrase -f keyfile
```

substituting a real pass-phrase (which only you know) and key file. We just created a SSH2 DSA key with 1024 bit key-phrase. If you type

```
ls -l keyfile*
```

you will see two files, `keyfile` and `keyfile.pub`. As you might guess, the `.pub` file is the public key file, the other is the private one.

Append the public key to those in the `.ssh` folder within the `svnuser` home directory:

```
cat keyfile.pub >> /home/svnuser/.ssh/authorized_keys
```

In order to use the private key we generated, we have to convert it to a putty format. This is because the private key file format is not specified by a standards body. After you download the private key file to your client PC, start PuTTYgen and use Conversions → Import key. Browse to your file `keyfile` which you got from the server the passphrase you used when creating the key. Finally click on Save private key and save the file as `keyfile.PPK`.

G.4.2. Créer des clés en utilisant PuTTYgen

Utilisez PuTTYgen afin de générer une paire de clé public/privée, et sauvegardez les. Copiez la clé public sur le serveur et ajoutez là à celles déjà présentent dans le dossier `.ssh`, sous le dossier dossier personnel `svnuser`:

```
cat keyfile.pub >> /home/svnuser/.ssh/authorized_keys
```

G.5. Tester en utilisant PuTTY

Afin de tester la connectivité, nous utiliserons PuTTY. Démarrez le programme et à partir de l'onglet Session, spécifiez comme nom d'hôte le nom ou l'adresse de votre serveur, SSH comme protocole,

et sauvegardez la session en tant que SvnConnection ou tout autre nom qui vous convient. À partir de l'onglet SSH, choisir SSH version 2 comme protocole par défaut. À partir de l'onglet Auth, saisir le chemin complet vers le fichier de clé privé .PPK que vous avez convertis précédemment. Retournez à l'onglet Sessions et cliquez sur le bouton Sauvegarde. Vous devriez maintenant voir SvnConnection dans la liste de vos sessions sauvegardées.

Cliquez sur Ouvrir et vous devriez voir une invite au format telnet. Utilisez svnuser comme nom d'utilisateur et si tout ce passe comme prévus, vous devriez être authentifié sans qu'un mot de passe ne vous soit demandé.

Vous pouvez avoir à éditer /etc/sshd_config sur le serveur. Effectuer l'édition dans le fichier en accord avec ce qui suit et redémarer le service SSH par la suite.

```
PubkeyAuthentication yes
PasswordAuthentication no
PermitEmptyPasswords no
ChallengeResponseAuthentication no
```

G.6. Tester SSH avec TortoiseSVN

Jusqu'à maintenant, nous avons seulement vérifié que l'authentification fonctionne sous SSH. Maintenant, nous devons nous assurer que notre connexion SSH nous permet d'exécuter svnserve. Sur le serveur, modifier /home/svnuser/.ssh/authorized_keys de la façon suivante afin de permettre à plusieurs auteurs subversion d'utiliser le même compte, svnuser. Veuillez noter que tous les auteurs subversion utilisent le même identifiant utilisateur, mais une clé d'authentification différente. Vous devez ainsi ajouter une ligne pour chaque auteur.

Note: Nous avons ici une seule ligne très longue et non coupée.

```
command="svnserve -t -r <ReposRootPath> --tunnel-user=<author>",
           no-port-forwarding,no-agent-forwarding,no-X11-forwarding,
           no-pty ssh-rsa <PublicKey> <Comment>
```

Plusieurs paramètres sont à changer, dépendamment de votre configuration.

<ReposRootPath> should be replaced with the path to the directory containing your repositories. This avoids the need to specify full server paths within URLs. Note that you must use forward slashes even on a Windows server, e.g. c:/svn/reposroot. In the examples below we assume that you have a repository folder within the repository root called repos.

<author> doit être remplacé par l'auteur svn associé à l'envoie au dépôt. Cela permet également à svnserve d'utiliser ses propres droits d'accès au fichier svnserve.conf.

<PublicKey> devrait être remplacé avec la clé publique que vous avez généré plus tôt.

<Comment> peut être n'importe quel commentaire de votre choix, mais il est utile pour faire correspondre le nom d'un auteur svn avec son vrai nom.

Effectuez un clic droit de la souris sur n'importe quel dossier de l'explorateur Windows et choisissez TortoiseSVN → Explorateur de référentiel. Vous serez invités à saisir une URL, utilisez le format suivant:

```
svn+ssh://svnuser@SvnConnection/repos
```

Que signifie cette URL? Le nom du schéma est svn+ssh et permet d'indiquer à TortoiseSVN comment traiter les requêtes au serveur. Après le double slash, vous indiquez à l'utilisateur de s'authentifier au

serveur, dans notre cas il s'agit de `svnuser`. Le nom de la session PuTTY doit être fournis après le `@`. Le nom de cette session contient tout les détails, comme où trouver la clé privée, l'adresse IP de notre serveur, ou le DNS. Finalement, nous devons fournir le chemin d'accès vers le référentiel, relatif au référentiel racine sur le serveur, tel que spécifié dans le fichier `authorized_keys`.

Cliquez sur le bouton OK et vous devriez par la suite être en mesure de naviguer dans le contenu du référentiel. Vous avez maintenant un tunnel SSH établis avec TortoiseSVN.

Notez que par défaut, TortoiseSVN utilise sa propre version de Plink pour établir la connexion. Cette stratégie permet d'empêcher l'affichage d'une fenêtre à chaque tentative d'authentification. Cependant, cela à pour conséquence que lorsqu'une erreur se produit, elle ne peut pas être affichée. Si vous recevez l'erreur « Impossible d'écrire sur la sortie standard », vous pouvez essayez de configurer Plink comme le client sous les paramètres réseaux de TortoiseSVN. Cela vous permettra de voir la vrai erreur générée par Plink.

G.7. Variantes de Configuration SSH

Une manière de simplifier l'URL dans TortoiseSVN est de définir l'utilisateur à l'intérieur de la session PuTTY. Pour cela vous devez charger votre session prédéfinie `ConnexionSVN` dans PuTTY et dans l'onglet Connexion définir l' utilisateur de la connexion automatique sur le nom d'utilisateur, par exemple `svnuser`. Sauvegardez votre session PuTTY comme précédemment et essayez l'URL suivante dans TortoiseSVN:

```
svn+ssh://ConnexionSvn/repos
```

Cette fois-ci nous fournissons uniquement la `ConnexionSVN` de la session PuTTY au client SSH que TortoiseSVN utilise (TortoisePlink.exe). Ce client va contrôler tous les détails nécessaires de la session.

Au moment d'écrire PuTTY ne vérifie pas toutes les configurations sauvegardées, alors si vous avez de multiples configurations avec le même nom de serveur, PuTTY prendra la première qui correpond. De même, si vous éditez la configuration par défaut et que vous la sauvegardez, le nom d'utilisateur de la connexion automatique n'est *pas* sauvegardé.

Beaucoup de gens aiment à utiliser Pageant pour stocker toutes leurs clés. Comme il est possible de stocker une clé dans une session PuTTY, vous n'avez pas toujours besoin de Pageant. Cependant, imaginez que vous vouliez stocker des clés pour plusieurs serveurs différents; dans ce cas il vous faudra éditer la session PuTTY encore et encore, en fonction du serveur avec lequel vous essayez de vous connecter. Dans cette situation Pageant prend tout son sens, car PuTTY, Plink, TortoisePlink ou n'importe quel outil basé sur PuTTY, lorsqu'il essaie de se connecter à un serveur SSH, vérifie toutes les clés privées que Pageant détient pour initier la connexion.

Pour cette tâche, exécutez simplement Pageant et ajoutez la clé privée. Celle-ci devrait être la même clé privée que vous avez définie dans la session PuTTY au-dessus. Si vous utilisez Pageant pour le stockage de clés privées, vous pouvez effacer la référence au fichier de la clé privée dans votre session PuTTY sauvegardée. Vous pouvez ajouter plus de clés pour d'autres serveurs, ou bien entendu d'autres utilisateurs.

Si vous ne voulez pas répéter cette manipulation après chaque redémarrage du client,, vous devez mettre Pageant dans la liste des programmes à exécuter au démarrage de Windows. Vous pouvez ajouter les clés avec leurs chemin complet comme arguments de Pageant.exe

La dernière façon de se connecter à un serveur SSH est tout simplement d'utiliser cette URL dans TortoiseSVN:

```
svn+ssh://svnuser@100.101.102.103/referentiel
svn+ssh://svnuser@mondomaine.com/referentiel
```

Comme vous pouvez le voir, nous n'utilisons pas une session PuTTY mais une adresse IP (ou un nom de domaine) comme cible. L'utilisateur est également stipulé, cela dit vous pourriez vous demander comment est trouvée la clé privée. Parce que TortoisePlink.exe n'est qu'une version modifiée de Plink, l'outil standard de la suite PuTTY, TortoiseSVN essaiera toutes les clés stockées dans Pageant.

Si vous utilisez cette dernière méthode, assurez-vous de ne pas avoir un nom d'utilisateur par défaut configuré dans PuTTY. Nous avons reçu pour ce cas de figure des signalements de bogues entraînant des coupures de connexion dans PuTTY. Pour supprimer l'utilisateur par défaut, effacez simplement `HKEY_CURRENT_USER\Software\SimonTatham\Putty\Sessions\Default%20Settings\HostName`

Glossaire

Ajouter	Une commande Subversion utilisée pour ajouter un fichier ou un répertoire à votre copie de travail. Les nouveaux éléments sont ajoutés au référentiel à la livraison.
Aller sur...	De même que « Mettre à jour à la révision » change la fenêtre de temps d'une copie de travail pour regarder un point différent dans l'histoire, « Aller sur... » modifie la fenêtre d'espace d'une copie de travail pour qu'elle pointe vers un endroit différent du référentiel. C'est particulièrement utile quand vous travaillez sur le tronc et les branches où seuls quelques fichiers diffèrent. Vous pouvez alors commuter votre copie de travail entre les deux et seuls les fichiers modifiés seront transférés.
Annoter	Cette commande n'est utilisée que pour les fichiers texte et elle annote chaque ligne pour montrer la révision du référentiel à laquelle elle a été changée et l'auteur du changement. Notre implémentation graphique s'appelle TortoiseBlame et il montre aussi la date de dernière livraison et son commentaire au survol du numéro de révision avec la souris.
BDB	Berkeley DB. Une base de données bien testée pour les référentiels, qui ne peut pas être utilisée sur les partages réseaux. Par défaut pour les référentiels pré 1.2.
Branche	Un terme fréquemment utilisé dans les système de contrôle de révisions pour décrire ce qu'il se passe quand le développement se scinde à un point particulier et suit 2 chemins séparés. Vous pouvez créer une branche en dehors de la ligne de développement principale pour développer une nouvelle fonctionnalité sans rendre la ligne principale instable. Ou vous pouvez faire une branche avec une version stable sur laquelle vous ne ferez que des corrections de bugs, tandis que les nouveaux développements seront faits sur la branche instable. Dans Subversion, une branche est implémentée comme une « copie bon marché ».
Conflit	Quand les changements du référentiel sont fusionnés avec les changements locaux, ces changements se produisent parfois sur les mêmes lignes. Dans ce cas, Subversion ne peut pas décider automatiquement quelle version utilisée et le fichier est dit en conflit. Vous devez éditer le fichier manuellement et résoudre le conflit avant de pouvoir livrer d'autres modifications.
Copie de travail	C'est votre « bac à sable » local, l'endroit où vous travaillez sur les fichiers versionnés et il réside normalement sur votre disque dur local. Vous créez une copie de travail en faisant une « Extraction » du référentiel et vous remettez vos modifications dans le référentiel en faisant une « Livraison ».
Copier	Dans un référentiel Subversion, vous pouvez créer une copie d'un unique fichier ou de toute une arborescence. Celles-ci sont implémentées comme des « copies bon marché » qui fonctionnent comme des liens vers l'original dans le sens où elles ne prennent quasiment pas de place. Faire une copie préserve l'historique de l'élément dans la copie, donc vous pouvez suivre les changements effectués avant que la copie n'ait été faite.

Exporter
: Cette commande crée une copie d'un répertoire versionné, comme une copie de travail, mais sans les répertoires locaux `.svn`.

Extraire
: Une commande Subversion qui crée une copie de travail locale dans un répertoire vide en téléchargeant les fichiers versionnés depuis le référentiel.

FSFS
: Un système de gestion de fichier de Subversion pour référentiels. Peut être utilisé pour les partages réseaux. Utilisé par défaut pour les référentiels à partir de la version 1.2.

Fusionner
: Le procédé par lequel les modifications du référentiel sont ajoutées dans votre copie de travail sans pertuber les changements que vous avez déjà faits localement. Parfois ces changements ne peuvent pas être réconciliés automatiquement et la copie de travail est dite en conflit.

  La fusion se produit automatiquement lors de la mise à jour de votre copie de travail. Vous pouvez aussi fusionner des changements spécifiques depuis une autre branche en utilisant la commande Fusionner de TortoiseSVN.

GPO
: Group policy object (Stratégie de groupe)

Historique
: Afficher l'historique des révisions d'un fichier ou d'un répertoire. Aussi appelé le « Journal ».

Importer
: La commande Subversion pour importer une hiérarchie de dossiers complète dans le référentiel en une seule révision.

Journal
: Afficher l'historique des révision d'un fichier ou d'un répertoire. Aussi connu comme l'« Historique ».

Livrer
: Cette commande Subversion est utilisée pour renvoyer les changements de votre copie de travail locale au référentiel, en créant une nouvelle révision du référentiel.

Mettre à jour
: Cette commande Subversion récupère les derniers changements depuis le référentiel vers votre copie de travail, en fusionnant les modifications faites par d'autres avec les modifications locales dans la copie de travail.

Nettoyer
: Pour citer le manuel Subversion : « Nettoie récursivement la copie de travail, en supprimant les verrous et en reprenant les opérations non terminées. Si vous obtenez une erreur `copie de travail verrouillée`, exécutez cette commande pour supprimer les verrous périmés et rendre votre copie de travail utilisable à nouveau. » Notez que dans ce contexte, les « verrous » font référence aux verrouillages du système de fichiers local et non au verrouillage du référentiel.

Patch
: Si la copie de travail n'a que des modifications sur des fichiers texte, il est possible d'utiliser la commande Subversion Voir les différences pour générer un unique fichier résumant ces changements dans le format de différences unifiées. Un fichier de ce type est souvent connu comme un « Patch » et il peut envoyé par email à quelqu'un d'autre (ou à une mailing list) et appliqué à une autre copie de travail. Une personne sans un accès pour livrer peut effectuer ces changements et soumettre un fichier patch à une personne autorisée à livrer pour qu'elle l'applique. Ou si vous n'êtes pas sûr d'un

changement, vous pouvez soumettre un patch aux autres pour qu'ils l'examinent.

Propriété

En plus de versionner vos répertoires et vos fichiers, Subversion vous permet d'ajouter des métadonnées versionnées - connues comme des « propriétés » - à chacun de vos fichiers et répertoires versionnés. Chaque propriété possède un nom et une valeur, plutôt qu'une clé de régistre. Subversion dispose de quelques propriétés spéciales qu'il utilise en interne, comme `svn:eol-style`. TortoiseSVN en a aussi, tel que `tsvn:logminsize`. Vous pouvez ajouter vos propres propriétés avec des noms et des valeurs de votre choix.

Propriété de révision (revprop)

Comme les fichiers peuvent avoir des propriétés, chaque révision du référentiel le peut aussi. Quelques revprops spéciales sont ajoutées automatiquement quand la révision est créée, à savoir : `svn:date svn:author svn:log` qui représentent respectivement la date de livraison, le livreur et le commentaire. Ces propriétés peuvent être édités mais elles ne sont pas versionnées, donc les changements sont permanents et ne peuvent pas être annulés.

Référentiel

Un référentiel est un endroit central où sont stockées et entretenues les données. Un référentiel peut être un endroit où se trouvent plusieurs bases de données ou des fichiers pour la distribution sur un réseau, ou un référentiel peut être un emplacement directement accessible à l'utilisateur sans devoir traverser de réseaux.

Relocaliser

Si votre référentiel bouge, peut-être parce que vous l'avez déplacé dans un autre répertoire de votre serveur, ou le nom de domaine du serveur a changé, vous devez « relocaliser » votre copie de travail pour que ces URLs du référentiel pointent vers le nouvel emplacement.

Note : vous ne devriez utiliser cette commande que si votre copie de travail fait référence au même emplacement dans le même référentiel, mais le référentiel a lui-même bougé. Dans d'autres circonstances, vous avez probablement besoin de la commande « Aller sur... » à la place.

Résoudre

Quand les fichiers d'une copie de travail sont laissés dans un état conflictuel suivant une fusion, ces conflits doivent être traités par une personne en utilisant un éditeur (ou peut-être TortoiseMerge). Ce procédé est connu comme « Résoudre les conflits ». Quand cela est fait, vous pouvez marqués les fichiers en conflit comme étant résolus, ce qui vous permet de les livrer.

Revenir en arrière

Subversion conserve une copie « primitive » locale de chaque fichier tel qu'il était lors de la dernière mise à jour de votre copie de travail. Si vous avez des changements et que vous décidez de les annuler, vous pouvez utiliser la commande « Revenir en arrière » pour revenir à la copie primitive.

Révision

Chaque fois que vous livrez un jeu de modifications, vous créez une nouvelle « révision » dans le référentiel. Chaque révision représente l'état de l'arborescence du référentiel à un certain point de son histoire. Si vous voulez revenir dans le temps, vous pouvez examiner le référentiel tel qu'il était à la révision N.

Dans un autre sens, une révision peut faire référence à un jeu de modifications qui ont été faites quand la révision a été créée.

Revision BASE	La révision de base courante d'un fichier ou d'un répertoire dans votre *copie de travail*. C'est la révision à laquelle se trouvait le fichier ou le répertoire, à la dernière extraction, mise à jour ou livraison. La révision BASE est normalement différente de la révision HEAD.
Révision HEAD	La dernière révision d'un fichier ou d'un répertoire dans le *référentiel*.
Supprimer	Quand vous supprimez un élément versionné (et que vous livrez le changement), l'élément n'existe plus dans le référentiel après la révision livrée. Mais il existe bien sûr toujours dans les révisions précédentes du référentiel, donc vous pouvez toujours y avoir accès. Si besoin, vous pouvez copier un élément supprimé et le « réssusciter » complètement avec son historique.
SVN	Une abbréviation pour Subversion fréquemment utilisée. Le nom du protocole personnalisé de Subversion utilisé par le serveur de référentiel « svnserve ».
Verrouiller	Quand vous retirez un verrou sur un élément versionné, vous le marquez comme non livrable dans le référentiel, sauf dans la copie de travail où il a été déverrouillé.
Voir les différences	Très utile quand vous voulez voir exactement quels changements ont été faits.

Index

Symboles

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

V

W