TortoiseSVN

Un client Subversion pour Windows

Version 1.14

Stefan Küng
Lübbe Onken
Simon Large

TortoiseSVN: Un client Subversion pour Windows: Version 1.14

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

Date de publication 2022/09/06 19:53:28 (r29447)

Table des matières

Liste des illustrations

Liste des tableaux

Avant-propos

Le contrôle de versions est l'art de gérer les changements d'information. Cela a longtemps été un outil critique pour les programmeurs, qui passent typiquement leur temps à faire de petites modifications à leurs logiciels et à défaire ces changements le lendemain. Imaginez une équipe de ces programmeurs travaillant en parallèle - et peut-être en même temps sur les même fichiers ! - et vous comprenez pourquoi un bon système est nécessaire pour *gérer le chaos potentiel*.

1. Qu'est-ce que TortoiseSVN ?

TortoiseSVN est un client Windows open-source gratuit pour le système de contrôle de version *Apache™ Subversion®*. TortoiseSVN gère des fichiers et répertoires dans le temps. Les fichiers sont stockés dans un *dépôt* central. Le dépôt est un peu comme un serveur de fichiers ordinaire, sauf qu'il se rappelle chaque changement fait à vos fichiers et répertoires. Cela vous permet de récupérer les anciennes versions de vos fichiers et examiner l'histoire de comment et quand vos données ont changé, et qui les a changé. C'est pourquoi beaucoup de gens voient Subversion et des systèmes de contrôle de versions en général, comme d'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.

2. Les fonctionnalités de TortoiseSVN

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

Intégration dans le shell

TortoiseSVN s'intègre complètement dans le shell Windows (c'est-à-dire l'explorateur). Ce qui signifie que vous pouvez continuer à travailler avec les outils qui vous sont familiers. Ainsi, 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 quand même garder en tête que TortoiseSVN est intentionnellement développé comme une extension pour l'explorateur Windows. Il est donc possible que l'intégration ne soit pas aussi complète dans d'autres applications, par exemple que les icônes superposées ne soient pas affichées.

Icônes superposées

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

Interface Utilisateur

Lorsque vous listez les changements d'un fichier ou d'un répertoire, vous pouvez cliquer sur une révision pour en voir le commentaire. Vous pouvez également voir une liste des fichiers modifiés - faites simplement un double clic sur un fichier pour voir ce qui a été changé.

La boîte de dialogue Livrer liste tous les items qui seront envoyés dans la livraison. Chaque item a une case à cocher, ainsi vous pouvez sélectionner ceux que vous voulez inclure dans la livraison. Les fichiers non versionnés peuvent aussi être listés, au cas où vous oublieriez d'ajouter ces fichiers.

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'historique 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 dépôt 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 dépôt, 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 dépôt, 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 dépôt, 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 dépôt.

3. Licence

TortoiseSVN est un projet Open Source développé sous la licence GNU General Public (GPL). Il est gratuit pour le téléchargement et l'utilisation, soit personnellement soit commercialement, sur n'importe quel nombre de PC.

Bien que la plupart des utilisateurs téléchargent l'installateur, vous avez également accès à l'intégralité du code source du logiciel. Vous pouvez y naviguer en suivant le lienhttps://osdn.net/projects/tortoisesvn/scm/svn/. La version actuelle de développement se trouve sous `/trunk/` et les versions publiées sous `/tags/`.

4. Développement

TortoiseSVN et Subversion sont tout deux développés par une communauté de personnes. Elles viennent de pays du monde entier, unissant leur travail pour créer de merveilleux logiciels.

4.1. 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 intégré au shell de Windows a été inspirée

par le client semblable pour CVS nommé TortoiseCVS. Tim étudia le code source de TortoiseCVS et l'utilisa comme base pour TortoiseSVN. Il commença alors le projet, enregistra le domaine `tortoisesvn.org` et mit le code source en ligne.

Pendant ce temps, Stefan Küng cherchait un bon système de contrôle de version gratuit et trouva Subversion et le source de TortoiseSVN. Puisque TortoiseSVN n'était toujours pas prêt à l'emploi, il a 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.

Au fur et à mesure que Subversion devenait plus stable, il a attiré de plus en plus d'utilisateurs qui ont aussi commencé à utiliser TortoiseSVN comme client Subversion. La base utilisateurs 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. Il s'occupe maintenant du site Web et gère les nombreuses traductions.

Au fil du temps, d'autres systèmes de contrôle de versions ont eu leur propre client Tortoise, ce qui a posé un problème avec l'Explorer : Le nombre de ces icônes superposées est limité, et même un seul client Tortoise peut facilement dépasser cette limite. C'est pourquoi Stefan Küng a implémenté le composant TortoiseOverlays, qui permet à tous les clients Tortoise d'utiliser les mêmes icônes superposées. Aujourd'hui tous les client Tortoise open source et même quelques clients en dehors de Tortoise partagent ce composant.

4.2. Remerciements

Tim Kemp
: pour avoir démarré le projet TortoiseSVN

Stefan Küng
: pour le dur travail pour avoir fait de TortoiseSVN ce qu'il est maintenant, et sa gestion du projet

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

Simon Large
: pour la maintenance de la documentation

Stefan Fuhrmann
: pour le cache des commentaires et le graphe des révisions

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 envoyée

5. Guide de lecture

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 et préfèrent une interface graphique. 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.

Cet Avant-propos explique ce qu'est TortoiseSVN, parle un peu du projet TortoiseSVN, de la communauté qui travaille dessus, et sur les conditions d'utilisation et de distribution.

Le Chapitre 1, *Pour commencer* explique comment installer TortoiseSVN sur votre ordinateur, et comment commencer à l'utiliser immédiatement.

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 3, *Le Dépôt* explique comment installer un dépôt local, ce qui est utile pour tester Subversion et TortoiseSVN en utilisant un seul PC. Il explique aussi brièvement l'administration de dépôt, ce qui s'applique également aux dépôts localisés sur un serveur.

Le Chapitre 4, *Guide d'utilisation quotidienne* est la section 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. Elle passe ensuite à des sujets plus avancés.

Le Chapitre 5, *Moniteur de projet* explique comment vous pouvez mettre en place un suivi de vos projets Subversion pour ne pas manquer de livraisons importantes de la part des autres membres de votre équipe.

Chapitre 6, *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.

L'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`.

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 de 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 : Appuyez sur OK pour continuer.

Les actions utilisateur sont mises en évidence en gras. **Alt+A** : appuyez sur la touche **Alt** de votre clavier et appuyez en même temps sur la touche **A**. Glisser avec le bouton droit : appuyez sur le bouton droit de la souris et, en gardant le bouton appuyé, faites *glisser* les éléments à leur 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. Pour commencer

Cette section s'adresse aux personnes qui souhaitent savoir à quoi sert TortoiseSVN et donne un exemple d'utilisation. Elle explique comment installer TortoiseSVN, paramétrer un dépôt et vous explique les opérations les plus courantes.

1.1. Installer TortoiseSVN

1.1.1. Configuration requise

TortoiseSVN fonctionne sous Windows Vista ou ultérieur et est disponible aussi bien en 32-bit qu'en 64-bit. Le programme d'installation pour la version 64-bit de Windows comprend également les extensions 32-bit. Cela veut dire que vous n'avez pas besoin d'installer la version 32-bit en plus pour avoir le menu contextuel de TortoiseSVN et les icônes superposées dans les applications 32-bit.

Le support pour Windows 98, Windows ME et Windows NT4 a été arrêté à la version 1.2.0, le support pour Windows 2000 et XP jusqu'au SP2 a été arrêté à la version 1.7.0. Le support pour Windows XP SP3 a été arrêté à la version 1.9.0. Vous pouvez encore téléchargé et installer des versions ultérieures au besoin.

1.1.2. 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. N'oubliez pas de redémarrer après l'installation.

Important

L'installation de TortoiseSVN nécessite des privilèges administrateur. L'installateur vous demandera un mot de passe administrateur si nécessaire.

Les packs de langue sont disponibles pour traduire l'interface utilisateur de TortoiseSVN dans plusieurs langues. Veuillez vérifier Annexe G, *Paquetages linguistiques et correcteurs orthographiques* pour davantage d'information sur la procédure d'installation des packs.

Si vous rencontrez des problèmes pendant ou après l'installation de TortoiseSVN, veuillez vous référer à notre FAQ en ligne à l'adresse https://tortoisesvn.net/faq.html.

1.2. Concepts de base

Avant de se mettre au travail avec de vrais fichiers, il est important d'avoir une vue d'ensemble de la façon dont Subversion fonctionne et des termes qui sont utilisés.

Le Dépôt
: Subversion utilise une base de données centrale qui contient tous vos fichiers sous contrôle de version avec leur historique complet. Cette base de données est appelée le *dépôt*. Le dépôt tourne normalement sur un serveur de fichiers exécutant le programme Subversion, qui fournit le contenu aux clients Subversion (comme TortoiseSVN) sur demande. Si vous ne sauvegardez qu'une chose, sauvegardez votre dépôt car il est la copie principale de toutes vos données.

Copie de travail
: C'est là que vous commencez le vrai travail. Chaque développeur a sa propre copie de travail, parfois appelé un bac à sable, sur son PC local. Vous pouvez récupérer la dernière version à partir du dépôt, y travailler localement sans affecter les versions des autres développeurs, puis, une fois que vous êtes satisfait de vos modifications, les livrer vers le dépôt.

  Une copie de travail Subversion ne contient pas l'historique du projet, mais il conserve une copie des fichiers tels qu'ils existent dans le dépôt avant que vous ne commenciez à faire des changements. Cela signifie qu'il est facile de vérifier exactement quels changements vous avez faits.

Vous devez également savoir où trouver TortoiseSVN parce qu'il n'y a pas beaucoup à voir dans le menu Démarrer. C'est parce que TortoiseSVN est une extension du shell, alors tout d'abord, démarrez l'explorateur Windows. Faites un clic droit sur un dossier dans l'explorateur et vous devriez voir quelques nouvelles entrées dans le menu contextuel comme ceci :

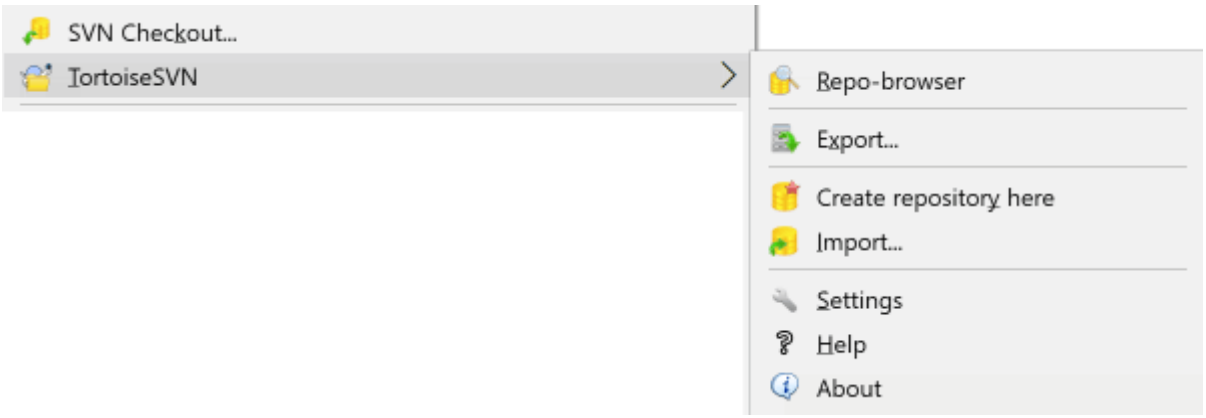


Figure 1.1. Le menu TortoiseSVN pour les dossiers non versionnés

1.3. Faites un essai

Cette section vous montre comment tester certaines des fonctionnalités les plus couramment utilisés sur un dépôt de test de petite taille. Naturellement, elle n'explique pas tout — ce n'est que le guide de démarrage rapide, après tout. Une fois que vous aurez acquis les bases, vous devez prendre le temps de lire le reste de ce guide, qui vous explique les choses beaucoup plus en détail. Il donne également davantage d'explications sur la manière de bien configurer un serveur Subversion.

1.3.1. Créer un dépôt

Pour un projet réel, vous utiliserez un dépôt mis en place dans un endroit sûr et un serveur Subversion pour le contrôler. Dans ce tutoriel, nous allons utiliser les fonctionnalités du dépôt local de Subversion qui permet un accès direct à un dépôt créé sur votre disque dur sans avoir besoin d'un serveur.

Tout d'abord, créez un nouveau dossier vide sur votre PC. Il peut être placé n'importe où, mais dans ce tutoriel, nous appelons l'appeler `C:\svn_repos`. Faites maintenant un clic droit sur le dossier et choisir dans le menu contextuel TortoiseSVN → Créer un dépôt ici.... Le dépôt est alors créé à l'intérieur du dossier, prêt à l'usage. Nous allons aussi créer la structure interne par défaut du dossier en cliquant sur le bouton Créer la structure du dossier.

Important

La fonctionnalité de dépôt local est très utile pour des tests et évaluations, mais à moins que vous ne travailliez comme seul développeur sur un seul PC, vous devriez toujours utiliser un serveur Subversion adéquat. Il est tentant dans une petite entreprise d'éviter le travail de mise en place d'un serveur et d'accéder simplement à votre dépôt sur un partage réseau. Ne faites jamais cela. Vous perdrez des données. Lisez Section 3.1.4, « Accéder à un dépôt situé dans un partage réseau » pour savoir pourquoi cela est une mauvaise idée, et comment configurer un serveur.

1.3.2. Importer un projet

Nous avons à présent un référentiel, mais il est complètement vide pour le moment. Supposons que j'aie un ensemble de fichiers dans `C:\Projets\Widget1` que je voudrais ajouter. Accédez au répertoire `Widget1` dans l'explorateur et faites un clic droit dessus. Maintenant, sélectionnez TortoiseSVN → Importer... qui ouvre une boîte de dialogue.

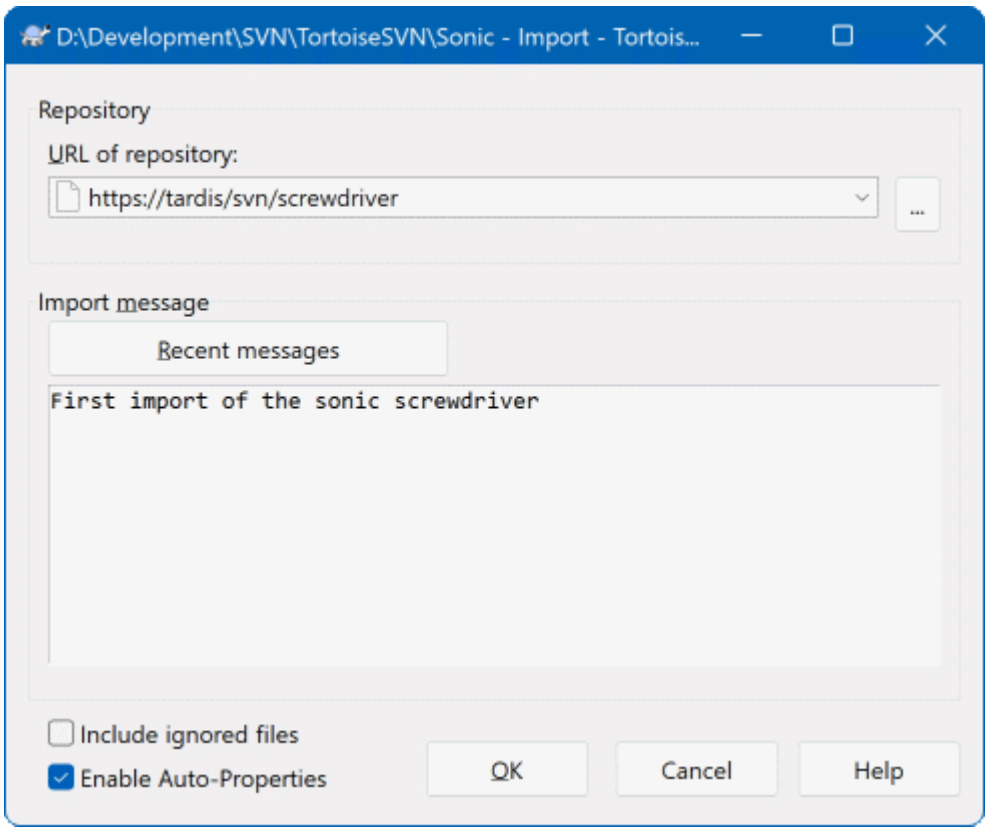


Figure 1.2. La boîte de dialogue Importer

Un dépôt Subversion est référencé par une URL, ce qui permet de le spécifier de manière univoque partout sur Internet. Dans notre exemple, nous avons besoin de pointer sur notre propre dépôt local qui a une URL du type `file:///c:/svn_repos/trunk`, auquel nous allons ajouter notre nom de projet `Widget1`. Notez qu'il y a 3 barres obliques après `file:` et que ce sont des barres obliques (et non inverses) qui sont utilisées ensuite.

L'autre caractéristique importante de cette boîte de dialogue est la partie Commentaire de l'import qui vous permet d'écrire un message décrivant ce que vous faites. Quand vous regardez votre historique de projet, ces commentaires de livraison sont un guide précieux pour savoir quelles modifications ont été faites et pourquoi. Dans notre exemple, on peut écrire quelque chose de simple comme « Importation du projet Widget1 ». Cliquez sur OK et le dossier est ajouté à votre dépôt.

1.3.3. Extraire une copie de travail

Maintenant que nous disposons d'un projet dans notre dépôt, nous devons créer une copie de travail à utiliser pour le travail quotidien. Notez que l'action d'importer un dossier ne le convertit pas automatiquement en une copie de travail. Le terme propre à Subversion pour créer une nouvelle copie de travail est `Extraire`. Nous allons faire une extraction du dossier Widget1 de notre dépôt dans un dossier de développement sur le PC appelé `C:\Projects\Widget1-Dev`. Créez ce dossier, puis faites un clic droit dessus et choisissez TortoiseSVN → Extraire.... Entrez ensuite l'URL sur laquelle faire l'extraction, dans notre cas `file:///c:/svn_repos/trunk/Widget1` et cliquez sur OK. Notre dossier de développement est alors rempli avec les fichiers du dépôt.

Important

Dans les réglages par défaut, la commande Extraire n'est pas située dans le sous-menu TortoiseSVN mais est affichée directement dans le menu contextuel de l'explorateur. Les commandes de TortoiseSVN qui ne sont pas dans le sous-menu ont le terme SVN préfixé: SVN Extraire...

Vous remarquerez que l'apparence de ce dossier est différente de celle de notre dossier d'origine. Chaque fichier a une coche verte dans le coin en bas à gauche. Ce sont des icônes d'état TortoiseSVN qui ne sont présentes que dans une copie de travail. L'état vert indique que le fichier est inchangé par rapport à la version dans le dépôt.

1.3.4. Faire des modifications

Il est temps de se mettre au travail. Dans le répertoire `Widget1-Dev`, nous commençons à éditer les fichiers — disons que nous apportons des modifications à `Widget1.c` et `ReadMe.txt`. Notez que les icônes superposées sur ces fichiers sont passées au rouge, ce qui indique que des modifications ont été apportées en local.

Mais quels sont les changements ? Faites un clic droit sur l'un des fichiers modifiés et sélectionnez TortoiseSVN → Voir les différences. L'outil de comparaison de fichier de TortoiseSVN démarre, vous montrant exactement quelles lignes ont été modifiées.

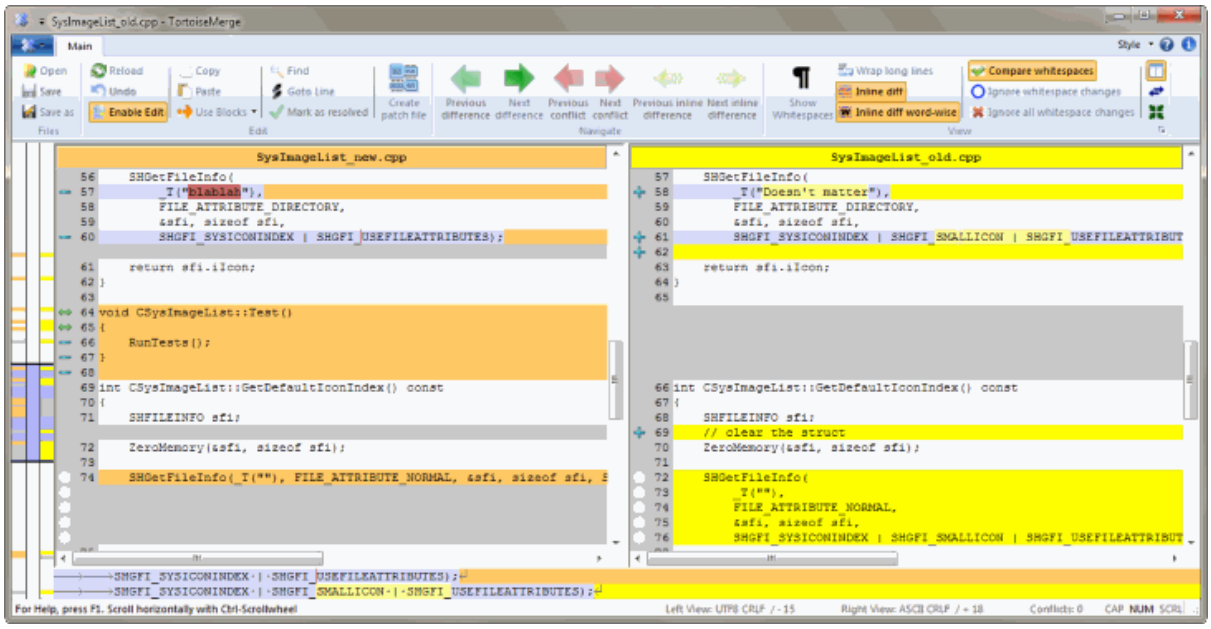


Figure 1.3. Visualiseur de différences

OK, nous sommes satisfaits des changements, mettons à jour le dépôt. Cette action est appelée `Livrer` des modifications. Faites un clic droit sur le dossier `Widget1-dev` et sélectionnez TortoiseSVN → Livrer. La boîte de dialogue de livraison répertorie les fichiers modifiés, chacun avec une case à cocher. Vous pouvez choisir un sous-ensemble de ces fichiers, mais dans ce cas nous allons livrer les modifications apportées aux deux fichiers. Entrez un message pour décrire en quoi consiste le changement et cliquez sur OK. La boîte de dialogue de progression affiche les fichiers en cours de téléchargement dans le dépôt et vous avez terminé.

1.3.5. Ajouter plus de fichiers

Avec le développement du projet, vous aurez besoin d'ajouter de nouveaux fichiers - disons que vous ajoutez de nouvelles fonctionnalités dans `Extras.c` et une référence dans `Makefile`. Faites un clic droit sur le dossier et TortoiseSVN → Ajouter. La boîte de dialogue Ajouter vous montre maintenant tous les fichiers non versionnés et vous pouvez choisir lesquels vous voulez ajouter. Une autre façon d'ajouter des fichiers serait de faire un clic droit sur le fichier lui-même et de sélectionner TortoiseSVN → Ajouter.

Maintenant, quand vous allez livrer le dossier, le nouveau fichier apparaît comme *Ajouté* et le fichier existant comme *Modifié*. Notez que vous pouvez double-cliquer sur le fichier modifié pour vérifier exactement quels changements ont été effectués.

1.3.6. Voir l'historique du projet

L'une des fonctionnalités les plus utiles de TortoiseSVN est la boîte de dialogue de journal. Celle-ci vous affiche la liste de toutes les modifications que vous avez livrées sur un fichier ou un dossier, et montre les messages de

changement détaillés que vous avez entrés (vous *avez bien* entré un message de changement comme suggéré ? Si non, vous voyez maintenant pourquoi c'est important).

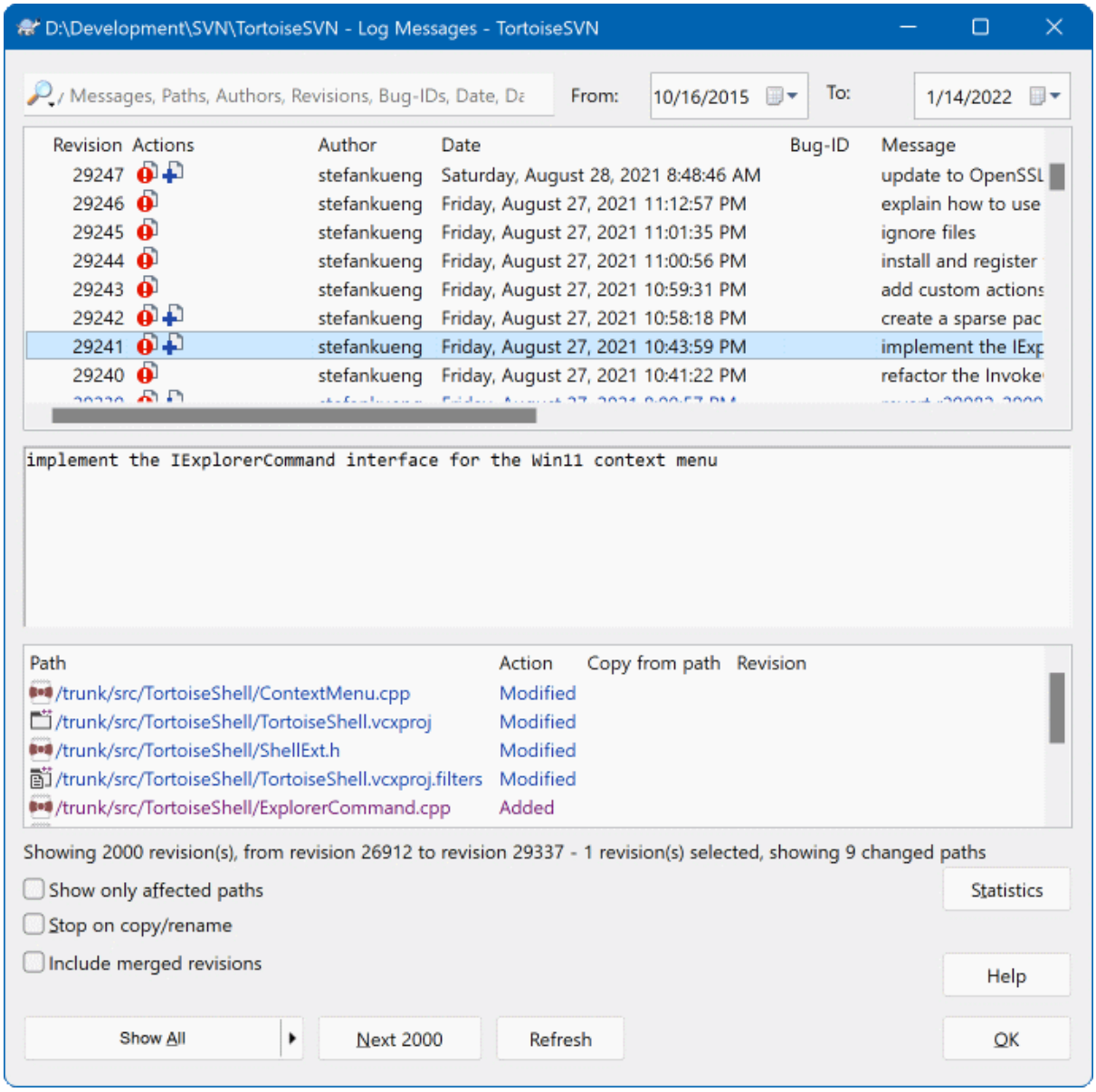


Figure 1.4. La boîte de dialogue de journal

OK, j'ai triché un peu ici, j'ai utilisé une capture d'écran à partir du référentiel TortoiseSVN.

Le panneau supérieur montre une liste des révisions livrées, avec le début du message de livraison. Si vous sélectionnez une de ces révisions, le volet du milieu affiche le message de log complet pour cette révision et le panneau du bas affiche une liste des fichiers et des dossiers modifiés.

Chacun de ces panneaux possède un menu contextuel qui vous donne plusieurs autres moyens d'utiliser les informations. Dans le volet inférieur, vous pouvez double-cliquer sur un fichier pour voir exactement quels changements ont été faits dans cette révision. Lisez Section 4.10, « La boîte de dialogue de journal de révision » pour avoir tout le descriptif.

1.3.7. Annuler des modifications

Une caractéristique de tous les systèmes de contrôle de révision, c'est qu'ils vous permettent d'annuler les modifications que vous avez faites précédemment. Comme on peut s'y attendre, TortoiseSVN facilite cette action.

Si vous voulez annuler les changements que vous n'avez pas encore livrés et reprendre la version précédant vos modifications, le menu TortoiseSVN → Revenir en arrière est votre ami. Cela annule vos modifications (en les mettant dans la corbeille, au cas où) et revient à la version livrée avec laquelle vous avez commencé. Si vous voulez annuler seulement quelques-uns des changements, vous pouvez utiliser TortoiseMerge pour voir les différences et annuler, de manière sélective, les lignes modifiées.

Si vous souhaitez annuler les effets d'une révision particulière, lancez la boîte de dialogue de journal et trouvez la révision en question. Sélectionnez menu contextuel → Annuler les changements de cette révision et ces modifications seront annulées.

1.4. Pour continuer...

Ce guide vous a donné un aperçu très rapide de certaines des caractéristiques les plus importantes et les plus utiles de TortoiseSVN, mais il y en a bien sûr beaucoup d'autres que nous n'avons pas couvertes. Nous vous recommandons fortement de prendre le temps de lire le reste de ce manuel, en particulier Chapitre 4, *Guide d'utilisation quotidienne* qui vous donne beaucoup plus de détails sur les opérations quotidiennes.

Nous nous sommes donné beaucoup de mal pour nous assurer qu'il est à la fois informatif et facile à lire, mais nous reconnaissons qu'il est volumineux ! Prenez votre temps et n'ayez pas peur d'essayer sur un dépôt de test. La meilleure façon d'apprendre est de l'utiliser.

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 brève et informelle introduction à Subversion. Si vous découvrez le contrôle de version, ce chapitre est certainement pour vous. Nous commencerons par une discussion sur les concepts généraux du contrôle de version, nous ferons notre chemin vers les idées spécifiques derrière Subversion et nous montrerons quelques exemples simples de Subversion en utilisation.

Bien que les exemples dans ce chapitre montrent des gens partageant des collections de code source de programmes, gardez à l'esprit que Subversion peut gérer n'importe quelle sorte de collection de fichiers - il ne sert pas qu'à aider des programmeurs.

2.1. Le Dépôt

Subversion est un système centralisé pour partager l'information. Son cœur est un *dépôt*, qui est un centre de stockage de données. Le dépôt stocke l'information sous forme d'une *arborescence de système de fichiers* - une hiérarchie typique de fichiers et de répertoires. Autant de *clients* qu'on veut se connectent au dépôt, puis lisent ou écrivent dans ces fichiers. En écrivant des données, un client rend l'information disponible pour les autres ; en lisant des données, le client reçoit l'information des autres.

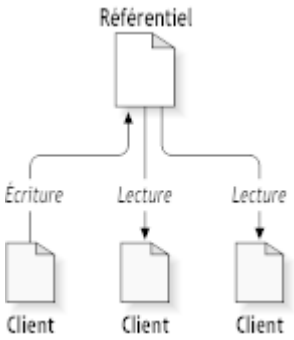


Figure 2.1. Un système client/serveur typique

Alors pourquoi est-ce si intéressant ? Jusqu'ici, cela ressemble à la définition d'un serveur de fichiers typique. Et en effet, le dépôt *est* une sorte de serveur de fichiers, mais il n'est pas de votre genre habituel. Ce qui rend le dépôt 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 dépôt, il voit normalement seulement la dernière version de l'arborescence des fichiers. Mais le client peut aussi voir les états *précédents* du système de fichiers. 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 cœur 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 va-t-il permettre aux utilisateurs de partager l'information, mais les empêcher de se marcher accidentellement sur

les pieds ? Il n'est que trop facile pour les utilisateurs d'écraser accidentellement les modifications des autres sur le dépôt.

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 dépôt en même temps. Si Harry sauvegarde ses changements sur le dépôt en premier, il est possible qu'ensuite (quelques moments plus tard) Sally puisse accidentellement les écraser avec sa propre nouvelle version du fichier. Même si la version d'Harry du fichier ne sera pas perdue pour toujours (parce que le système se rappelle de 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 en premier lieu. Le travail d'Harry est quand même de facto perdu — ou du moins absent de la dernière version du fichier — et probablement par accident. C'est clairement une situation que nous voulons éviter !

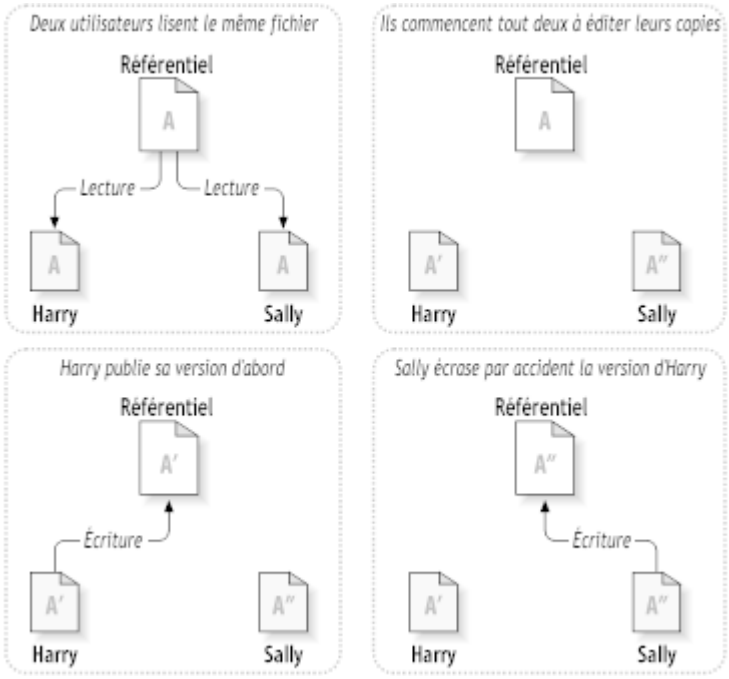


Figure 2.2. Le problème à éviter

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 dépôt ne permet qu'à une seule personne à la foid de modifier un fichier. Tout 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 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 dépôt 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 à son tour le verrouiller et y apporter ses modifications.

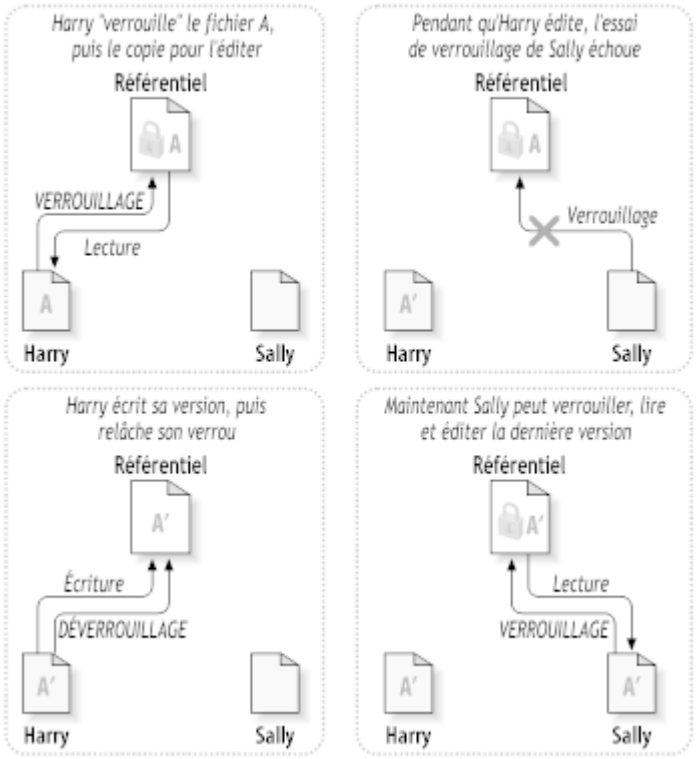


Figure 2.3. La solution Verrouiller-Modifier-Déverrouiller

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, elle est bloquée. Et ensuite Harry part en vacances. Maintenant Sally doit appeler un administrateur pour relâcher le verrou d'Harry. La situation finit par causer beaucoup de retard inutile et de temps perdu.

- *Le verrouillage peut causer une sérialisation inutile.* Que se passe-t-il si Harry édite le début d'un fichier texte, et que 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 rien de bien grave n'arriverait, à supposer que les changements aient été correctement fusionnés ensemble. Ils n'ont aucun besoin d'attendre chacun 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 que les changements faits à chacun soient sémantiquement incompatibles. Soudain, A et B ne fonctionnent plus ensemble. Le système de verrouillage n'a pas pu empêcher le problème - pourtant il a fourni d'une certaine façon 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 et isolée, et ainsi cela ne les incite pas à discuter dès le début de leurs modifications incompatibles.

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 dépôt 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 dépôt. Ils travaillent en même temps et modifient localement le même fichier A. Sally sauvegarde ses changements dans

le dépôt d'abord. Quand Harry essaie de sauvegarder ses changements, le dépôt l'informe que son fichier A est *périmé*. Autrement dit, dans le dépôt, 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 dépôt 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 dépôt.

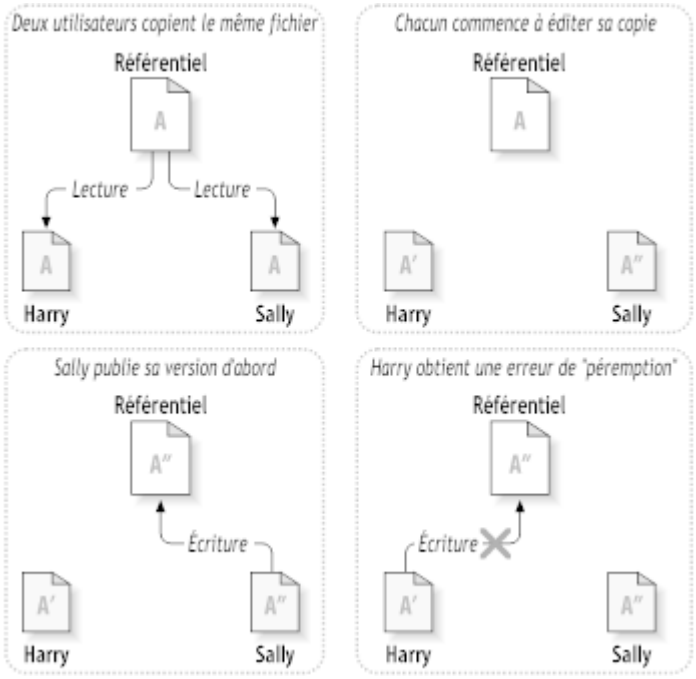


Figure 2.4. La solution Copier-Modifier-Fusionner

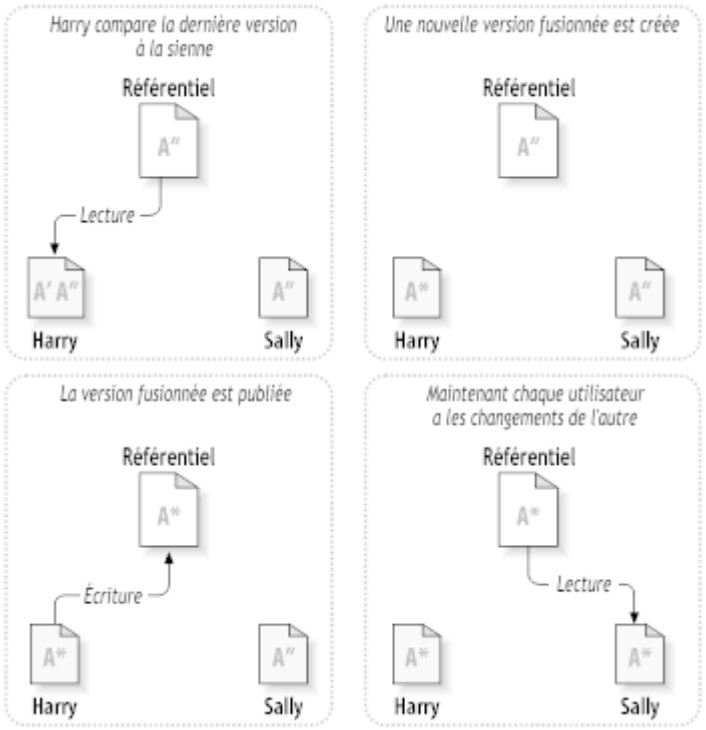


Figure 2.5. ...Suite du modèle Copier-Modifier-Fusionner

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 dépôt 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 dépôt.

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 dépôt 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 avez 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 dépôt, 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. En particulier, votre copie de travail contient un sous-répertoire nommé `.svn`, aussi connu comme le *répertoire administratif* de la copie de travail. Les fichiers de ce répertoire administratif aident Subversion à reconnaître quels fichiers contiennent des changements non publiés et quels fichiers sont périmés par rapport au travail des autres. Avant la version 1.7, Subversion maintenait un répertoire administratif `.svn` dans chaque répertoire versionné de votre copie de travail. Subversion 1.7 adopte une approche complètement différente et chaque copie de travail a maintenant un seul répertoire administratif, qui est situé immédiatement sous la racine de cette copie de travail.

Un dépôt 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 dépôt. Dans cette optique, la copie de travail d'un utilisateur correspondra d'habitude à un sous-arbre particulier du dépôt.

Par exemple, supposons que vous ayez un dépôt contenant deux projets d'application.

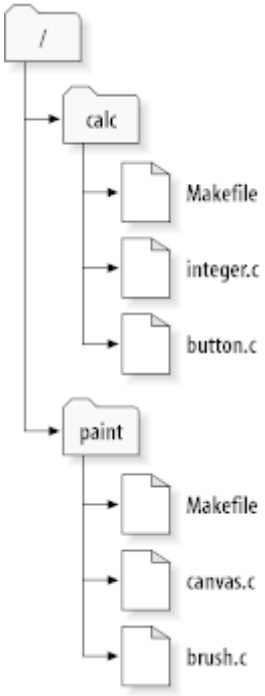


Figure 2.6. Le système de fichiers du dépôt

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

Afin d'obtenir une copie de travail, il faut *extraire* une partie de l'arborescence du dépôt. (Le terme anglais *check out* pourrait suggérer qu'il s'agit de verrouiller ou de réserver des ressources du dépôt, mais ce n'est pas le cas. Il s'agit simplement d'une copie privée, également appelé bac à sable, du projet pour l'utilisateur.)

Supposons que vous fassiez 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'action de publier vos changements est plus communément appelée *livrer* (ou *valider*) les changements au dépôt.

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

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

Supposons que vous ayez 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 ne modifie les copies de travail qu'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 tous les 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 dépôt, pour décider quels fichiers doivent être actualisés.

2.3.2. URL de dépôt

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

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

Tableau 2.1. URL d'accès au dépôt

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 dépôts 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 dépôt :

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

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 accéder à un répertoire FSFS via un partage réseau, mais ce n'est *pas* recommandé pour plusieurs raisons :

- Vous donnez un accès direct en écriture à tous les utilisateurs, ils pourraient donc accidentellement supprimer ou corrompre le système de fichiers du dépôt.
- Tous les protocoles de partage de fichiers en réseau ne supportent pas le verrouillage que requiert Subversion. Un beau jour, vous découvrirez que votre dépôt a été sournoisement *corrompu*.
- Vous devez paramétrer les permissions d'accès de la bonne façon. SAMBA est particulièrement difficile à cet égard.
- Si un utilisateur installe une nouvelle version du client qui met à jour le format du dépôt, tous les autres seront alors dans l'incapacité d'accéder au dépôt jusqu'à ce qu'ils mettent eux aussi leur client à la nouvelle version.

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 dépôt, 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 dépôt 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 dépôt récemment créé est numérotée zéro et ne consiste en rien d'autre qu'un répertoire racine vide.

Une façon agréable de visualiser le dépôt 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 dépôt ressemble après chaque livraison.

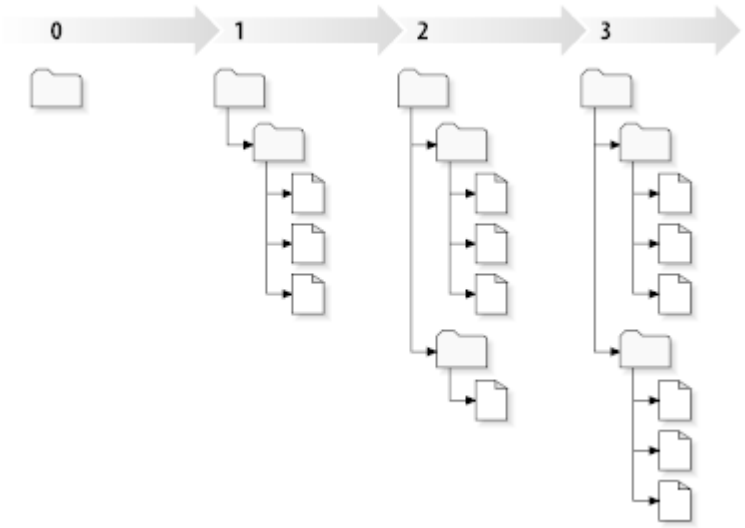


Figure 2.7. Le Dépôt

> **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 dépôt après un certain changement livré. Une autre façon d'y penser est que cette révision N représente l'état du système de fichiers du dépôt après que la Nième livraison. Quand un utilisateur de Subversion parle de « la révision 5 de toto.c

Il est important de noter que les copies de travail ne correspondent pas toujours à une seule révision dans le dépôt ; elles peuvent contenir des fichiers de plusieurs révisions différentes. Par exemple, supposons que vous extrayiez une copie de travail d'un dépôt 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 dépôt. Cependant, supposons que vous fassiez un changement sur `button.c` et que vous livriez ce changement. En admettant qu'aucune autre livraison n'ait eu lieu, votre livraison créera la révision 5 du dépôt, 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 sera toujours présent dans `button.c`. Dans cet exemple, le texte de `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 à jour. 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 dépôt.

2.3.4. Comment les copies de travail suivent le dépôt

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 dépôt.

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

Inchangé et à jour
: Le fichier est inchangé dans le répertoire de travail et aucun changement sur ce fichier n'a été livré au dépôt 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 à jour
: Le fichier a été changé dans le répertoire de travail et aucun changement sur ce fichier n'a été livré au dépôt depuis sa révision de base. Il y a des changements locaux qui n'ont pas été livrés au dépôt, 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 dépôt. 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 dépôt. Une **livraison** du fichier échouera avec une erreur *périmé*. Il faut d'abord mettre à jour le fichier ; la commande **mettre à jour** essayera de fusionner les deux versions. Si Subversion ne parvient 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 de dépôt central, de copie de travail du client et de tableau d'arbres de révision de dépôt.
- Nous avons vu quelques exemples simples montrant 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 Dépôt

Peu importe le protocole que vous utilisez pour avoir accès à vos dépôts, vous devez toujours créer au moins un dépôt. 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 dépôt Subversion, il est temps de le faire maintenant.

3.1. Création de dépôt

3.1.1. Créer un dépôt 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 dépôts.
2. Créez un autre dossier `MonNouveauDépôt` dans `D:\SVN\`.
3. Ouvrez l'invite de commande (ou DOS-Box), allez dans `D:\SVN\` et tapez

```
svnadmin create --fs-type fsfs MonNouveauDépôt
```

Maintenant vous avez un nouveau dépôt situé dans `D:\SVN\MonNouveauDépôt`.

3.1.2. Créer le dépôt avec TortoiseSVN

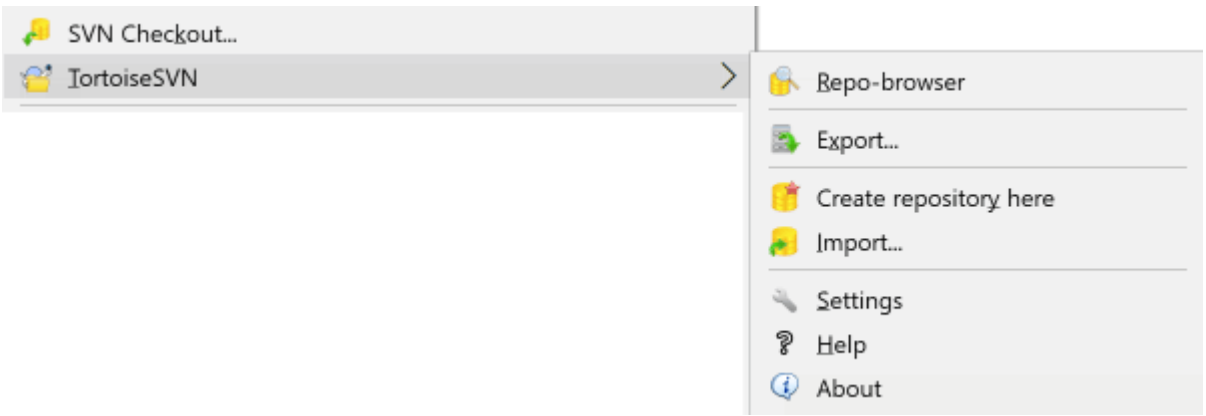


Figure 3.1. Le menu TortoiseSVN pour les dossiers non versionnés

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

   Un dépôt 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.

   Il vous sera également demandé si vous souhaitez créer une structure de répertoire dans le dépôt. Renseignez-vous sur les options de structuration ici : Section 3.1.5, « Disposition du dépôt ».

   TortoiseSVN créera une icône de dossier personnalisé lors de la création d'un dépôt afin que vous puissiez identifier les dépôts locaux plus facilement. Si vous créez un dépôt en utilisant le client en ligne de commande, cette icône de dossier n'est pas créée.

Astuce

Nous vous recommandons également de ne pas utiliser l'accès `file://` du tout, sauf pour des tests locaux. Utiliser un serveur est plus sécurisé et fiable pour toute utilisation comptant plus d'un développeur.

3.1.3. Accès local au dépôt

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

Pour avoir accès à un dépôt placé sur une partage réseau vous pouvez utiliser soit le mappage de disque, soit le chemin UNC. Pour les chemins UNC, la forme est `file://NomDuServeur/chemin/vers/dépôt/`. 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/dépôt` Cette forme est toujours supportée, mais n'est pas recommandée.

3.1.4. Accéder à un dépôt situé dans un partage réseau

Bien qu'il soit possible en théorie d'utiliser un dépôt FSFS sur un partage réseau et d'avoir un accès multi-utilisateurs en utilisant le protocole `file://`, ce n'est certainement *pas* recommandé. En fait, nous le déconseillons *fortement*, et ne supportons pas un tel usage pour les raisons suivantes :

- Premièrement, vous donnez le droit d'accès en écriture dans le dépôt à tous les utilisateurs, de telle sorte que chacun peut supprimer accidentellement le dépôt ou le rendre complètement inutilisable d'une façon ou d'une autre.
- Deuxièmement, tous les protocoles d'échange de fichiers ne supportent pas le verrouillage dont Subversion a besoin, donc votre dépôt peut être corrompu. Cela ne va peut-être pas se produire tout de suite, mais un jour deux utilisateurs vont essayer d'accéder au dépôt en même temps.
- Troisièmement, les droits d'accès aux fichiers doivent être attribués très précisément. Vous pouvez vous en tirer sur un partage Windows, mais SAMBA est particulièrement compliqué.
- Si un utilisateur installe une nouvelle version du client qui met à jour le format du dépôt, tous les autres seront alors dans l'incapacité d'accéder au dépôt jusqu'à ce qu'ils mettent eux aussi leur client à la nouvelle version.

Les accès de type `file://` sont uniquement prévus pour une utilisation locale et mono-utilisateur, en particulier à des fins de test et de debug. Lorsque vous voulez partager votre dépôt, vous devez *vraiment* mettre en place un vrai serveur, et c'est loin d'être aussi compliqué qu'il y parait. Lisez Section 3.5, « Accéder au dépôt » pour obtenir des conseils sur le choix et l'installation d'un serveur.

3.1.5. Disposition du dépôt

Avant que vous n'importiez vos données dans le dépôt, vous devriez d'abord réfléchir 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 manières standard et recommandées d'organiser un dépôt. La plupart des personnes créent un répertoire `trunk` contenant la « version principale » de développement, un répertoire `branches` qui contient les copies de travail parallèles et un répertoire `tags` qui contient les versions stables. Si un dépôt ne contient qu'un seul projet, ces répertoires sont créés à la base du dépôt :

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

Comme cette structure est la plus couramment utilisée, lorsque vous créez un nouveau dépôt avec TortoiseSVN, il vous proposera aussi de créer la structure de répertoire à votre place.

Si un dépôt contient plusieurs 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 `/trunk /tags /branches` au niveau supérieur, il n'y a rien qui vous impose de copier le tronc en entier pour chaque branche et chaque étiquette, et d'une certaine façon, c'est cette structure qui offre le plus de flexibilité.

Pour les projets sans rapport vous pouvez préférer utiliser des dépôts séparés. Quand vous livrez des changements, c'est le numéro de révision du dépôt entier qui change, pas le numéro de révision du projet. Avoir 2 projets sans rapport qui partagent un dépôt peut signifier de grands écarts dans les numéros de révision. Les projets Subversion et TortoiseSVN apparaissent à la même adresse hôte, mais sont des dépôts 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 quelle 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 dépôt à tout moment. Comme 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 dépôt, jonglez juste avec les répertoires.

Si vous n'avez pas encore créé la structure de base de votre dépôt, vous devriez le faire maintenant. Il y a deux manières de le faire. Si vous souhaitez simplement une structure `/trunk /tags /branches`, vous pouvez utiliser l'explorateur de dépôt 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 dépôt via un clic droit sur le dossier qui contient cette structure de dossier et en choisissant TortoiseSVN → Importer.... Dans la boîte de dialogue d'importation entrez l'URL de votre dépôt et cliquez sur OK. Cela importera votre dossier temporaire dans la racine du dépôt pour créer la structure de base du dépôt.

Notez que le nom du dossier que vous importez n'apparait pas dans le dépôt, seulement son contenu. Par exemple, créez l'arborescence suivante :

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

Importez `C:\Temp\New` à la racine du dépôt, elle ressemblera alors à ceci :

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

3.2. Sauvegarde de dépôt

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

La manière la plus simple (mais pas la plus recommandée) est de copier le dossier du dépôt sur votre support de sauvegarde. 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. Si quoi que ce soit accède à votre dépôt pendant la copie (navigateur Internet laissé ouvert, WebSVN, etc.) la sauvegarde sera inutile.

La méthode recommandée est d'exécuter

```
svnadmin hotcopy chemin/du/dépôt chemin/de/la/sauvegarde
```

pour créer une copie de votre dépôt de manière sûre. Sauvegardez ensuite la copie.

L'outil `svnadmin` est installé automatiquement quand vous installez le client de ligne de commande de Subversion. La manière la plus simple de l'obtenir est de cocher l'option pour inclure les outils de ligne de commande lors de l'installation de TortoiseSVN, mais si vous le préférez, vous pouvez télécharger la dernière version des outils de ligne de commande directement depuis le site Internet de *Subversion* [https://subversion.apache.org/packages.html#windows].

3.3. Scripts hook côté serveur

Un script hook, ou script associé, est un programme déclenché par un certain événement du dépôt, comme la création d'une nouvelle révision ou la modification d'une propriété non versionnée. Chaque hook reçoit suffisamment d'informations pour dire de quel événement il s'agit, sur quelle(s) cible(s) il doit fonctionner et le nom d'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 référée au chapitre sur *Scripts hook* [http://svnbook.red-bean.com/en/1.8/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 hook sont exécutés par le serveur qui héberge le dépôt. TortoiseSVN permet également de configurer des scripts hook côté client à exécuter en local en réponse à certains événements. Voir Section 4.31.8, « Scripts hook côté client » pour plus d'information.

Des exemples de scripts hook se trouvent dans le répertoire `hooks` du dépôt. Ces exemples de scripts sont utilisables sur des serveurs Unix/Linux, mais doivent être modifiés si votre serveur est sur une machine Windows. Le hook peut être un fichier batch ou un exécutable. L'exemple ci-dessous montre un fichier batch qui pourrait être utilisé pour implémenter un hook pre-revprop-change.

```
rem N'autoriser que les modifications de commentaires.
if "%4" == "svn:log" exit 0
echo Impossible de modifier la propriété '%4' >&2
exit 1
```

Veuillez remarquer que tout ce qui est envoyé sur stdout est ignoré. Si vous voulez qu'un message apparaisse dans la boîte de dialogue de refus de livraison, il faut l'envoyer sur stderr. Dans un fichier batch, cela se fait en utilisant `>&2`.

Surcharger les hooks

Si un script de hook refuse votre livraison alors ce refus est définitif. Mais vous pouvez construire un mécanisme de contournement dans ce même script en utilisant la technique du *Mot Magique*. Si le script veut refuser l'opération, il commence par chercher dans le message du journal la permission spéciale, soit une phrase à un endroit fixe ou peut-être le nom du fichier avec un préfixe. S'il trouve le mot magique alors il autorise la livraison à s'exécuter. Si la phrase n'est pas trouvée alors il peut bloquer la livraison avec un message du type « Vous n'avez pas dit le mot magique ». :-)

3.4. Liens d'extraction

Si vous voulez donner l'accès à votre dépôt 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 dépôt 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.projet.org/svn/trunk">
</a>
```

Bien sûr, cela aura encore meilleure apparence si vous incluez une image correcte. Vous pouvez utiliser le *logo TortoiseSVN* [https://tortoisesvn.net/images/TortoiseCheckout.png] ou vous pouvez fournir 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 dépôt

Pour utiliser TortoiseSVN (ou un autre client Subversion), vous avez besoin d'un hébergement pour vos dépôts. 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 chiffrés. Vous utiliserez alors `https://`, `svn+ssh://`, ou encore `svn://` avec SASL.

Si vous utilisez un hébergeur public comme *SourceForge* [https://sourceforge.net] ou que votre serveur a déjà été configuré par quelqu'un d'autre, alors vous n'avez besoin de rien faire d'autre. Allez directement au 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 dépôts locaux sont probablement la meilleure solution. Créez juste un dépôt sur votre ordinateur comme décrit dans Chapitre 3, *Le Dépôt*. 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 dépôt destiné à être utilisé par plusieurs utilisateurs sur un partage réseau n'est pas conseillé. Lisez Section 3.1.4, « Accéder à un dépôt 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.

Des informations plus détaillées sur les options du serveur Subversion et comment choisir la meilleure architecture pour votre cas, peuvent être trouvées dans le livre Subversion sous *Configuration du serveur* [http://svnbook.red-bean.com/en/1.8/svn.serverconfig.html].

Aux temps anciens de Subversion, installer un serveur nécessitait une bonne compréhension de la configuration du serveur, et dans les versions précédentes de ce manuel nous incluions des descriptions détaillées de la mise en place d'un serveur. Depuis, les choses sont devenues plus faciles, car il existe maintenant plusieurs installateurs pré-packagés de serveur, qui vous guident tout au long du processus d'installation et de configuration. Voici des liens vers certains des installateurs que nous connaissons :

- *VisualSVN* [https://www.visualsvn.com/server/]
- *CollabNet* [https://www.collab.net/products/subversion]

Vous pouvez toujours trouver les liens les plus récents sur le site Internet de *Subversion* [https://subversion.apache.org/packages.html].

Vous pouvez trouver d'autres guides pratiques sur le site Internet de *TortoiseSVN* [https://tortoisesvn.net/usefultips.html].

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.

Si vous avez besoin d'une introduction au contrôle de version avec Subversion, alors nous vous recommandons la lecture de l'excellent livre : *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 signaler sur la liste de diffusion 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 en excuser. Nous travaillons sur TortoiseSVN pendant notre temps libre.

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

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

4.1. Fonctionnalités générales

Cette section décrit certaines fonctionnalités de TortoiseSVN qui s'appliquent à à peu près tout dans le manuel. Notez que beaucoup de ces caractéristiques n'apparaîtront que dans une copie de travail Subversion.

4.1.1. Icônes superposées

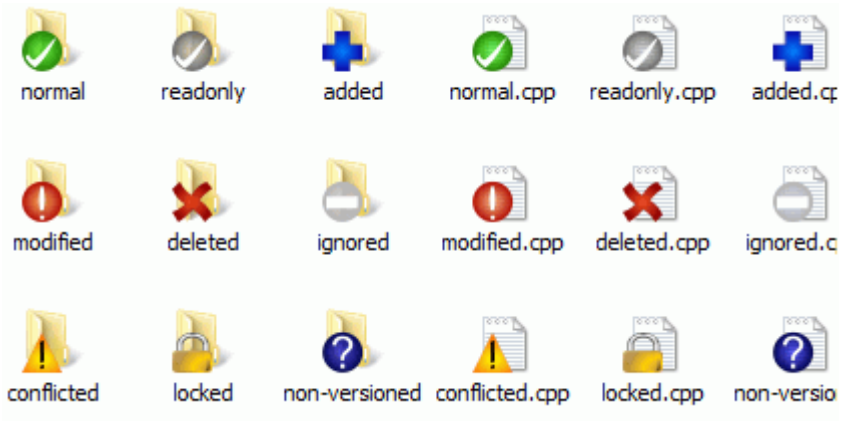


Figure 4.1. L'explorateur affichant des icônes superposées

Une des fonctionnalités les plus visibles de TortoiseSVN est la superposition d'icônes qui apparaissent sur les fichiers de votre copie de travail. Cela vous permet de voir au premier coup d'œil quels fichiers ont été modifiés. Référez-vous à Section 4.7.1, « Icônes superposées » pour découvrir que représentent les différentes icônes.

4.1.2. Menus contextuels

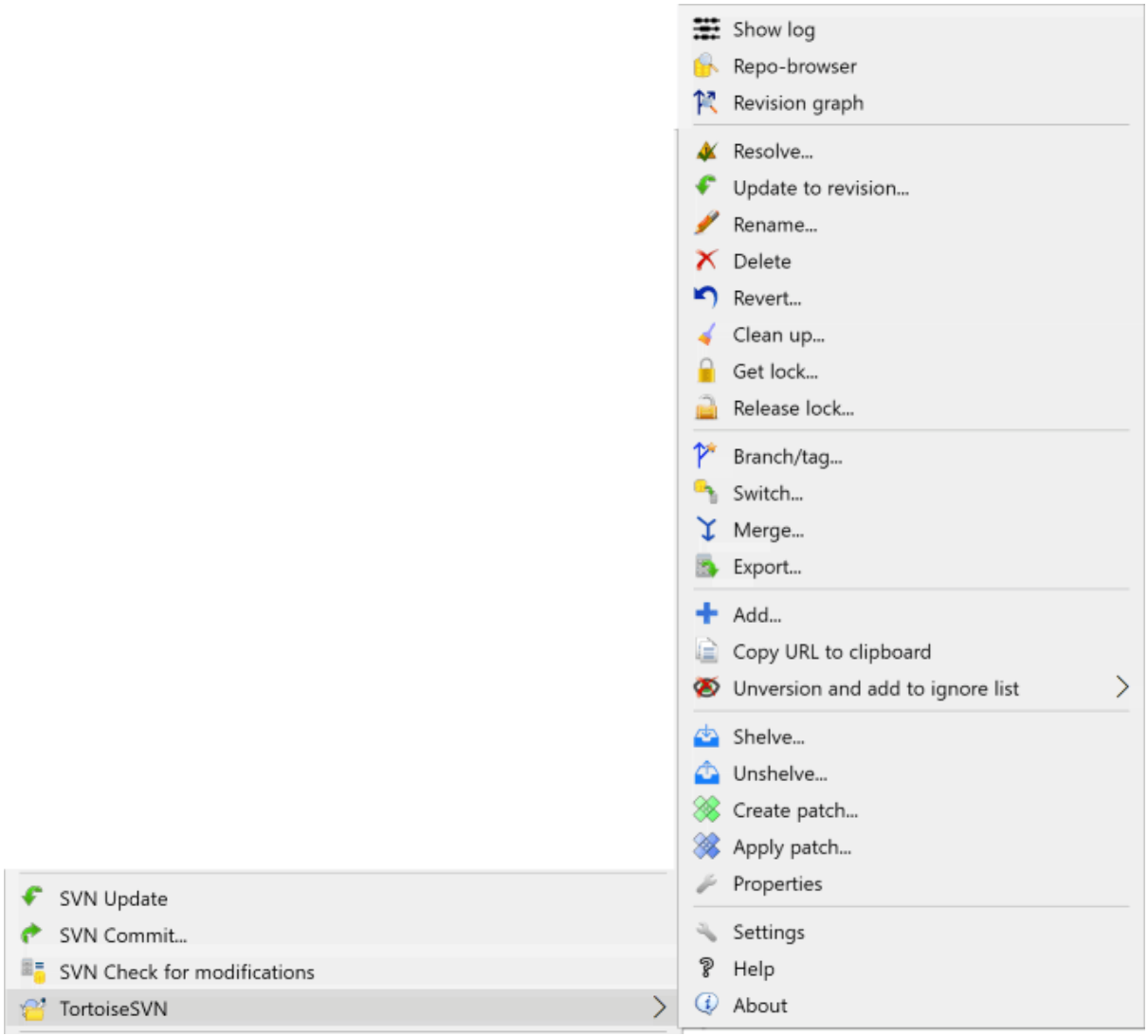


Figure 4.2. Menu contextuel pour un répertoire sous contrôle de version

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 de si le fichier ou le dossier ou son dossier parent sont sous contrôle de version ou non. Les commandes du menu TortoiseSVN sont également disponibles dans le menu Fichier de l'explorateur.

Astuce

Certaines commandes qui sont très rarement utilisées ne sont disponibles que dans le menu contextuel étendu. Pour faire apparaître le menu contextuel étendu, maintenez enfoncée la touche **Maj** lorsque vous faites un clic 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 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 plusieurs éléments sélectionnés.

4.1.3. Glisser-déposer

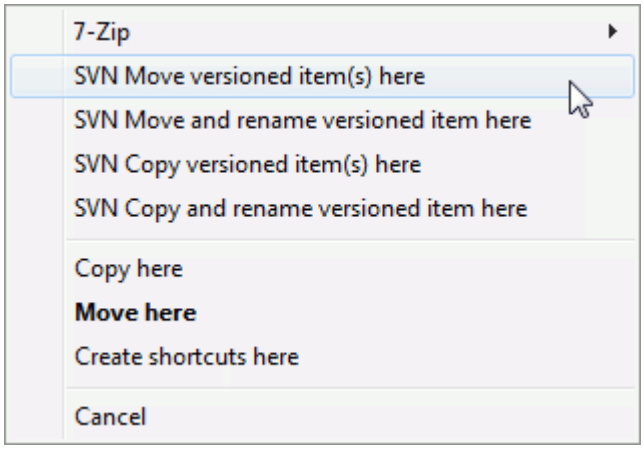


Figure 4.4. Menu pour un répertoire sous contrôle de version lors d'un glisser-déplacer avec le bouton droit

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 en une touche la plus utile. Par exemple... Dans l'explorateur, cela rafraîchira les icônes superposées sur votre copie de travail. Dans la boîte de dialogue de livraison, il parcourra à nouveau la copie de travail pour voir ce qui pourrait être livré. Dans la boîte de dialogue de journal de révision, il entrera à nouveau en contact avec le dépôt 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 le copier-coller dans un e-mail. Utilisez Ctrl-A pour sélectionner le message d'erreur et ensuite...

Ctrl-C
: Copier le texte sélectionné. Dans le cas où aucun texte n'est sélectionné, mais que, par exemple, une entrée de liste ou une boîte de message l'est, le contenu de cette entrée de liste ou de cette boîte est copié dans le presse-papier.

4.1.5. Authentification

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

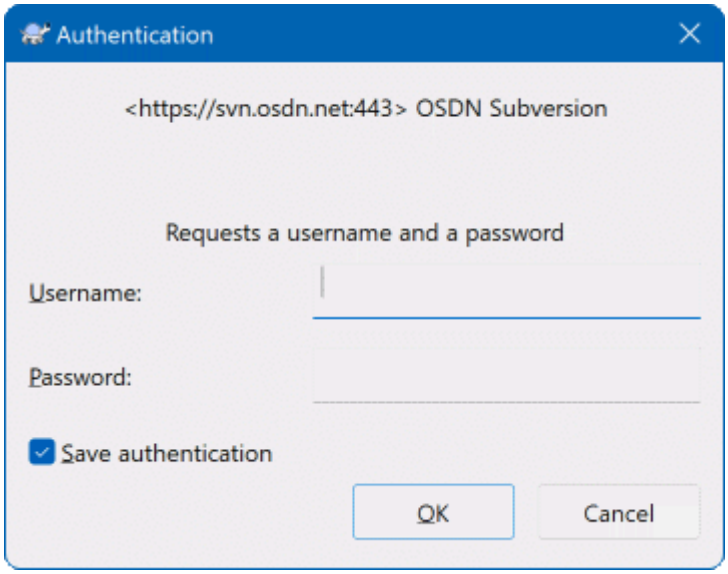


Figure 4.5. Boîte de dialogue d'authentification

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` contient les accréditations pour l'authentification de base (nom d'utilisateur/mot de passe). Notez que les mots de passe sont stockés en utilisant l'API WinCrypt, pas en clair.
- `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, vous pouvez le faire à partir de la page Données sauvegardées de la boîte de dialogue de configuration de TortoiseSVN. Le bouton Effacer tout effacera les données d'authentification en cache pour tous les dépôts. Le bouton Effacer..., quant à lui, affichera une boîte de dialogue dans laquelle vous pourrez choisir quelles données d'authentification en cache effacer. Référez-vous à Section 4.31.6, « Configuration des données sauvegardées ».

Certaines personnes préfèrent effacer les données d'authentification quand elles se déconnectent de Windows, ou quand elles éteignent l'ordinateur. La façon de faire cela est d'utiliser un script de déconnexion pour supprimer le répertoire `%APPDATA%\Subversion\auth`, par exemple

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

Vous pouvez trouver une description de la façon d'installer ce genre de scripts à http://www.windows-help-central.com/windows-shutdown-script.html.

Pour plus d'informations sur la façon de configurer votre serveur pour l'authentification et contrôle d'accès, reportez-vous à Section 3.5, « Accéder au dépôt » .

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 dépôt

4.2.1. Importer

Si vous importez des éléments dans un dépôt contenant déjà des projets, alors la structure du dépôt aura déjà été décidée. Si vous importez des éléments dans un nouveau dépôt alors il est temps de penser à la manière de l'organiser. Lisez Section 3.1.5, « Disposition du dépôt » pour avoir des conseils.

Cette section décrit la commande Importer de Subversion, qui a été conçue pour importer la hiérarchie d'un dossier dans le dépôt d'un seul coup. Même si ça fait le boulot, cela a plusieurs 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 dépôt.

Pour ces raisons nous vous recommandons de ne pas utiliser du tout la commande Importer, mais plutôt de suivre la méthode en 2 étapes décrite dans Section 4.2.2, « Importer en place », sauf si vous faites la simple étape de création de la structure initiale `/trunk /tags /branches` dans votre dépôt. Puisque vous êtes là, voici comment l'import basique fonctionne...

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

1. Supprimer 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. Organiser les fichiers en dossiers et sous-dossiers. Bien qu'il soit possible de renommer/déplacer les fichiers plus tard, il est fortement recommandé de clarifier la structure de votre projet avant l'importation !

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

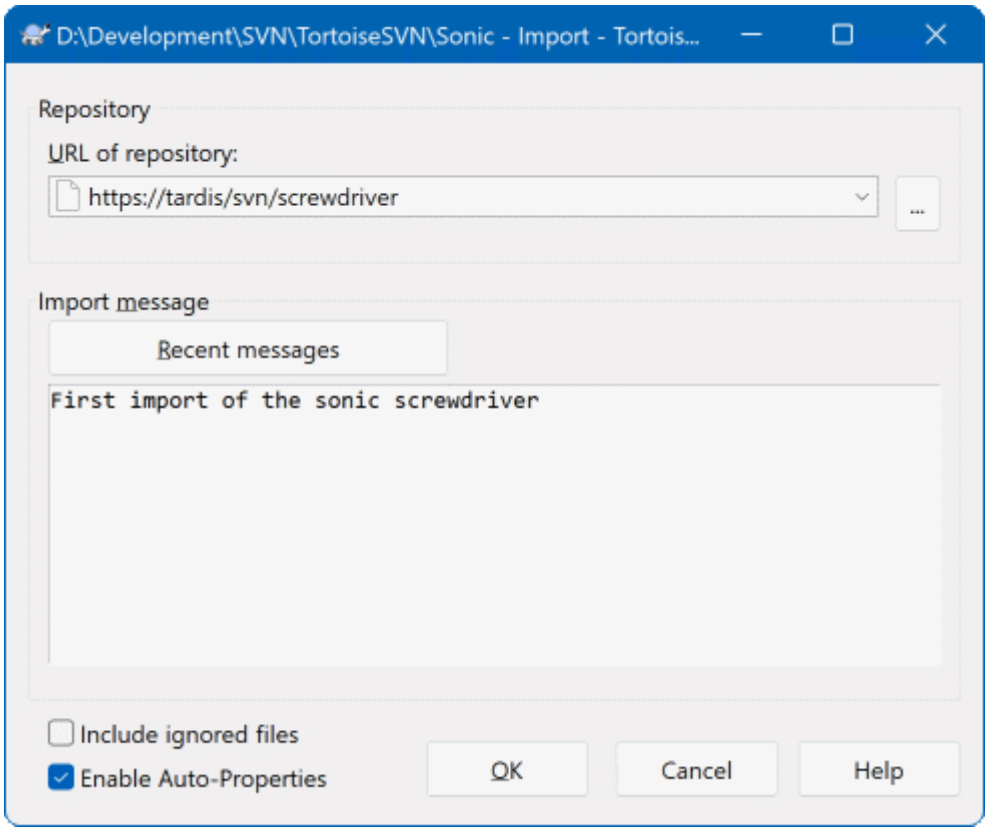


Figure 4.6. La boîte de dialogue Importer

Dans cette boîte de dialogue, vous devez saisir l'URL du dépôt dans lequel vous voulez importer votre projet. Il est très important de comprendre que le répertoire local que vous importez n'apparaît pas lui-même dans le dépôt, 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`, alors vous pourriez être surpris de voir que vos sous-répertoires iront directement sous `trunk` au lieu d'être dans un sous-répertoire `Widget`. Vous devez spécifier le sous-répertoire comme faisant partie de l'URL, `http://monnomdedomaine.com/svn/trunk/Widget-X`. Remarquez que la commande Importer créera automatiquement les sous-répertoires dans le dépôt s'ils n'existent pas.

Le message d'importation est utilisé comme 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 4.31.1, « Configuration générale » pour plus d'informations sur la configuration d'un filtre d'exclusion.

Dès que vous appuyez sur OK TortoiseSVN importe dans le dépôt l'arborescence complète des répertoires, fichiers compris. Le projet est maintenant sous contrôle de version dans le dépôt. 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 dépôt, et que vous souhaitez y ajouter une nouvelle arborescence, suivez les étapes suivantes :

1. Utilisez l'explorateur de dépôt pour créer un nouveau dossier de projet directement dans le dépôt. Si vous utilisez une des dispositions standard, vous voudrez probablement le créer comme un sous-dossier de trunk, plutôt qu'à la racine du dépôt. L'explorateur de dépôt montre la structure du dépôt juste comme l'explorateur Windows, afin que vous puissiez voir comment les choses sont organisées.

2. Extrayez le nouveau répertoire par-dessus le répertoire que vous voulez importer. Un message d'avertissement va s'afficher, indiquant que le répertoire local n'est pas vide. Ignorez l'avertissement. Vous disposez maintenant d'un répertoire de plus haut niveau versionné, dont le contenu n'est pas versionné.

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 sous contrôle de version un fichier qui contient des données spécifiques aux utilisateurs. Cela signifie que vous avez un fichier que chaque développeur/utilisateur doit adapter 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.

À titre d'exemple, vous pouvez regarder le script de build de TortoiseSVN. Il appelle un fichier nommé `default.build.user` qui n'existe pas dans le dépôt. Seul le fichier `default.build.user.tmpl` existe. `default.build.user.tmpl` est le fichier modèle, dont chaque développeur doit créer une copie et renommer cette copie en `default.build.user`. À l'intérieur de ce fichier, nous avons ajouté des commentaires pour que les utilisateurs voient quelles sont les lignes qu'ils doivent éditer et modifier selon leur paramétrage local pour qu'il fonctionne.

Afin de ne pas déranger les utilisateurs, nous avons aussi ajouté le fichier `default.build.user` à la liste des fichiers ignorés de son répertoire parent, c'est-à-dire que nous avons positionné 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 dépôt.

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 :

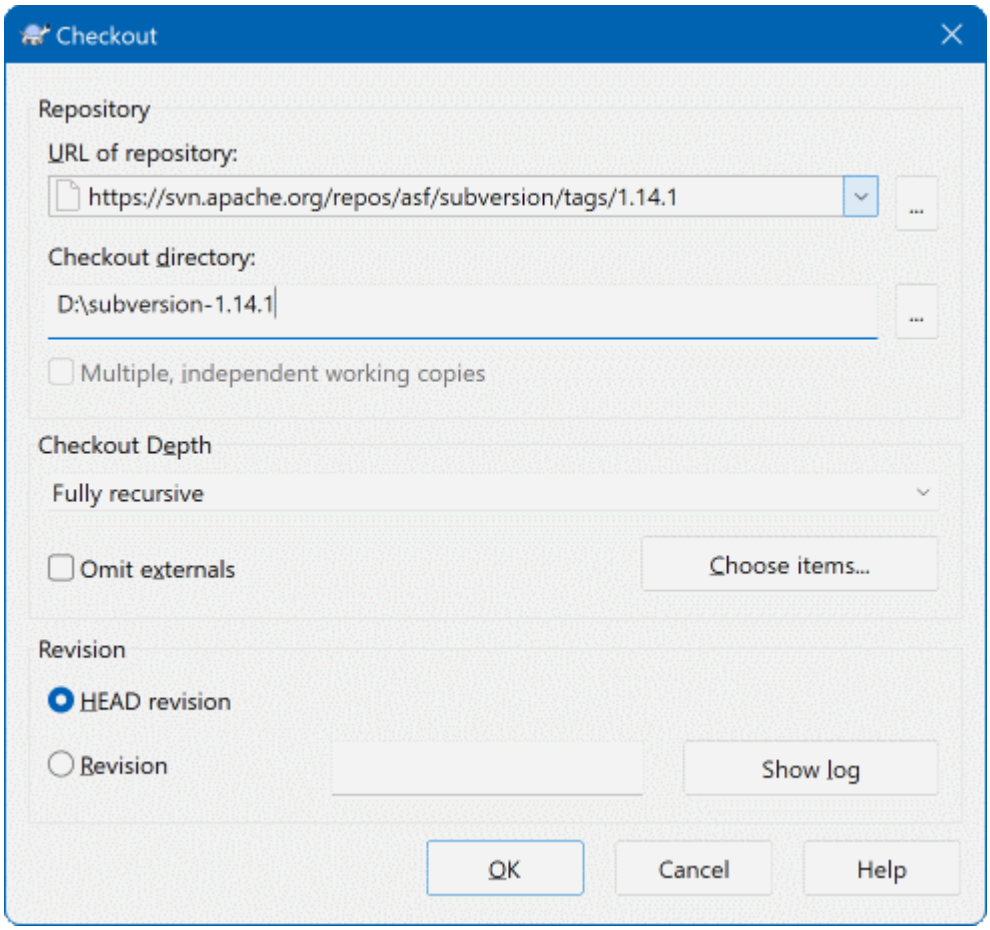


Figure 4.7. La boîte de dialogue Extraire

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

Important

Dans les réglages par défaut, la commande Extraire n'est pas située dans le sous-menu TortoiseSVN mais est affichée directement dans le menu contextuel de l'explorateur. Les commandes de TortoiseSVN qui ne sont pas dans le sous-menu ont le terme SVN préfixé: SVN Extraire...

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 choisir les sous-répertoires à mettre à jour récursivement.

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

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

Uniquement les fichiers
: Extrait le répertoire spécifié, y compris les fichiers qu'il contient, mais n'extrait aucun sous-répertoire.

Uniquement cet élément
: Extrait juste le répertoire. Ne le remplit pas 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 défaut pour toutes les autres fenêtres qui permettent de spécifier la profondeur.

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

Pour sélectionner facilement les seuls éléments que vous voulez pour l'extraction et forcer la copie de travail obtenue à ne garder que ces éléments, cliquez sur le bouton Choisir les éléments.... Ceci ouvre une nouvelle boîte de dialogue où vous pouvez cocher tous les éléments dont vous voulez dans votre copie de travail et décocher tous les éléments dont vous ne voulez pas. La copie de travail qui en résulte est alors appelée une `extraction éparse` . La mise à jour d'une telle copie de travail ne récupérera pas les fichiers et dossiers manquants, mais ne mettra à jour que ceux que vous avez déjà dans votre copie de travail.

Si vous procédez à l'extraction d'une copie de travail éparse (c'est-à-dire en choisissant autre chose que `totalement récursive` pour la profondeur d'extraction), vous poivez facilement ajouter ou supprimer des sous-répertoires plus tard en utilisant une des méthodes suivantes.

4.3.1.1. Mise à jour éparse avec Mettre à jour à la révision

Faites un clic droit sur le répertoire d'extraction, puis utilisez TortoiseSVN → Mettre à jour à la révision et sélectionnez Choisir les éléments.... Cela ouvre la même boîte de dialogue que dans l'extraction de départ et vous permet de sélectionner ou de désélectionner des éléments à inclure dans l'extraction. Cette méthode est très souple, mais elle peut être lente, car chaque élément dans le répertoire est mis à jour individuellement.

4.3.1.2. Mise à jour éparse avec l'explorateur de dépôt

Faites un clic droit sur le répertoire d'extraction, puis utilisez TortoiseSVN → Explorateur de dépôt pour afficher l'explorateur de dépôt. Trouvez le sous-dossier que vous voulez ajouter à votre copie de travail, puis utilisez Menu contextuel → Mettre à jour l'élément à la révision....

4.3.1.3. Mise à jour éparse avec Vérifier les modifications

Dans la boîte de dialogue Vérifier les modifications, faites d'abord **Maj** + clic sur le bouton Vérifier le dépôt. Tous les fichiers et dossiers qui sont dans le dépôt mais que vous n'avez pas extraits apparaîtront avec le statut des propriétés distantes à `modifié`. Faites un clic droit sur le(s) dossier(s) que vous voulez ajouter à votre copie de travail, puis sélectionnez Menu contextuel → Mettre 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. Extrayez le répertoire parent avec la profondeur « Juste cet élément ». Vous avez maintenant un répertoire de premier niveau vide.

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

3. Faites un clic droit sur `Projet03` et Menu contextuel → Mettre à jour à la révision.... Laissez les paramètres par défaut et cliquez sur OK. Ce répertoire est à présent complètement rempli.

   Répétez le même processus pour `Projet25`.

4. Allez dans `Projet76/SousProj` et faites de même. Cette fois-ci, notez que le répertoire `Projet76` est vide, à l'exception de `SousProj`, 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 profondeur donnée, vous pouvez modifier cette profondeur ultérieurement pour obtenir plus ou moins de contenu en utilisant Menu contextuel → Mettre à jour à la révision.... Dans cette boîte de dialogue, prenez soin de cocher la case Rendre la profondeur persistante.

Utilisation d'une ancienne version de serveur

Les serveurs antérieurs à la version 1.5 ne supportent pas la demande de profondeur de copie de travail, et ne peuvent donc pas toujours gérer efficacement les requêtes. La commande fonctionnera quand même, mais un serveur ancien risque d'envoyer toutes les données, laissant au client le soin de filtrer ce qui n'est pas pertinent, ce qui peut causer beaucoup de trafic réseau. Si c'est possible, vous devriez mettre à jour votre serveur au minimum vers la version 1.5.

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 références externes .

Important

Si Omettre les références externes est coché, ou si vous voulez augmenter la profondeur, vous devrez faire des mises à jour de votre copie de travail en utilisant TortoiseSVN → Mettre à jour à la révision... au lieu de TortoiseSVN → Mettre à jour. La mise à jour standard inclura toutes les références externes et conservera la profondeur actuelle.

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 dépôt 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 4.27, « Exporter une copie de travail Subversion » pour savoir comment le faire.

4.4. Livrer vos changements dans le dépôt

Envoyer les changements que vous avez faits dans votre copie de travail est appelé *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, soit 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 prêt à livrer vos changements. Choisissez n'importe quel fichier et/ou dossier que vous voulez livrer, puis TortoiseSVN → Livrer....

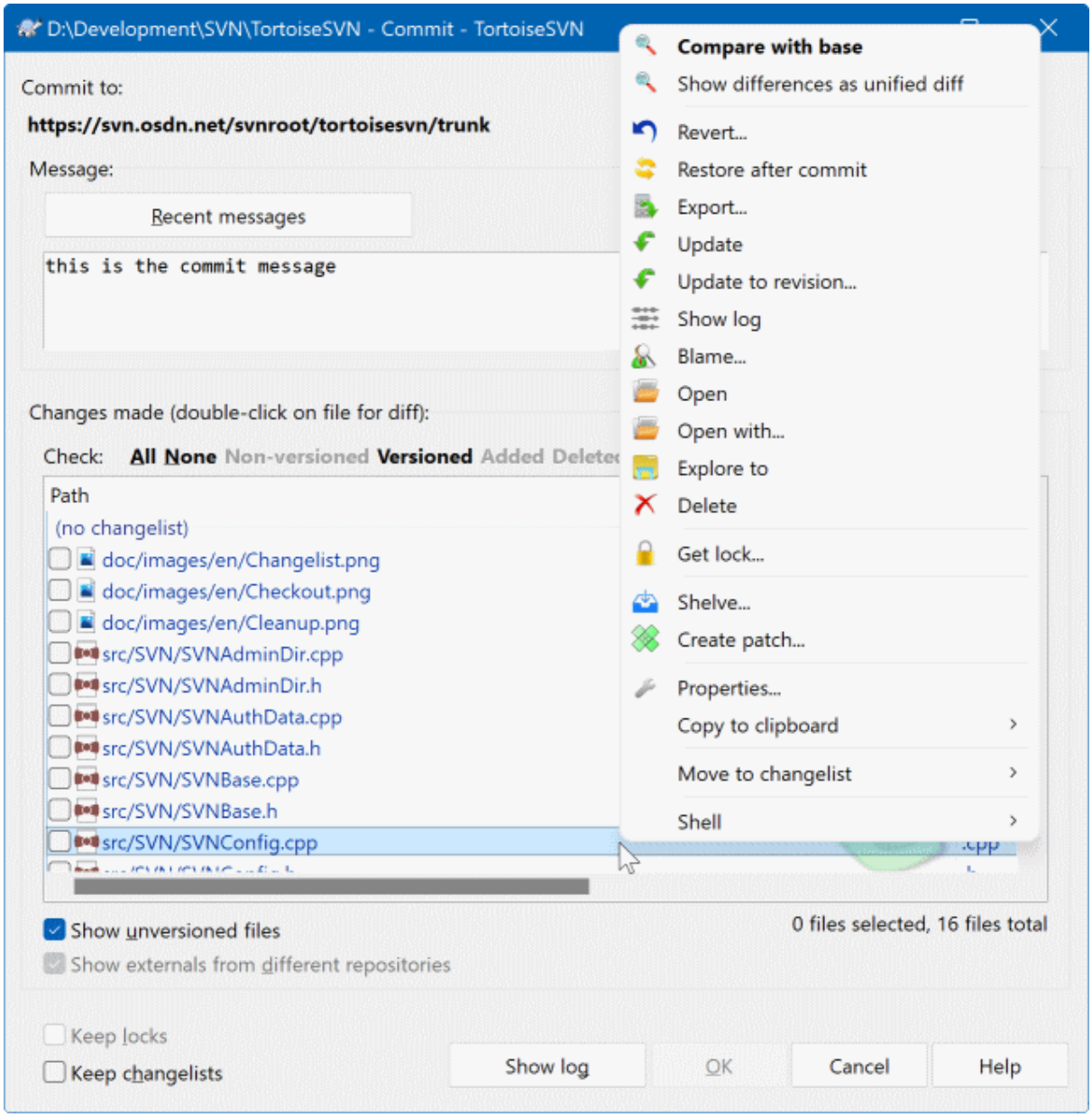


Figure 4.8. La boîte de dialogue Livrer

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 ce fichier. Si vous voulez inclure un fichier non versionné, cochez ce fichier pour l'ajouter à la livraison.

Pour cocher ou décocher rapidement des catégories de fichiers, comme tous les fichiers versionnés, ou tous les fichiers modifiés, cliquez sur le lien correspondant juste au-dessus de la liste affichée des éléments modifiés.

Pour des informations sur les couleurs et les éléments supoerposés en fonction de leur statut, veuillez consulter Section 4.7.3, « Statut local et distant ».

Les éléments qui ont été commutés vers un chemin de dépôt 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 sélectionnés. Quand vous livrez un dossier, la boîte de dialogue Livrer sélectionnera automatiquement les fichiers modifiés. Si vous oubliez un nouveau fichier que vous avez créé, livrer le dossier le trouvera de toute façon. Livrer d'un dossier ne veut *pas* dire que chaque fichier est marqué comme changé ; cela vous rend simplement la vie plus facile en faisant plus de travail pour vous.

Beaucoup de fichiers non versionnés dans la boîte de dialogue Livrer

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

- ajouter le fichier (ou un modèle avec des caractères de remplacement) à la liste de fichiers à 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.
- ajouter le fichier à la liste `svn:global-ignores` en utilisant TortoiseSVN → Ajouter à la liste des ignorés (récursivement). Cela affectera le répertoire dans lequel vous positionnez la propriété `svn:global-ignores`, ainsi que tous ses sous-répertoires.

Lire Section 4.14, « 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 comparaison pour afficher vos changements. Le menu contextuel vous donnera plus d'options, comme affiché sur la capture d'écran. Vous pouvez aussi faire glisser 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 cocher située à leur gauche. Pour les répertoires, vous pouvez utiliser **Maj**-sélectionner pour le faire récursivement.

Les colonnes affichées dans le panneau du bas sont personnalisables. Si vous faites un clic 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 en-têtes. Ces personnalisations sont conservées, donc vous verrez les mêmes en-têtes 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 4.31.1, « Configuration générale » pour plus d'informations sur l'édition du fichier de configuration de Subversion.

Avertissement en livrant dans une étiquette

Généralement, les livraisons se font dans le trunk ou dans une branche, mais pas dans une étiquette. Après tout, une étiquette est considérée comme fixe et ne devrait pas être modifiée.

Si vous essayez de faire une livraison dans l'URL d'une étiquette, TortoiseSVN affichera d'abord une boîte de dialogue de confirmation pour s'assurer que c'est bien ce que vous voulez faire, car la plupart du temps ces livraisons sont faites par erreur.

Cependant, cette vérification ne fonctionne que si la disposition du dépôt est une de celles recommandées, c'est-à-dire qu'elle utilise les noms `trunk`, `branches` et `tags` pour marquer les trois zones principales. Si votre disposition est différente, la détection de ce qui est une étiquette, une branche ou la branche principale (qu'on appelle aussi les `tests de classification`) peut se paramétrer dans la boîte de dialogue de configuration : Section 4.31.2, « Options du Graphe des Révisions »

Glisser-déposer

Vous pouvez glisser des fichiers dans la boîte de dialogue Livrer depuis un autre emplacement, tant que les copies de travail sont extraites du même dépôt. Par exemple, supposons que vous ayez 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 depuis les 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 dépôt.

Glisser-déplacer des fichiers depuis la liste au bas de la boîte de dialogue de livraison vers le champ d'édition du message de journal insèrera les chemins comme texte brut dans ce champ d'édition. Cela est utile si vous voulez écrire des messages de journal de livraison qui incluent les chemins qui sont affectés par la livraison.

Réparation des renommages externes

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

Réparation des copies externes

Si vous avez fait une copie de fichier sans utiliser la commande appropriée de Subversion, vous pouvez réparer l'erreur afin que le fichier copié ne perde pas son historique. Sélectionnez juste les deux fichiers (le fichier source et le fichier copié pas encore versionné) et utilisez Menu Contextuel → Réparer la copie pour associer les deux fichiers en tant que copie.

4.4.2. Listes de changements

La boîte de dialogue de livraison supporte la fonctionnalité de liste de changements de Subversion pour vous aider à regrouper les fichiers qui sont en lien les uns avec les autres. Pour en savoir plus sur cette fonctionnalité, voyez Section 4.8, « Listes de changements ».

4.4.3. Livrer uniquement des morceaux de fichiers

Parfois, vous ne voulez livrer qu'une partie des changements que vous avez apportés à un fichier. Ce genre de situation se produit généralement quand vous travaillez sur quelque chose, mais qu'une correction urgente doit être livrée, et qu'il se trouve que cette correction est dans le même fichier que celui sur lequel vous êtes en train de travailler.

Faites un clic droit sur le fichier et utilisez Menu contextuel → Restauration après la livraison. Cela créera une copie du fichier dans son état actuel. Ensuite vous pouvez éditer le fichier, par exemple dans un éditeur de texte, et

annuler tous les changements que vous ne voulez pas livrer maintenant. Après avoir sauvegardé ces changements, vous pouvez livrer le fichier.

Utiliser TortoiseMerge

Si vous utilisez TortoiseMerge pour éditer le fichier, vous pouvez soit éditer les changements comme vous avez l'habitude de le faire, soit marquer tous les changements que vous voulez inclure. Faites un clic droit sur un bloc modifié et utiliez Menu contextuel → Marquer ce changement pour inclure ce changement. Quand vous avez terminé, faites un clic droit et utilisez Menu contextuel → Conserver seulement les changements marqués, ce qui modifiera la vue de droite pour n'inclure que les changements que vous avez marqués auparavant et annuler les changements que vous n'avez pas marqués.

Une fois que la livraison est effectuée, la copie du fichier est restaurée automatiquement, et vous avez votre fichier avec toutes les modifications qui n'ont pas été livrées.

4.4.4. Exclure des éléments de la livraison

Il y a parfois des fichiers versionnés qui sont fréquemment modifiés mais que vous ne voulez vraiment pas livrer à chaque fois. Cela peut parfois indiquer un défaut dans votre processus de génération : pourquoi ces fichiers sont-ils versionnés ? Ne devriez-vous pas utiliser des fichiers templates ? Cependant, c'est parfois inévitable. Un cas typique est quand 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 quand seul l'horodatage a changé.

Pour vous aider à gérer ce genre de cas problématiques, nous avons prédéfini une liste de changements appelée `ignore-on-commit` (« 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.

4.4.5. 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 e-mails. Pour appliquer un style à un `texte`, utilisez `*texte*` pour le gras, `_texte_` pour souligner, et `^texte^` pour l'italique.

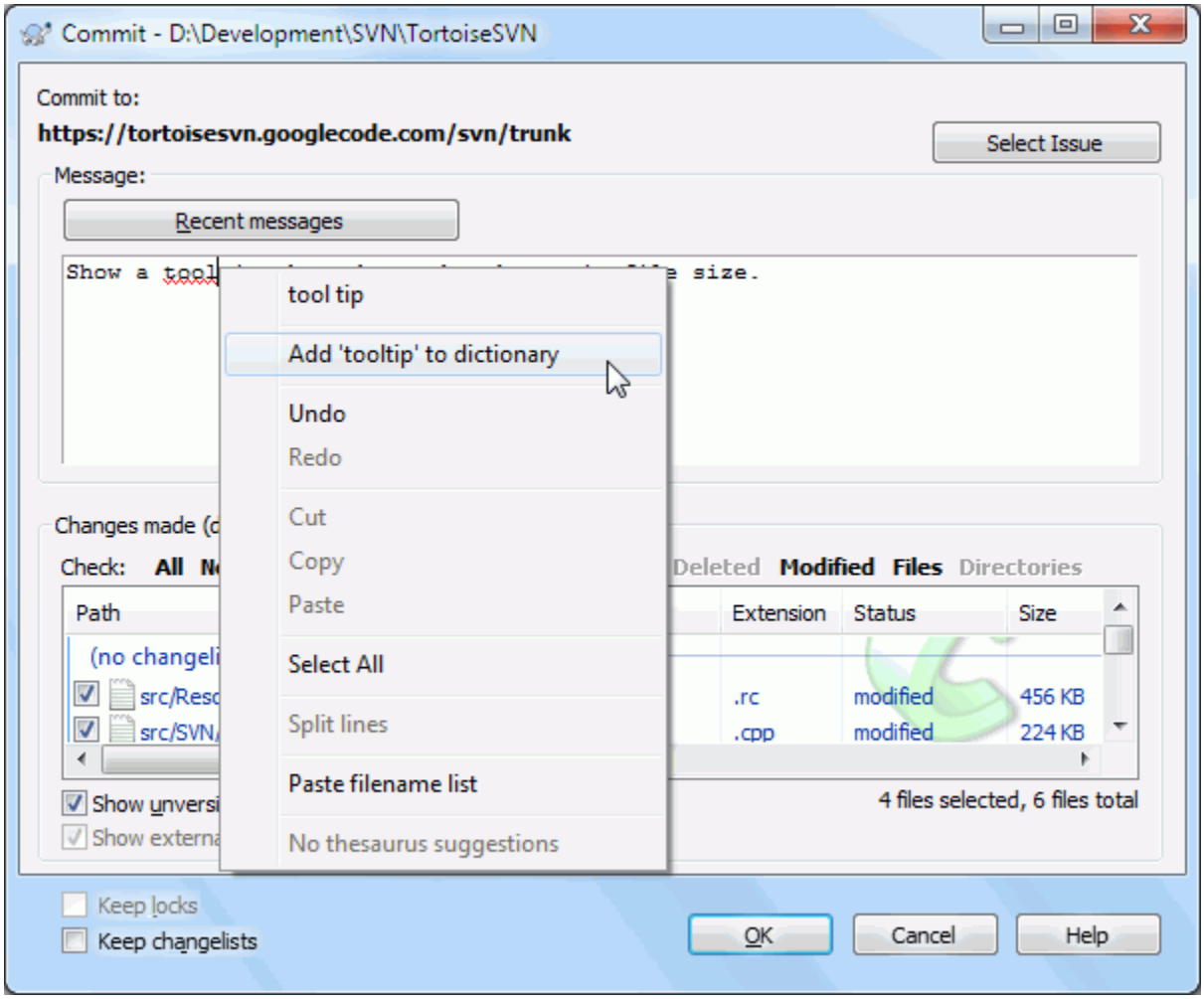


Figure 4.9. Le vérificateur d'orthographe de la boîte de dialogue Livrer

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 comprend aussi une fonctionnalité de complétion automatique de noms de fichier et de fonction. Elle utilise des expressions régulières pour extraire les noms de classe et de fonction des fichiers (texte) que vous livrez, ainsi que les noms de fichiers eux-mêmes. Si un mot que vous tapez correspond à quelque chose dans la liste (une fois que vous avez tapé au moins 3 caractères ou appuyé sur **Ctrl+Espace**), une liste déroulante apparaît pour vous permettre de sélectionner le nom complet. Les expressions régulières fournies avec TortoiseSVN sont contenues dans le dossier `bin` de l'installation de TortoiseSVN. Vous pouvez également définir vos propres expressions régulières et les stocker dans `%APPDATA%\TortoiseSVN\autolist.txt`. Naturellement, votre fichier autolist privé ne sera pas écrasé lors d'une mise à jour de votre installation de TortoiseSVN. Si vous n'êtes pas à l'aise avec les expressions régulières, jetez un œil à l'introduction à https://fr.wikipedia.org/wiki/Expression_r%C3%A9guli%C3%A8re. Vous pouvez également accéder à de la documentation en ligne et un tutoriel en anglais à https://www.regular-expressions.info/.

Écrire l'expression régulière parfaite peut s'avérer complexe. Pour vous aider à la trouver, il y a une boîte de dialogue de test qui vous permet de saisir une expression, puis de taper des noms de fichier pour la tester. Vous pouvez la démarrer depuis l'invite de commandes avec la commande `TortoiseProc.exe /command:autotexttest`.

La fenêtre de commentaire incorpore également un générateur de messages de livraison type. Ces messages type apparaissent dans la liste déroulante de complétion automatique une fois que vous tapez un raccourci de message

type, et on peut ensuite insérer l'intégralité du texte du message type en sélectionnant ce message type dans la liste déroulante de complétion automatique. Les messages type fournis avec TortoiseSVN sont situés dans le dossier `bin` de l'installation de TortoiseSVN. Vous pouvez aussi définir vos propres messages type et les stocker dans `%APPDATA%\TortoiseSVN\snippet.txt`. # est la marque de commentaire. On peut insérer un retour à la ligne en l'échappant de cette façon : `\n` et `\r`. Pour insérer un antislash, il faut l'échapper de cette façon : `\\`.

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 récents depuis la page Données sauvegardées de la fenêtre de configuration de TortoiseSVN. Vous pouvez également supprimer les messages individuellement depuis la boîte de dialogue Messages récents en utillisant la touche **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 champ texte.

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

Propriétés de dossier spéciales

Il y a plusieurs propriétés de dossier spéciales qui peuvent être utilisées pour vous aider à contrôler le formatage des commentaires de livraison et la langue utilisée par le module de vérificateur d'orthographe. Lisez Section 4.18, « 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 régulières, ajoutez simplement les références des bugs dans le commentaire. Pour en savoir plus : Section 4.29, « Intégration avec des systèmes de gestion de bug / gestion d'incidents ».

4.4.6. 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 de couleurs 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 couleurs par défaut, mais vous pouvez personnaliser ces couleurs en utilisant la boîte de dialogue de configuration. Lisez Section 4.31.1.5, « Configuration des couleurs de TortoiseSVN » pour plus d'informations.

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

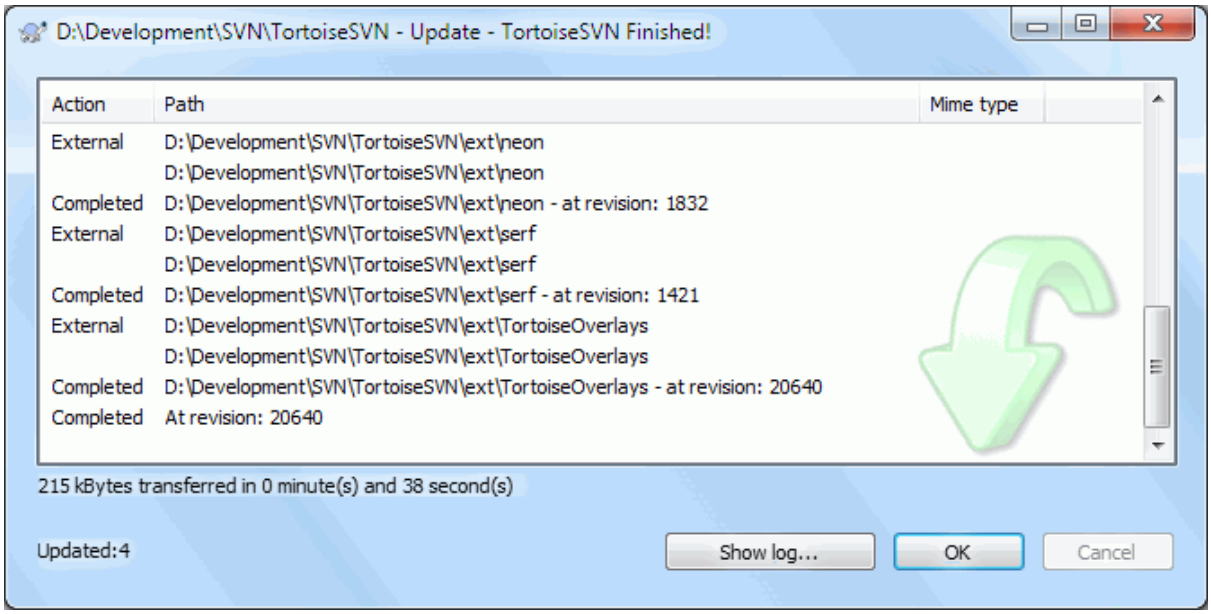


Figure 4.11. La boîte de dialogue de progression montrant une mise à jour terminée

Vous devriez vous assurer régulièrement que les changements des autres sont répercutés dans votre copie de travail. Le processus d'obtention des changements du serveur vers votre copie locale s'appelle la *mise à jour*. Vous pouvez mettre à jour des fichiers seuls, un jeu de fichiers sélectionnés, ou des hiérarchies entières de répertoires de manière récursive. Pour mettre à jour, sélectionnez les fichiers et/ou les répertoires de votre choix, faites un clic droit et choisissez TortoiseSVN → Mettre à jour dans le menu contextuel de l'explorateur. Une fenêtre apparaîtra montrant la progression de la mise à jour. Les changements faits par les autres seront fusionnés avec vos fichiers, en gardant les changements que vous pourriez avoir faits aux mêmes fichiers. Le dépôt *n'est pas* affecté par une mise à jour.

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 dépôt fusionnés avec vos changements locaux avec succès.

Rouge clair
: Changements du dépôt 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 dépôt.

C'est la combinaison de couleurs par défaut, mais vous pouvez personnaliser ces couleurs en utilisant la boîte de dialogue de configuration. Lisez Section 4.31.1.5, « 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**.

La commande Mettre à jour standard n'a pas d'options et se contente de mettre à jour votre copie de travail à la révision "HEAD" du dépôt, ce qui est le cas d'utilisation le plus fréquent. Si vous voulez un contrôle plus précis du processus de mise à jour, il faut utiliser TortoiseSVN → Mettre à jour à la révision... à la place. Cela vous permet de mettre à jour votre copie de travail à une révision spécifique, pas nécessairement la plus récente. Si votre copie de travail est à la révision 100, mais que vous voulez qu'elle reflète l'état qu'elle avait à la révision 50, alors mettez simplement à jour à la révision 50.

Dans la même boîte de dialogue vous pouvez également choisir la *profondeur* à laquelle mettre à jour le dossier courant. Les termes utilisés sont décrits dans Section 4.3.1, « Profondeur d'extraction ». La profondeur par défaut est Copie de travail, qui préserve le paramétrage de profondeur existant. Vous pouvez également positionner la profondeur à `persistante`, ce qui signifie que les mises à jour suivantes utiliseront cette nouvelle profondeur, autrement dit que cette profondeur devient la nouvelle profondeur par défaut.

Pour rendre plus facile d'inclure ou d'exclure des éléments spécifiques de l'extraction, cliquez sur le bouton Choisir les éléments.... Il ouvre une nouvelle boîte de dialogue où vous pouvez cocher tous les éléments dont vous voulez dans votre copie de travail et décocher tous les éléments dont vous ne voulez pas.

Vous pouvez également choisir d'ignorer ou non les projets externes dans la mise à jour (c'est-à-dire les projets référencés avec `svn:externals`).

Attention

Si vous mettez à jour un fichier ou un dossier à une révision spécifique, vous ne devriez pas le modifier. Vous obtiendrez des messages d'erreur de `péremption` si vous essayez de les livrer ! Si vous voulez annuler les changements d'un fichier et recommencer à partir d'une révision précédente, vous pouvez revenir à une révision précédente à partir de la boîte de dialogue de journal de révision. Jetez un coup d'œil à Section B.4, « Annuler des révisions dans le dépôt » pour avoir plus d'instructions et des méthodes alternatives.

Mettre à jour à la révision peut être utile de temps à autre pour voir à quoi ressemblait votre projet à un certain moment. Mais en général, ne mettre à jour que quelques fichiers à une révision précédente n'est pas une bonne idée car votre copie de travail est alors dans un état incohérent. Si le fichier que vous mettez à jour a été renommé,

vous pouvez même le voir disparaître de votre copie de travail parce qu'aucun fichier de ce nom n'existait dans la révision précédente. Remarquez également que l'élément aura une icône superposée normale (verte), ce qui le rend complètement indiscernable des autres fichiers qui sont à jour.

Si vous voulez simplement une copie locale d'une version ancienne d'un fichier, il vaut mieux utiliser la commande Menu contextuel → Enregistrer la révision sous... depuis la boîte de dialogue de journal de ce fichier.

Plusieurs fichiers/répertoires

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

4.6. Résoudre des conflits

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

conflits de fichier
: Un conflit de fichier se produit si deux (ou plus) développeurs changent les mêmes lignes d'un fichier.

conflits d'arborescence
: Un conflit d'arborescence se produit quand un développeur a déplacé/renommé/supprimé un fichier ou un dossier qu'un autre développeur a soit lui aussi déplacé/renommé/supprimé, soit simplement modifié.

4.6.1. Conflits de fichier

Un conflit de fichier se produit quand plusieurs développeurs changent les mêmes lignes d'un fichier. Comme Subversion ne connaît rien à votre projet, il laisse aux développeurs le soin de résoudre les conflits. La région en conflit dans un fichier texte est marquée comme ceci :

```
<<<<<<< nom de fichier
vos changements
=======
code récupéré du dépôt
>>>>>>> révision
```

De plus, pour chaque fichier en conflit, Subversion positionne trois fichiers supplémentaires 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 dépôt.

Vous pouvez soit lancer un outil externe de fusion ou un éditeur de conflits avec TortoiseSVN → Editer les conflits, soit utiliser l'éditeur de texte de votre choix pour résoudre le conflit manuellement. C'est à vous de décider à quoi le code doit ressembler, de faire les changements nécessaires et de sauvegarder le fichier. Utiliser un outil de fusion comme TortoiseMerge ou un autre outil populaire est généralement la solution la plus simple, car ils présentent généralement les fichiers concernés dans une vue à 3 colonnes et vous n'avez pas besoin de vous soucier

des marqueurs de conflit. Si vous utilisez un éditeur de texte, alors vous devrez chercher les lignes qui commencent par la chaîne de caractères <<<<<<<.

Exécutez ensuite la commande TortoiseSVN → Résolu... et livrez vos modifications au dépôt. Veuillez noter que la commande de résolution ne résout pas vraiment le conflit. Elle 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 dépôt, utilisez simplement la commande Revenir en arrière. Si vous voulez garder votre version et écraser la version du dépôt, utilisez la commande Résolu..., puis livrez 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 de propriété

Un conflit de propriété se produit lorsque plusieurs développeurs ont changé la même propriété. Comme avec le contenu des fichiers, la résolution du conflit ne peut être faite que par les développeurs.

Si un des changements doit remplacer l'autre, alors choisissez l'option Résoudre en utilisant la propriété locale ou Résoudre en utilisant la propriété distante. S'il faut fusionner les changements, alors sélectionnez Editer la propriété manuellement, déterminez quelle doit être la valeur de la propriété et marquez comme résolu.

4.6.3. Conflits d'arborescence

Un conflit dans l'arborescence se produit quand un développeur a déplacé/renommé/supprimé un fichier ou un dossier qu'un autre développeur a soit lui aussi déplacé/renommé/supprimé, soit simplement modifié. Plusieurs situations peuvent générer un conflit d'arborescence, et chacune a une solution différente.

Quand un fichier est marqué pour suppression par la commande Supprimer de Subversion, il est aussi supprimé du système de fichiers local. De ce fait, même s'il génère un conflit d'arborescence, il ne peut pas afficher une icône superposée indiquant son état en conflit, et on ne peut pas faire un clic droit dessus pour résoudre le conflit. Utilisez la boîte de dialogue Vérifier les modifications à la place pour pouvoir accéder à l'option Editer les conflits.

TortoiseSVN peut aider à trouver les endroits où fusionner les modifications, mais cela ne résoudra pas forcément tous les conflits. Souvenez-vous qu'après une mise à jour, la base de travail contiendra la révision de chaque élément tel qu'il était dans le dépôt 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 dépôt, pas à l'état dans lequel était l'élément avant que vous fassiez vos modifications.

4.6.3.1. Suppression locale, édition de la mise à jour

1. Le développeur A modifie `toto.c` et le livre sur le dépôt.
2. Dans le même temps, le développeur B a renommé `toto.c` en `titi.c` dans sa copie de travail, ou simplement supprimé `toto.c` ou son dossier parent.

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

- `toto.c` a été supprimé de la copie de travail, mais est marqué comme étant en conflit d'arborescence.
- Si le conflit provient d'un renommage plutôt que d'une suppression, alors `titi.c` est marqué comme ajouté, mais ne contient pas les modifications 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 `toto.c` dans le fichier renommé `titi.c`. Pour

de simples suppressions de fichier ou de répertoire, il peut choisir de garder l'élément avec les modifications du développeur A et annuler la suppression. Ou bien, en se contentant de marquer le conflit comme étant résolu sans rien faire d'autre, il peut annuler les modifications du développeur A.

La boîte de dialogue d'édition de conflit propose de fusionner les changements si elle parvient à trouver le fichier original qui a été renommé en `titi.c`. S'il y a plusieurs fichiers qui sont des points de départ potentiels, alors elle montre un bouton pour chacun de ces fichiers pour vous permettre de choisir le bon.

4.6.3.2. Modification locale, suppression dans la mise à jour

1. Le développeur A renomme `toto.c` en `titi.c` et livre les modifications dans le dépôt.

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

Ou dans le cas d'un renommage de dossier...

1. Le développeur A renomme le dossier parent `DossierToto` en `DossierTiti` et le livre dans le dépôt.

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

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

- `titi.c` est ajouté dans la copie de travail comme un fichier ordinaire.
- `toto.c` est marqué comme ayant été ajouté (avec historique) et présente un conflit d'arborescence.

Pour un conflit de dossier :

- `DossierTiti` est ajouté dans la copie de travail comme un dossier ordinaire.
- `DossierToto` est marqué comme ayant été ajouté (avec historique) et présente un conflit d'arborescence.

  `Foo.c` est marqué comme modifié.

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

Pour fusionner ses changements locaux avec la réorganisation, le développeur B doit d'abord déterminer le nouveau nom et le nouvel emplacement du fichier en conflit `toto.c` dans le dépôt. Cela peut se faire avec la boîte de dialogue de journal. Ensuite, il faut utiliser le bouton qui indique le bon fichier source pour résoudre le conflit.

Si le développeur B décide que les changements de A étaient faux, alors il doit choisir le bouton Marquer comme résolu dans la boîte de dialogue d'édition de conflit. Cela marque le fichier ou le dossier en conflit comme résolu, mais les changements du développeur A doivent être supprimés à la main. Là aussi, la boîte de dialogue de journal aide à retrouver ce qui a été déplacé.

4.6.3.3. Suppression locale, suppression dans la mise à jour

1. Le développeur A renomme `toto.c` en `titi.c` et livre les modifications dans le dépôt.

2. Le développeur B renomme `toto.c` en `tata.c`.

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

- `tata.c` est marqué comme ayant été ajouté avec historique.
- `titi.c` est ajouté dans la copie de travail avec le statut 'normal'.
- `toto.c` est marqué comme étant supprimé et présente un conflit d'arborescence.

Pour résoudre ce conflit, le développeur B doit déterminer le nouveau nom et le nouvel emplacement du fichier en conflit `toto.c` dans le dépôt. Cela peut se faire avec la boîte de dialogue de journal.

Ensuite, le développeur B doit décider lequel des nouveaux noms de `toto.c`garder - celui du développeur A ou celui issu du renommage qu'il a lui-même effectué.

Une fois que le développeur B a résolu manuellement le conflit, le conflit d'arborescence doit être marqué comme résolu grâce au bouton approprié dans la boîte de dialogue d'édition de conflit.

4.6.3.4. Copie locale manquante, modification dans la mise à jour

1. Le développeur A, qui travaille sur le tronc, modifie `toto.c` et le livre dans le dépôt.

2. Le développeur B, qui travaille sur une branche, renomme `toto.c` en `titi.c` et le livre dans le dépôt

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

- `titi.c` est déjà dans la copie de travail avec le statut 'normal'.

- `toto.c` est marqué comme manquant avec un conflit d'arborescence.

Pour résoudre ce conflit, le développeur B doit marquer le fichier comme résolu dans la boîte de dialogue d'édition de conflit, ce qui le retirera de la liste des conflits. Le développeur B doit alors décider s'il faut copier le fichier manquant `toto.c` depuis le dépôt, ou fusionner les changements du développeur A `toto.c` avec le fichier renommé `titi.c`, ou encore ignorer les changements en marquant le conflit comme résolu sans rien faire d'autre.

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

4.6.3.5. Modification locale, suppression dans la fusion

1. Le développeur A, qui travaille sur le tronc, renomme `toto.c` en `titi.c` et le livre dans le dépôt.

2. Le développeur B, qui travaille sur une branche, modifie le fichier `toto.c` et le livre dans le dépôt.

1. Le développeur A, qui travaille sur le tronc, renomme le dossier parent `DossierToto` en `DossierTiti` et le livre dans le dépôt.

2. Le développeur B, qui travaille sur une branche, modifie `toto.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 d'arborescence :

- `titi.c` est marqué comme étant ajouté.

- `toto.c` est marqué comme ayant été modifié et présente un conflit d'arborescence.

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

Pour fusionner ses changements locaux avec la réorganisation, le développeur B doit d'abord déterminer le nouveau nom et le nouvel emplacement du fichier en conflit `toto.c` dans le dépôt. Cela peut se faire avec la boîte de dialogue de journal pour la source de la fusion. L'éditeur de conflit ne montre que le journal de la copie de travail, car il ne sait pas quel chemin a été utilisé dans la fusion, donc vous devez trouver cela vous-même. Les changements doivent alors être fusionnés à la main, car il n'y a actuellement aucune façon d'automatiser ou même de simplifier le processus. Une fois que les changements ont été portés, le chemin en conflit est redondant et peut être supprimé.

Si le développeur B décide que les changements de A étaient faux, alors il doit utiliser le bouton Marquer comme résolu dans la boîte de dialogue d'éditeur de conflit. Cela marque le fichier ou le dossier en conflit comme résolu,

mais les changements du développeur A doivent être supprimés à la main. Là encore, la boîte de dialogue de journal pour la source de la fusion aide à retrouver ce qui a été déplacé.

4.6.3.6. Suppression locale, suppression dans la fusion

1. Le développeur A, qui travaille sur le tronc, renomme `toto.c` en `titi.c` et le livre dans le dépôt.

2. Le développeur B, qui travaille sur une branche, renomme `toto.c` en `tata.c` et le livre dans le dépôt.

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

- `tata.c` est marqué comme étant dans un état normal (non modifé).
- `titi.c` est marqué comme ajouté avec son historique.
- `toto.c` est marqué comme manquant et présente un conflit d'arborescence.

Pour résoudre ce conflit, le développeur B doit déterminer le nouveau nom et le nouvel emplacement du fichier en conflit `toto.c` dans le dépôt. Cela peut se faire avec la boîte de dialogue de journal pour la source de la fusion.

Ensuite, le développeur B doit décider lequel des nouveaux noms de `toto.c`garder - celui du développeur A ou celui issu du renommage qu'il a lui-même effectué.

Une fois que le développeur B a résolu manuellement le conflit, le conflit d'arborescence doit être marqué comme résolu grâce au bouton approprié dans la boîte de dialogue d'édition de conflit.

4.6.3.7. Autres conflits d'arborescence

Il y a d'autres cas qui sont considérés comme des conflits d'arborescence tout simplement parce que le conflit concerne un dossier plutôt qu'un fichier. Par exemple, si vous ajoutez un dossier qui porte le même nom à la fois dans le tronc et dans une branche puis que vous essayez de fusionner, vous obtiendrez un conflit d'arborescence. Si vous souhaitez conserver le dossier dans la fusion cible, marquez simplement le conflit comme résolu. Si vous souhaitez utiliser celui de la source, alors vous devez d'abord utiliser la commande SVN Supprimer sur celui de la cible, puis exécuter à nouveau la fusion. Si vous avez besoin de quelque chose de plus complexe, vous devrez le résoudre manuellement.

4.7. Obtenir des informations sur le statut

Pendant que vous travaillez sur votre copie de travail, vous avez souvent besoin de savoir quels fichiers vous avez changés, ajoutés, supprimés ou renommés, ou même quels fichiers ont été changés et livrés par les autres.

4.7.1. Icônes superposées

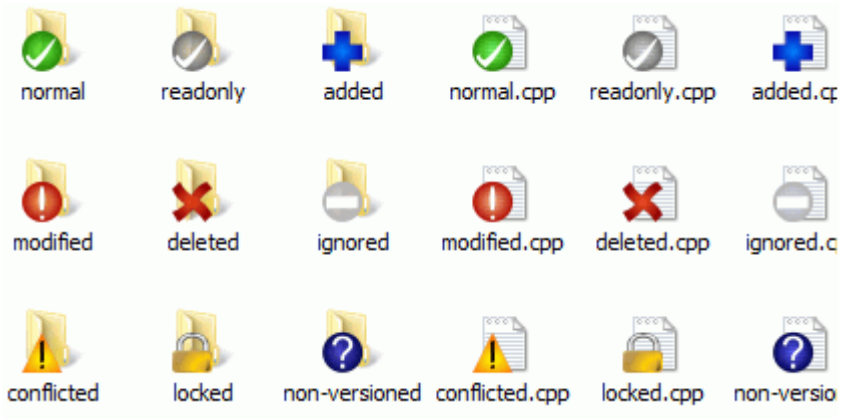


Figure 4.12. L'explorateur affichant des icônes superposées

Maintenant que vous avez extrait une copie de travail à partir d'un dépôt Subversion, vous pouvez voir que vos fichiers ont de nouvelles icônes dans l'explorateur Windows. C'est une des raisons pour lesquelles TortoiseSVN est si populaire. TortoiseSVN ajoute une icône superposée à chaque icône de fichier, par-dessus l'icône de fichier originale. Suivant le statut Subversion du fichier, l'icône superposée est différente.

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

Dès que vous commencez à éditer un fichier, le statut passe à `modifié` et l'icône superposée 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 doivent donc être livrés.

Si un `conflit` se produit lors d'une mise à jour, alors l'icône superposée 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 superposée 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 superposée 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 certains fichiers ou dossiers à l'intérieur du dossier ont été marqués comme *à supprimer* du contrôle de version, ou qu'un fichier versionné est absent d'un dossier.

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

Le signe « sens interdit » vous indique qu'un fichier ou un dossier a été marqué *à ignorer* par le contrôle de version. Cette icône superposée est optionnelle.

Cette icône est associée aux éléments qui ne sont pas versionnés, mais qui ne sont pas à ignorer. Cette icône superposée est optionnelle.

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

Maintenant qu'il y a de plus en plus de clients Tortoise dans les parages (TortoiseCVS, TortoiseHg, ...) la limite du nombre d'icônes devient vraiment problématique. Pour contourner cela, le projet TortoiseSVN a introduit un jeu d'icônes communes partagées, chargé dans une DLL, qui peut être utilisé par tous les clients Tortoise. Vérifiez avec le fournisseur de votre client si cela y a déjà été intégré :-)

Pour avoir une description de la correspondance entre les icônes superposées et les statuts de Subversion ainsi que d'autres détails techniques, lisez Section F.1, « Icônes superposées ».

4.7.2. État détaillé

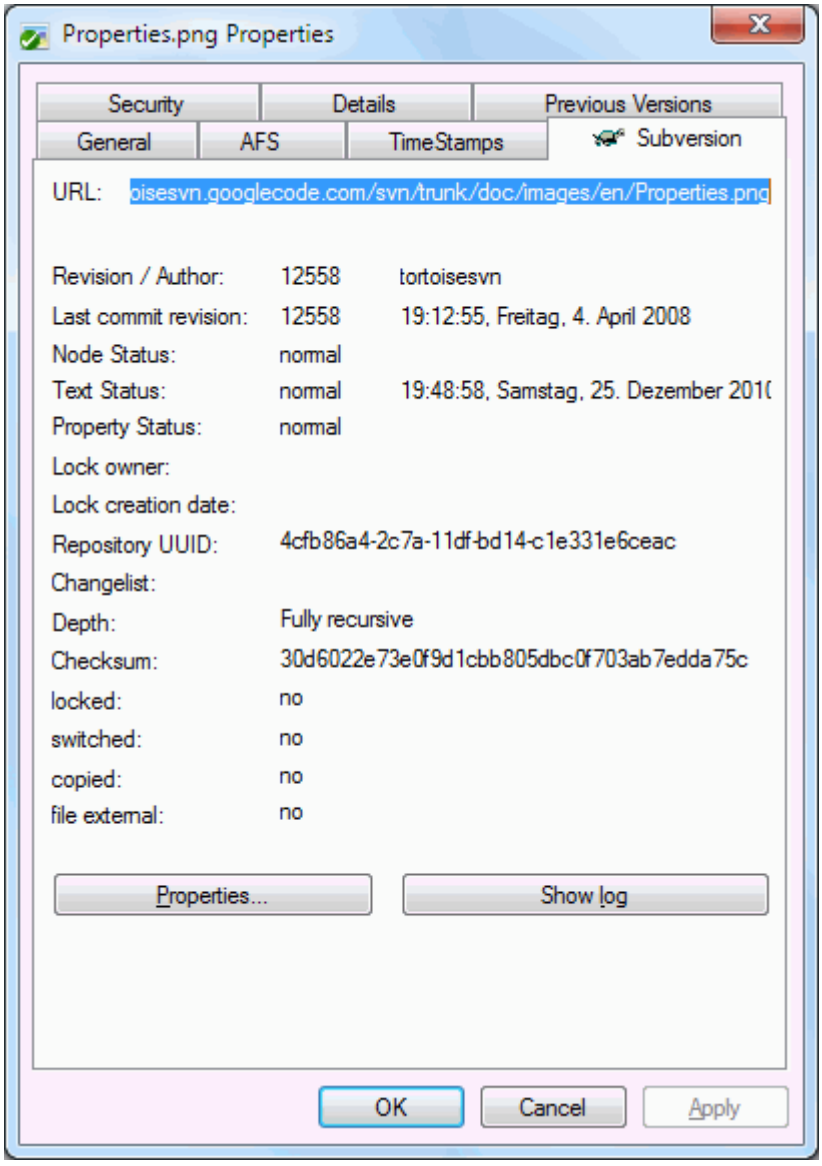


Figure 4.13. Page de propriétés de l'explorateur, onglet Subversion

Parfois vous voulez avoir des informations sur un fichier/répertoire qui soient plus détaillées que la simple icône superposée. Vous pouvez obtenir toutes les informations que Subversion fournit dans la boîte de dialogue de propriétés de l'explorateur. Sélectionnez simplement le fichier ou le répertoire et choisissez Menu Windows → Propriétés dans le menu contextuel (remarque : 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 propriétés pour les fichiers/dossiers sous le contrôle de Subversion, où vous verrez toutes les informations pertinentes concernant le fichier/répertoire sélectionné.

4.7.3. Statut local et distant

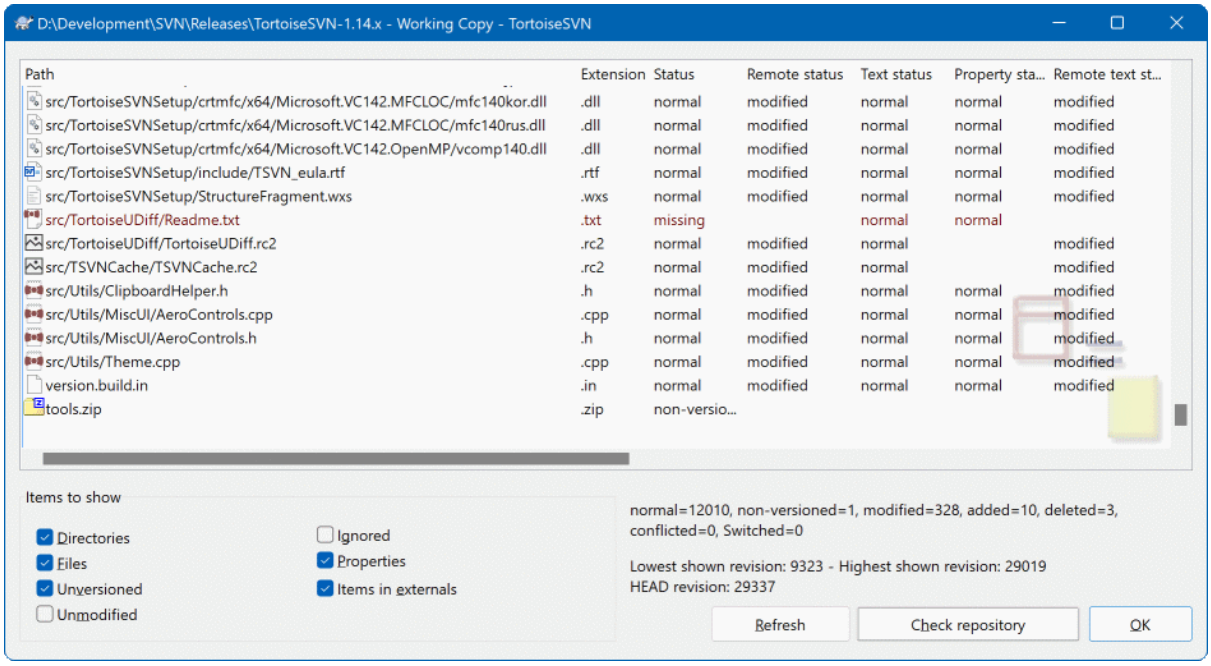


Figure 4.14. Vérifier les modifications

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

Si vous cliquez sur Vérifier le dépôt alors vous pouvez aussi chercher les changements dans le dépôt. 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 depuis le dépôt sans mettre à jour le dossier entier. Par défaut, le bouton Vérifier le dépôt ne récupère que le statut du dépôt à la profondeur de la copie de travail. Si vous voulez voir tous les fichiers et répertoires du dépôt, même ceux que vous n'avez pas extraits, vous devez cliquer sur le bouton Vérifier le dépôt en appuyant sur la touche **Maj**.

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

Bleu
: Éléments modifiés en local.

  Si des fichiers non modifiés se trouvent dans un répertoire qui a été déplacé, le statut affichera un signe + dans la colonne statut, et il sera également coloré en bleu.

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.

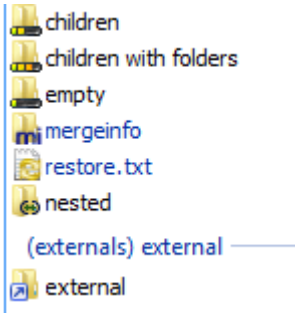
Vert
: Éléments modifiés en local et dans le dépôt. Les changements seront fusionnés lors de la mise à jour. Cela *peut* produire des conflits à la mise à jour.

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

Noir
: Éléments inchangés et non versionnés.

C'est la combinaison de couleurs par défaut, mais vous pouvez personnaliser ces couleurs en utilisant la boîte de dialogue de configuration. Lisez Section 4.31.1.5, « Configuration des couleurs de TortoiseSVN » pour plus d'informations.

Les icônes superposées sont également utilisées pour indiquer d'autres états. La capture d'écran ci-dessous montre toutes les icônes superposées qui peuvent être affichées si nécessaire.



Des icônes superposées sont affichées pour les états suivants :

Les éléments qui ont été basculés vers un chemin différent dans le dépôt sont également indiquée avec un marqueur (s). Vous avez peut-être basculé quelque chose pendant que vous travailliez sur une branche et vous avez oublié de rebasculer sur le trunk. Ceci est votre signal d'alarme ! Le menu contextuel vous permet de les rebasculer vers le chemin normal.

À partir du menu contextuel de la boîte de dialogue, vous pouvez afficher une comparaison des changements. Vérifiez les changements locaux que *vous* avez faits en utilisant Menu contextuel → Comparer avec la Base. Vérifiez les changements du dépôt 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 apparaîtra 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 faire glisser depuis ici vers une autre application comme un éditeur de texte ou IDE, ou vous pouvez enregistrer une copie en le faisant simplement glisser vers un dossier dans l'explorateur.

Les colonnes affichées sont personnalisables. Si vous faites un clic droit sur n'importe quel en-tête de colonne, un menu contextuel apparaîtra pour vous permettre de choisir les colonnes à afficher. Vous pouvez aussi changer la largeur de colonne en faisant glisser le bord des en-têtes. Ces personnalisations sont préservées, donc vous verrez les mêmes en-têtes 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 à plat 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 simplement 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 sont donc affichés comme fichiers manquants et non versionnés. Pour éviter de perdre l'historique, vous devez indiquer à Subversion le lien entre les deux. Sélectionnez le nom manquant (nom précédent) et le nouveau nom (non versionné) et utilisez Menu Contextuel → Réparer le déplacement pour associer les deux fichiers en tant que renommage.

Réparation des copies externes

Si vous avez fait une copie de fichier sans utiliser la commande appropriée de Subversion, vous pouvez réparer l'erreur afin que le fichier copié ne perde pas son historique. Sélectionnez juste les deux fichiers (le fichier source et le fichier copié pas encore versionné) et utilisez Menu Contextuel → Réparer la copie pour associer les deux fichiers en tant que copie.

4.7.4. Voir les différences

Souvent vous voulez regarder à l'intérieur de vos fichiers, pour examiner 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 différences, 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. La fonctionnalité de *liste de changements* vous aide à regrouper les fichiers, ce qui vous permet de voir plus facilement ce que vous faites. Bien sûr, ceci ne peut fonctionner que si les changements ne se chevauchent pas. Si deux tâches différentes affectent le même fichier, il devient impossible de séparer les changements.

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

Sélectionnez un ou plusieurs fichiers et utilisez Menu contextuel → Déplacer vers la liste de changements pour ajouter un élément à une liste de changements. Au départ, il n'y aura pas de liste de changements, donc la

première fois que vous ferez ceci vous devrez créer une nouvelle liste de changements. 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 changements, vous pouvez y glisser/déposer des éléments depuis une autre liste de changements, ou depuis l'explorateur Windows. L'avantage de le faire depuis l'explorateur est de permettre d'ajouter des éléments à une liste de changements avant qu'ils ne soient modifiés. Vous pourriez faire de même depuis la boîte de dialogue de vérification des modifications, mais seulement en affichant tous les fichiers non modifiés.

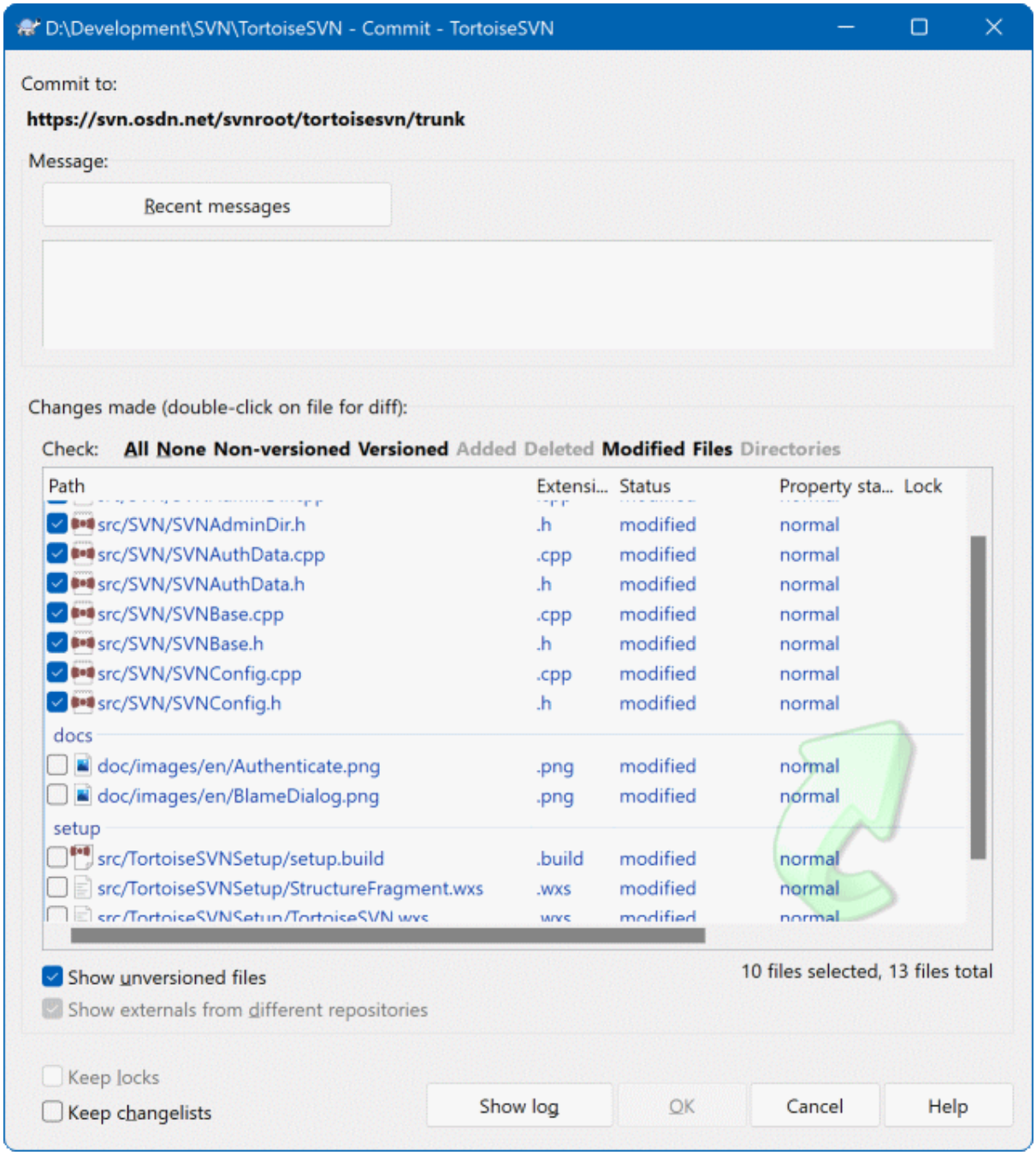


Figure 4.15. Boîte de dialogue de livraison avec des listes de changements

Dans la boîte de dialogue de livraison vous pouvez voir ces mêmes fichiers, regroupés par liste de changements. En plus de donner directement une indication visuelle sur les regroupements, les en-têtes des groupes peuvent également vous servir à sélectionner les fichiers à livrer.

TortoiseSVN se réserve pour son propre usage une liste de changements, appelée `ignore-on-commit`. Elle est utilisée pour marquer les fichiers versionnés que vous souhaitez rarement livrer même si vous les avez modifiés en local. Cette fonctionnalité est décrite dans Section 4.4.4, « Exclure des éléments de la livraison ».

Lorsqu'on livre des fichiers faisant partie d'une liste de changements, on s'attend généralement à ce qu'ils n'aient plus besoin d'en faire 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 changements en bas de la boîte de dialogue de livraison.

Astuce

Les listes de changements sont une fonctionnalité locale du client. Créer et supprimer ces listes n'aura d'effet ni sur le dépôt, ni sur les copies de travail des autres. C'est juste un moyen pratique d'organiser vos fichiers.

Avertissement

Prenez garde que si vous utilisez des listes de changements, les externes ne s'afficheront plus dans des groupes séparés. Une fois qu'il y a des listes de changements, les fichiers et les dossiers sont groupés par liste de changements et plus par statut externe ou pas.

4.9. Archivage

Il arrive parfois, et plus souvent que vous ne le souhaiteriez, que vous deviez arrêter ce que vous étiez en train de faire pour travailler sur autre chose. Par exemple, on a découvert un problème majeur qu'il faut résoudre immédiatement et vous devez interrompre le développement d'une nouvelle fonctionnalité. Si vous le pouvez, vous devriez livrer les changements que vous avez faits jusqu'à présent puis commencer à travailler sur le problème urgent, mais souvent ces changements ne compilent pas encore ou ne sont pas prêts à être livrés.

Si vous ne pouvez pas livrer vos changements locaux pour le moment, vous devez les mettre de côté pendant que vous travaillez sur le problème urgent. C'est exactement ce que la fonctionnalité d'*archivage* vous permet de faire : vous pouvez ranger vos changements locaux dans une archive, remettre votre copie de travail dans un état propre et travailler sur le problème. Une fois que vous en avez terminé avec le problème urgent et que vous avez livré ces changements, vous pouvez *désarchiver* votre travail archivé et reprendre votre travail sur la tâche précédente.

Deux nouvelles commandes ont été implémentées pour cela, une pour archiver et une pour désarchiver.

Pour archiver vos changements locaux, sélectionnez votre copie de travail et utilisez Menu contextuel → Archiver. La boîte de dialogue suivante vous permet de sélectionner les fichiers que vous voulez archiver et de leur donner un nom sous lequel vous voulez les ranger.

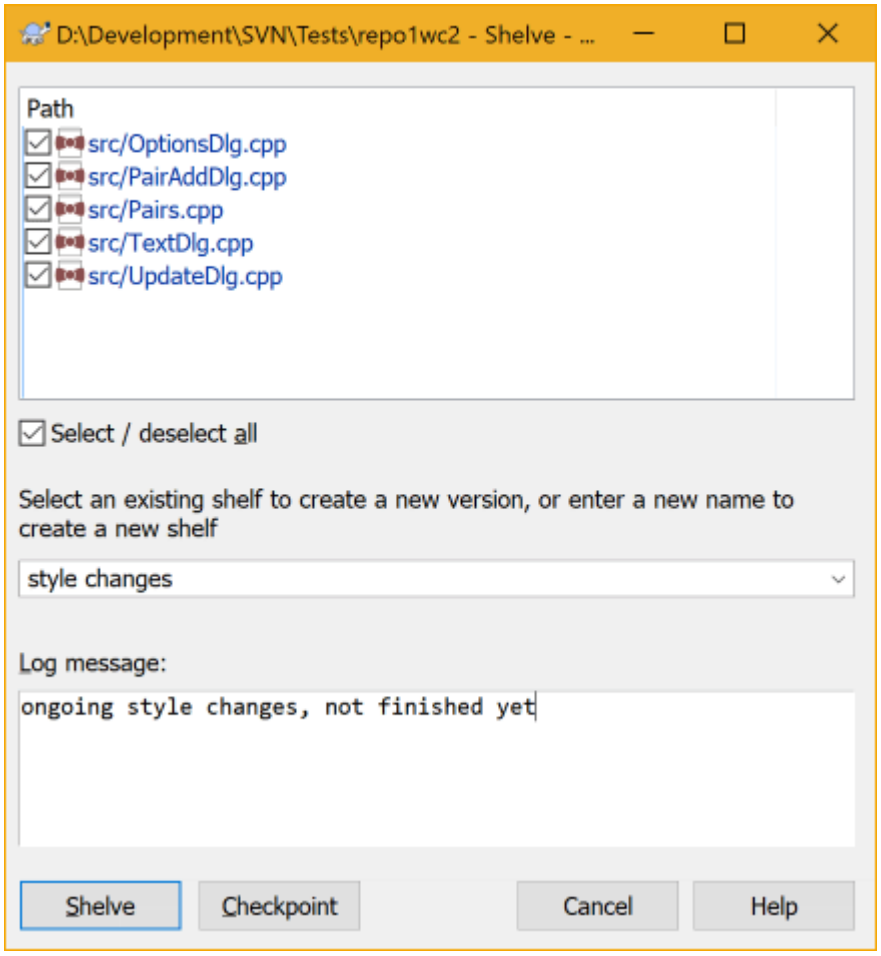


Figure 4.16. Boîte de dialogue d'archivage

Si vous sélectionnez une archive existante, alors une nouvelle version de cette archive est créée. Si vous donnez un nouveau nom, une nouvelle archive est créée pour les fichiers sélectionnés.

Si vous cliquez sur le bouton Archiver, l'archive est créée et les fichiers de votre copie de travail sont réinitialisés à un état propre. Si vous cliquez sur le bouton Point de contrôle, l'archive est créée mais vos modifications locales sont conservées.

Pour désarchiver vos changements, utilisez Menu contextuel → Désarchiver pour afficher la boîte de dialogue de désarchivage. Cette boîte de dialogue vous présente une liste de tous les éléments archivés. Sélectionnez l'élément archivé que vous voulez et la version à réappliquer à votre copie de travail et cliquer sur Appliquer.

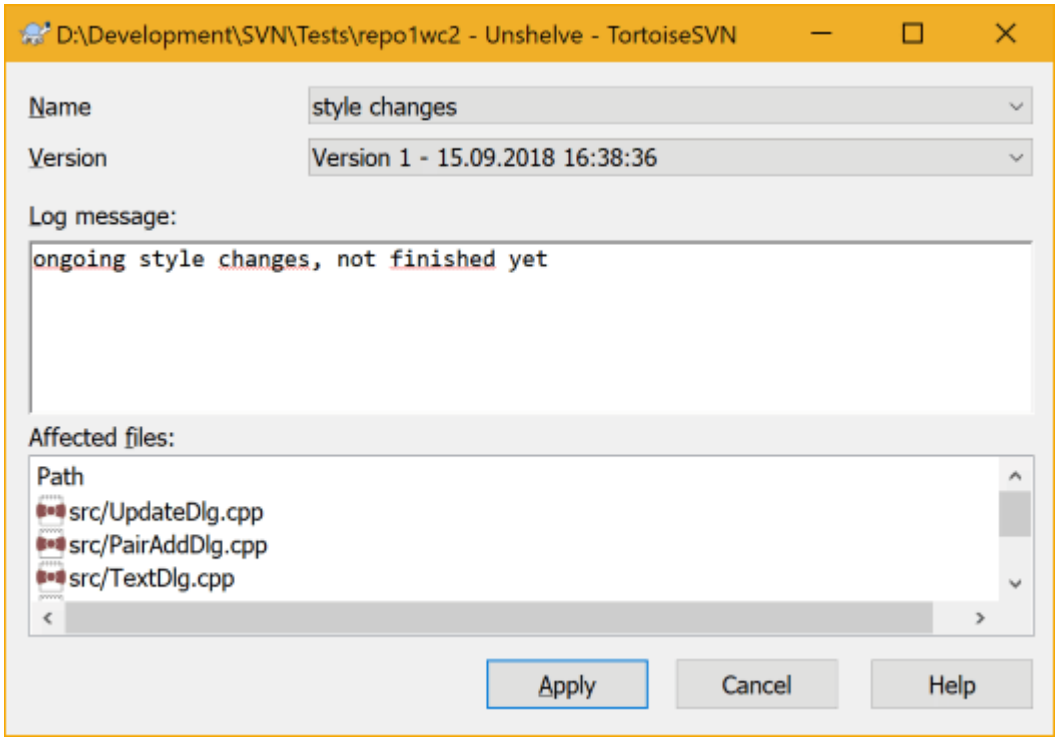


Figure 4.17. Boîte de dialogue de désarchivage

Astuce

Les archives sont une fonctionnalité purement locale du client. Créer et supprimer des archives n'aura pas d'effet sur le dépôt, ni sur la copie de travail de quelqu'un d'autre.

Expérimental

La fonctionnalité d'archivage est toujours à l'état `expérimental`.

Cela signifie que même si l'archivage fonctionne comme annoncé, il est dans un état où il fait encore l'objet de nombreux travaux d'amélioration. Cela signifie également qu'il n'y a aucune garantie que les archives que vous créez présentent une compatibilité ascendante, et que les versions futures ne seront peut-être pas capables de les utiliser. Et bien sûr, l'IHM peut également être amenée à changer dans des versions futures pour intégrer des fonctionnalités et comportements nouveaux.

4.10. La boîte de dialogue de journal de révision

À 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 de 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 des 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 depuis 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 dossiers qui ont été modifiés dans 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.

4.10.1. Appeler la boîte de dialogue de journal de révision

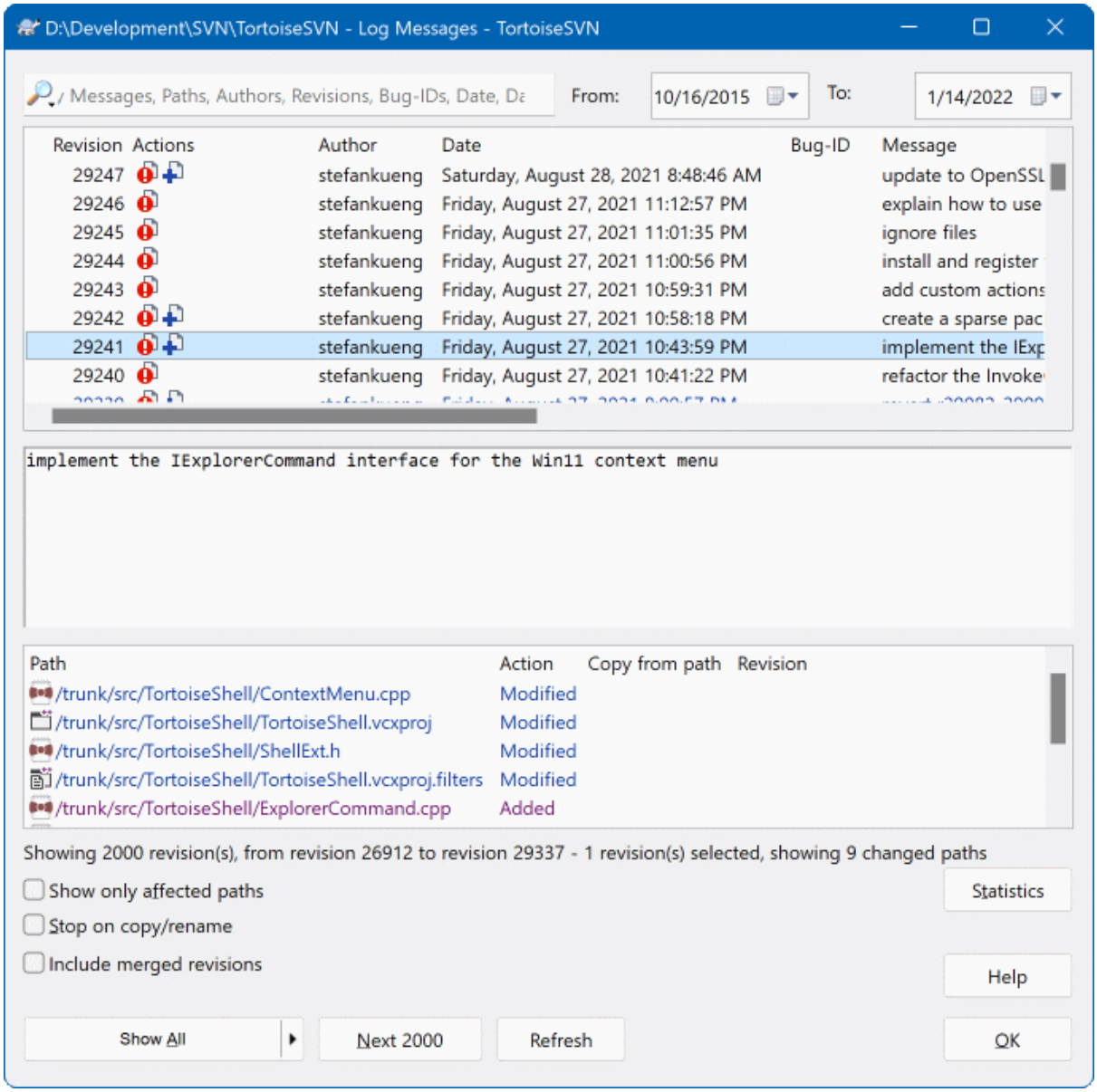


Figure 4.18. La boîte de dialogue de journal de révision

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 propriétés
- À partir de la boîte de dialogue de progression après la fin d'une mise à jour. Dans ce cas, la boîte de dialogue de journal ne montre que les révisions qui ont été modifiées depuis votre dernière mise à jour
- Depuis l'explorateur de dépôt

Si le dépôt n'est pas joignable vous verrez une fenêtre Travailler hors ligne ?, décrite dans Section 4.10.10, « Mode hors ligne ».

4.10.2. Actions dans le journal de révision

Le panneau supérieur a une colonne Actions contenant des icônes qui résument ce qui a été fait dans cette révision. Il y a quatre icônes différentes, chacune est affichée dans sa propre colonne.

Si un fichier ou un répertoire a été modifié dans la révision, l'icône *modifié* est affichée dans la première colonne.

Si un fichier ou un répertoire a été ajouté dans la révision, l'icône *ajouté* est affichée dans la deuxième colonne.

Si un fichier ou un répertoire a été supprimé dans la révision, l'icône *supprimé* s'affiche dans la troisième colonne.

Si un fichier ou un répertoire a été remplacé dans la révision, l'icône *remplacé* est affichée dans la quatrième colonne.

Si un fichier ou un répertoire a été déplacé ou renommé dans la révision, l'icône *déplacé* est affichée dans la quatrième colonne.

Si un fichier ou un répertoire a été remplacé dans la révision au moyen d'un déplacement ou d'un renommage, l'icône *déplacé remplacé* est affichée dans la quatrième colonne.

Si un fichier ou un répertoire a été fusionné dans la révision, l'icône *fusionné* est affichée dans la quatrième colonne.

Si un fichier ou un répertoire a fait l'objet d'une fusion inversée dans la révision, l'icône *fusionné inversé* est affichée dans la quatrième colonne.

4.10.3. Obtenir des informations supplémentaires

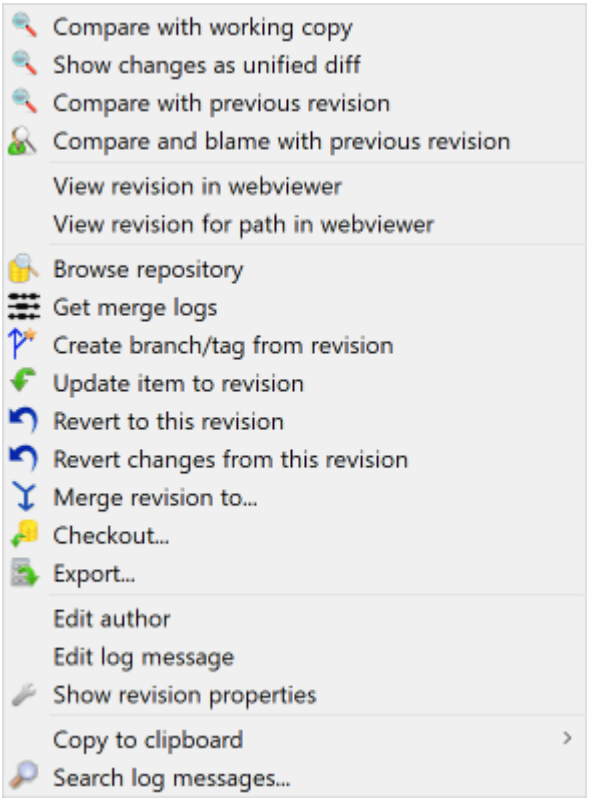


Figure 4.19. Le panneau supérieur de la boîte de dialogue de journal de révision avec le menu contextuel

Le panneau supérieur de la boîte de dialogue de journal a un menu contextuel qui vous permet d'accéder à beaucoup d'informations supplémentaires. Certains de ces éléments de menu n'apparaissent que quand on affiche le journal d'un fichier, et d'autres que quand on affiche le journal d'un dossier.

Comparaison avec la copie de travail
: Compare la révision sélectionnée avec votre copie de travail. L'outil de comparaison 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 avec la BASE de travail et annoter
: Annote la révision sélectionnée et le fichier dans votre BASE de travail et compare les rapports d'annotation avec un outil de comparaison visuel. Lisez Section 4.24.2, « Annoter les différences » pour plus de détails. (fichiers uniquement)

Voir les différences en mode diff unifié
: Affiche 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. C'est plus difficile à lire qu'une comparaison visuelle de fichiers, mais cela vous présentera tous les changements dans un format compact.

  Si vous gardez la touche **Maj** appuyée en cliquant sur l'élément de menu, une boîte de dialogue s'affiche où vous pouvez paramétrer les options pour le mode diff unifié. Ces options incluent la possibilité d'ignorer les changements de fins de ligne et de caractères d'espacement.

Comparer avec la révision précédente
: Compare la révision sélectionnée avec la révision précédente. Cette fonctionnalité est similaire à la comparaison avec votre copie de travail. Pour les dossiers, cette option montrera d'abord la boîte de dialogue des fichiers modifiés, vous permettant de sélectionner les fichiers à comparer.

Comparer avec la révision précédente et annoter
: Affiche la boîte de dialogue des fichiers modifiés pour vous permettre de sélectionner les fichiers. Annote la révision sélectionnée et la révision précédente, et compare les résultats en utilisant un outil de comparaison visuel. (dossiers uniquement)

Enregistrer la révision sous...
: Enregistre la révision sélectionnée dans un fichier pour conserver une ancienne version de ce fichier. (fichiers uniquement)

Ouvrir / Ouvrir avec...
: Ouvre le fichier sélectionné, soit grâce à l'éditeur par défaut pour ce type de fichier, soit grâce à un programme de votre choix. (fichiers uniquement)

Annoter...
: Annoter le fichier jusqu'à la révision sélectionnée. (Fichiers uniquement)

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

Créer une branche/étiquette depuis une révision
: Crée 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 que vous avez déjà livré des modifications qui n'étaient pas censées faire partie de cette version.

Mettre à jour l'élément à la révision
: Met 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 dépôt 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 plutôt Revenir à cette révision.

Revenir à cette révision
: Revient à une révision antérieure. Si vous avez fait plusieurs changements 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 changements sont annulés dans votre copie de travail : cette opération n'affecte donc *pas* le dépôt tant que vous n'avez pas livré ces changements. Notez que cela annulera *toutes* les modifications faites après la révision sélectionnée, en remplaçant le fichier/dossier avec cette révision antérieure.

  Si votre copie de travail est à l'état non modifiée, après avoir fait cette action elle sera marquée comme étant modifiée. Si vous avez déjà des changements en local, cette commande fusionnera les *annulations* de changements dans votre copie de travail.

  Ce qui se passe sous le capot est que Subversion procède à une fusion inverse de tous les changements effectués après la révision sélectionnée, annulant ainsi l'effet de ces livraisons précédentes.

  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 → Revenir en arrière depuis l'explorateur Windows, ce qui aura pour effet d'abandonner les modifications locales effectuées par cette action de fusion inverse.

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

Annuler les modifications de cette révision
: Annule 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 dépôt ! Notez que cela n'annulera que les

modifications de cette révision ; 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 est à l'état non modifiée, après avoir fait cette action elle sera marquée comme étant modifiée. Si vous avez déjà des changements en local, cette commande fusionnera les *annulations* de changements dans votre copie de travail.

Ce qui se passe sous le capot est que Subversion procède à une fusion inverse de cette révision particulière, annulant ainsi l'effet de cette livraison précédente.

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

Fusionner la révision avec...
: Fusionne la(les) révision(s) sélectionnée(s) dans une copie de travail séparée. Une boîte de dialogue de sélection de dossier vous permet de choisir la copie de travail 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 de travail non modifiée, 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 vers une autre.

Extraire...
: Fait une extraction propre de la révision sélectionnée. Cette commande affiche une boîte de dialogue pour vous demander de confirmer l'URL, le numéro de la révision, et de sélectionner l'emplacement de l'extraction.

Exporter...
: Exporte le fichier/répertoire sélectionné à la révision sélectionnée. Cette commande affiche une boîte de dialogue pour vous demander de confirmer l'URL, la révision, et de sélectionner l'emplacement de l'exportation.

Modifier l'auteur/le commentaire de révision
: Édite le commentaire de révision ou l'auteur attaché à une précédente livraison. Lisez Section 4.10.7, « Changer le commentaire et l'auteur » pour découvrir comment cela fonctionne.

Montrer les propriétés de la révision
: Affiche et édite toutes les propriétés d'une révision, outre le commentaire de révision et l'auteur. Voir Section 4.10.7, « Changer le commentaire et l'auteur ».

Ajouter au presse-papier
: Copie les commentaires des 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...
: Recherche 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.

Créer une revue avec le collaborateur de code...
: Cette commande n'est disponible que si l'outil Collaborateur de Code de SmartBear est installé. Quand elle est lancée pour la première fois, une boîte de dialogue demande à l'utilisateur de saisir les informations d'identification pour le Collaborateur de Code et pour SVN. Une fois que les paramètres sont enregistrés, la boîte de dialogue de paramètres n'est plus affichée quand la commande est lancée, sauf si l'utilisateur appuie sur **Ctrl** en même temps qu'il sélectionne l'élément de menu. La configuration et la(les) révision(s) choisie(s) sont utilisées pour lancer l'interface graphique du client du Collaborateur de Code, qui crée une nouvelle revue avec les révisions sélectionnées.

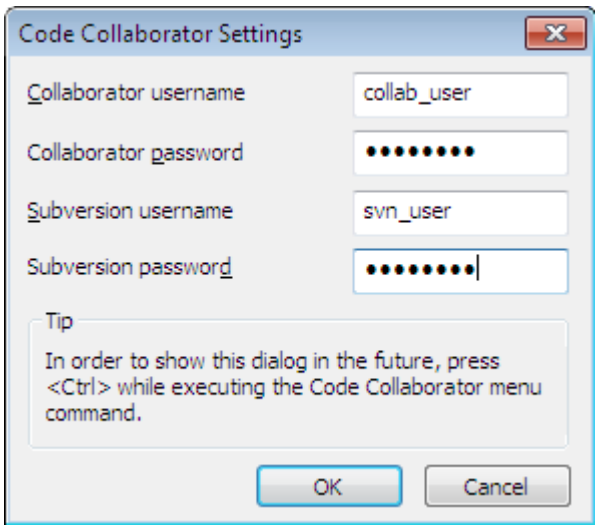


Figure 4.20. La boîte de dialogue de paramètres du Collaborateur de Code

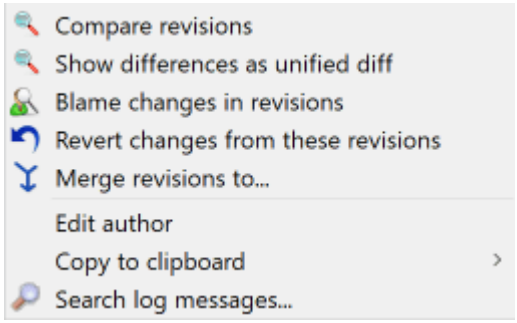


Figure 4.21. Menu contextuel du panneau supérieur avec 2 révisions sélectionnées

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 les révisions
: Compare les deux révisions choisies en utilisant un outil de comparaison visuel. L'outil de comparaison 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, consultez Section 4.11.3, « Comparer des dossiers ».

Annoter les révisions
: Annote les deux révisions et compare les rapports d'annotation en utilisant un outil de comparaison visuel. Lisez Section 4.24.2, « Annoter les différences » pour plus de détails.

Voir les différences en mode diff unifié
: Affiche 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
: Copie les commentaires de révision dans le presse-papiers comme décrit ci-dessus.

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

Si vous sélectionnez au moins deux révisions (en utilisant les modificateurs habituels **Ctrl** ou **Shift**), le menu contextuel inclura 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.

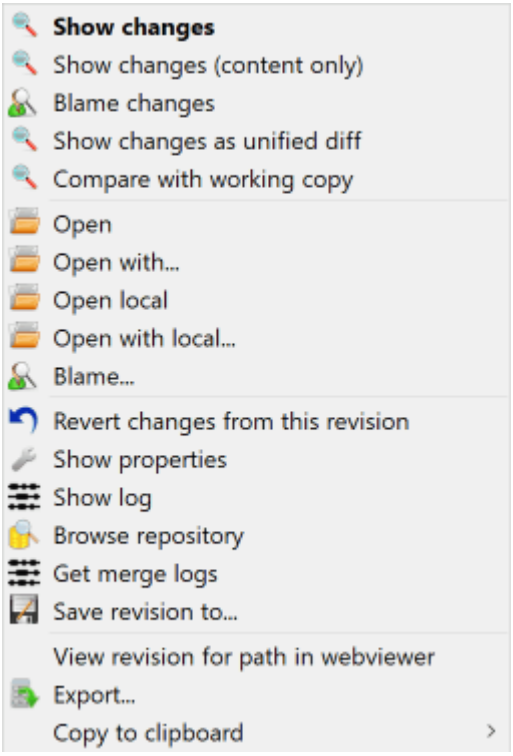


Figure 4.22. Le panneau inférieur de la boîte de dialogue de journal avec le menu contextuel

Le panneau inférieur de la boîte de dialogue de journal a aussi un menu contextuel qui vous permet de

Afficher les changements
: Affiche les changements apportés à la révision sélectionnée sur le fichier sélectionné.

Annoter les modifications
: Annote la révision sélectionnée et la révision précédente du fichier choisi et compare les rapports d'annotation en utilisant un outil de différenciation visuel. Lisez Section 4.24.2, « Annoter les différences » pour plus de détails.

Voir les différences en mode diff unifié
: Montre les modifications en tant que fichier de différences unifiées. Ce menu contextuel n'est disponible que 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...
: Ouvre la fenêtre d'annotation, vous permettant d'annoter jusqu'à la révision sélectionnée.

Annuler les modifications de cette révision
: Annule les changements effectués au fichier sélectionné dans cette révision.

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

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

Obtenir les journaux de fusion
: Montre les commentaires de révision pour le seul fichier sélectionné, y compris les modifications fusionnées. Plus de détail dans Section 4.10.6, « Fonctionnalités de suivi de fusion ».

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

Exporter...
: Exporte les éléments sélectionnés dans cette révision vers un dossier, en préservant la hiérarchie des fichiers.

Quand plusieurs fichiers sont sélectionnés dans le panneau inférieur de la boîte de dialogue de journal, le menu contextuel devient comme suit :

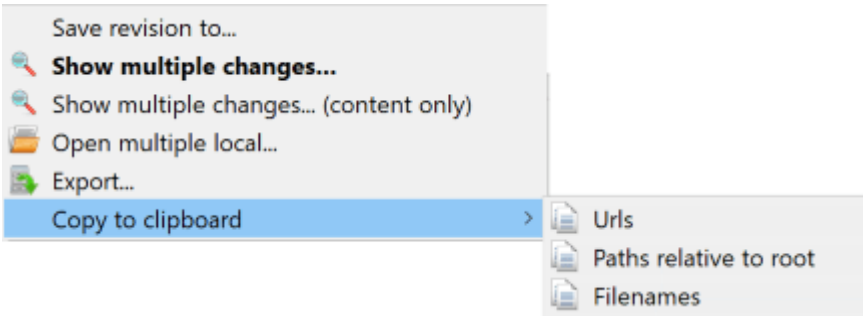


Figure 4.23. Le panneau inférieur de la boîte de dialogue de journal avec le menu contextuel quand plusieurs fichiers sont sélectionnés.

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

Afficher les modifications multiples...
: Affiche les changements apportés à la révision sélectionnée pour les fichiers sélectionnés. Remarquez que la fonctionnalité Afficher les changements est appelée plusieurs fois, ce qui peut lancer plusieurs instances de l'outil de visualisation de différences paramétré, ou simplement ajouter un nouvel onglet de comparaison dans votre outil de visualisation de différences. Si vous avez sélectionné plus de 15 fichiers, il vous sera demandé de confirmer votre action.

Ouvrir de multiples locaux...
: Ceci ouvrira les fichiers locaux de la copie de travail qui correspondent aux fichiers sélectionnés, avec l'application enregistrée pour leur extension. [Le comportement est celui que vous obtiendriez en double-cliquant sur le(s) fichier(s) de la copie de travail dans l'explorateur Windows.] Suivant l'association de l'extension de fichier à une application et les capacités de l'application, cette opération peut être lente. Dans le pire des cas, Windows peut lancer une nouvelle instance de l'application pour chacun des fichiers sélectionnés.

  Si vous gardez appuyé **Ctrl** en appelant cette commande, les fichiers de la copie de travail sont toujours chargés dans Visual Studio. Ceci ne fonctionne que quand les conditions suivantes sont réunies : Visual Studio doit s'exécuter dans le même contexte utilisateur et avoir le même niveau de sécurité de processus [exécuté en tant qu'administrateur ou non] que TortoiseProc.exe. Il peut être souhaitable d'avoir ouvert la solution contenant les fichiers modifiés, mais ce n'est pas absolument nécessaire. Seuls les fichiers existants sur le disque avec une des extensions [.cpp, .h, .cs, .rc, .resx, .xaml, .js, .html, .htm, .asp, .aspx, .php, .css et.xml] seront chargés. Il est possible d'ouvrir un maximum de 100 fichiers à la fois dans Visual Studio, et les fichiers sont toujours chargés dans de nouveaux onglets dans l'instance ouverte courante de Visual Studio. L'avantage de revoir les changements de code dans Visual Studio tient au fait que vous pouvez alors utiliser les outils de navigation de code, recherche de référence, analyse de code statique et autres intégrés à Visual Studio.

Exporter...
: Exporte les fichiers ou le dossier sélectionné à la révision sélectionnée. Une boîte de dialogue s'affiche pour vous permettre de confirmer l'UTL et la révision, et de sélectionner un emplacement pour l'exportation.

Astuce

Vous avez peut-être remarqué que nous parlons parfois de changements et parfois de différences. Quelle est la différence entre les deux ?

Subversion utilise des numéros de révision pour 2 choses différentes. Une révision représente généralement l'état du dépôt à un instant donné, mais elle peut également être utilisée pour représenter l'ensemble de changements qui a créé cette révision, par exemple« Fait dans la r1234 » signifie que les changements livrés dans la r1234 implémentent la fonctionnalité X. Pour indiquer plus clairement quel est le sens 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 *changements* faits dans cette révision. Dans Subversion, cette opération correspond à la commande `diff -r N-1:N` ou `diff -c N`.

Le panneau inférieur affiche les fichiers modifiés dans toutes les révisions sélectionnées. Le menu contextuel propose donc toujours d'afficher les *changements*.

4.10.4. Obtenir plus de commentaires

La boîte de dialogue de journal ne montre pas toujours toutes les modifications faites pour plusieurs raisons :

- Pour un grand dépôt, il peut y avoir des centaines ou même des milliers de modifications et les récupérer toutes pourrait prendre longtemps. Normalement vous n'êtes intéressé que par les modifications les plus récentes. Par défaut, le nombre de commentaires récupérés est limité à 100, mais vous pouvez changer cette valeur dans TortoiseSVN → Configuration (Section 4.31.1.2, « Réglages des boîtes de dialogue TortoiseSVN 1 »),

- Quand la case Arrêt à la copie/renommage est cochée, Voir le journal s'arrêtera au moment où le fichier ou le dossier sélectionné a été copiés depuis un autre emplacement dans le dépôt. Cela peut être utile en regardant les branches (ou les étiquettes) puisque la commande s'arrête à la racine de cette branche et donne une indication rapide des modifications faites dans cette branche seulement.

  Vous voudrez généralement 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.

Pour actualiser la dernière révision dans le cas où d'autres livraisons auraient eu lieu pendant que la boîte de dialogue du journal était ouverte, pressez la touche **F5**.

Pour actualiser le cache du journal, appuyez sur **Ctrl-F5**.

4.10.5. Révision courante de la copie de travail

La fenêtre de commentaires affichant les commentaires depuis HEAD et non depuis 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 commentaire de livraison correspondant à la révision de votre copie de travail est affiché en gras.

Quand vous affichez le journal pour un répertoire, la révision mise en évidence est la plus élevée trouvée dans ce répertoire, ce qui nécessite un parcours de la copie de travail. Ce parcours est effectué dans un autre thread pour ne pas ralentir l'affichage du journal, mais en conséquence la mise en évidence de la révision pour les répertoires peut ne pas apparaître immédiatement.

4.10.6. Fonctionnalités de suivi de fusion

Le logiciel Subversion en version 1.5 et ultérieure garde une trace des fusions à l'aide de propriétés. Cela nous permet d'obtenir un historique plus détaillé des changements fusionnés. Par exemple, si vous développez une nouvelle fonctionnalité dans une branche, puis que vous fusionnez cette branche dans le trunk, le développement de la fonctionnalité s'affichera dans le journal du trunk sous la forme d'une livraison unique pour la fusion, alors qu'il a pu y avoir 1000 livraisons pendant le développement dans la branche.

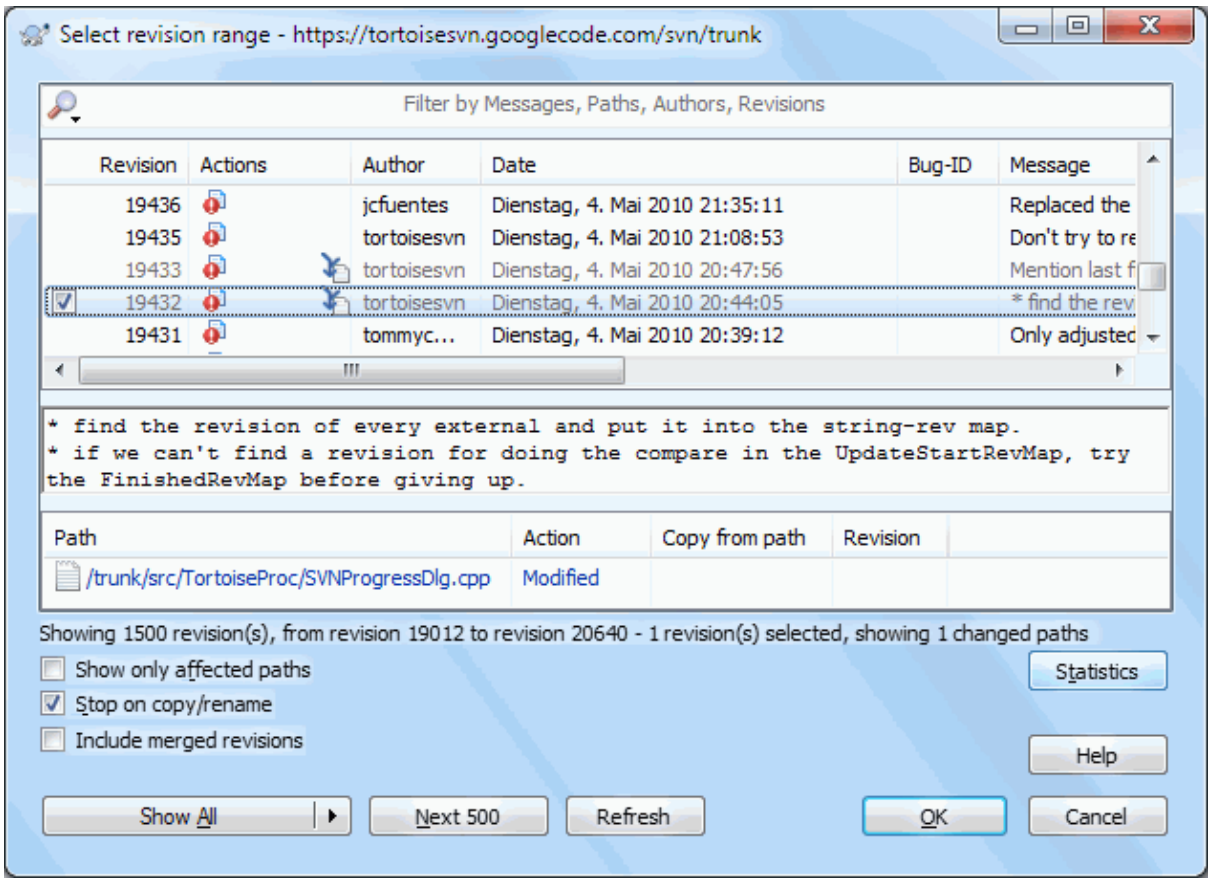


Figure 4.24. La boîte de dialogue de journal montrant les révisions avec suivi de fusion

Si vous voulez voir le détail des révisions qui ont été fusionnées dans cette livraison, utilisez la case à cocher Inclure les révisions fusionnées. Cela récupèrera à nouveau les commentaires de révision, mais en incluant 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 le trunk pour que la branche reste synchronisée avec la version principale.

Donc l'historique des fusions de la branche inclura une autre couche d'historique de fusions. Ces différentes couches sont affichées dans la fenêtre du journal avec une indentation différente.

4.10.7. Changer le commentaire et l'auteur

Les propriétés de révision, ou revprops, 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 dépôt, 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 permet de changer les propriétés de révision à tout moment. Mais comme ces changements ne peuvent pas être annulés (car ils ne sont pas versionnés) cette fonctionnalité est désactivée par défaut. Pour la faire marcher, vous devez mettre en place un hook pre-revprop-change. Référez-vous au chapitre sur les *Scripts hooks* [http://svnbook.red-bean.com/en/1.8/svn.reposadmin.create.html#svn.reposadmin.create.hooks] dans le livre de Subversion pour des détails sur la façon de faire. Lisez Section 3.3, « Scripts hook côté serveur » pour des notes complémentaires sur la façon d'implémenter des 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 de journal. Vous pouvez également éditer un commentaire de révision par l'intermédiaire du menu contextuel du panneau du milieu.

Avertissement

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

Important

Comme TortoiseSVN conserve un cache de toutes les informations de journal, les changements apportés à l'auteur et aux commentaires de livraison n'apparaîtront que sur votre installation locale. Les autres utilisateurs de TortoiseSVN verront toujours les (anciens) auteurs et commentaires de livraison cachés, jusqu'à ce qu'ils rafraîchissent le cache du journal. Voir Section 4.10.11, « Rafraîchissement de l'affichage »

4.10.8. 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 de journal. Les dates de début et de fin vous permettent de limiter les résultats à une plage de dates connues. La boîte de recherche vous permet de n'afficher que les messages qui contiennent une expression particulière.

Cliquez sur l'icône de recherche pour sélectionner le type d'information que vous souhaitez rechercher, et pour choisir le mode *regex*. Vous n'aurez généralement besoin que d'une simple recherche de sous-chaînes, mais si vous souhaitez plus de flexibilité, vous pouvez utiliser les expressions régulières. Si vous passez la souris au dessus de la boîte, une infobulle vous donnera des informations sur la façon d'utiliser les fonctions d'expressions régulières ou les fonctions de sous-chaînes. Le filtre fonctionne en vérifiant si les entrées du journal correspondent à votre chaîne de filtrage, puis seules les entrées qui *correspondent* à ce filtre sont affichées.

La recherche de sous-chaîne simple fonctionne d'une façon similaire à un moteur de recherche. Les chaînes à rechercher sont séparées par des espaces, et toutes les chaînes doivent avoir une correspondance. Vous pouvez utiliser le préfixe - pour spécifier qu'une sous-chaîne donnée ne doit pas être trouvée (correspondance inversée pour ce terme), et vous pouvez utiliser ! au début d'une expression pour inverser la correspondance pour

l'expression entière. Vous pouvez utiliser le préfixe + pour spécifier qu'une sous-chaîne doit être incluse, même si elle a précédemment été exclue par un -. Remarquez que l'ordre des inclusions et exclusions est ici significatif. Vous pouvez utiliser des guillemets autour d'une chaîne qui doit contenir des espaces, et si vous voulez chercher un guillemet dans la chaîne vous pouvez utiliser deux guillemets consécutifs comme séquence d'échappement. Remarquez que le caractère antislash *n'est pas* utilisé comme séquence d'échappement et n'a pas de signification spéciale dans des recherches de sous-chaînes simples. Des exemples faciliteront la compréhension :

```
Alice Bob -Eve
```

recherche des chaînes qui contiennent à la fois Alice et Bob, mais pas Eve.

```
Alice -Bob +Eve
```

recherche des chaînes qui soit contiennent Alice et ne contiennent pas Bob, soit contiennent Eve.

```
-Cas +CasSpécial
```

recherche des chaînes qui ne contiennent pas Cas, mais inclut quand même les chaînes qui contiennent CasSpécial.

```
!Alice Bob
```

recherche des chaînes qui ne contiennent pas à la fois Alice et Bob.

```
!-Alice -Bob
```

Vous vous souvenez des lois de De Morgan ? NON(NON Alice ET NON Bob) se réduit en (Alice OU Bob).

```
"Alice et Bob"
```

recherche l'expression exacte « Alice et Bob ».

```
""
```

recherche un guillemet n'importe où dans le texte.

```
"Alice dit ""bonjour"" à Bob"
```

recherche l'expression exacte « Alice dit "bonjour" à Bob ».

Décrire l'utilisation des recherches par expressions régulières dépasse le cadre de ce manuel, mais vous pouvez jeter un coup d'œil à la documentation en ligne et au didacticiel à http://www.regular-expressions.info/.

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

Vous pouvez aussi filtrer les noms de chemin dans le panneau du bas en utilisant la case à cocher Ne montrer que les chemins affectés. Les chemins affectés sont ceux qui contiennent le chemin utilisé pour montrer le journal. Si vous parcourez le journal pour un dossier, cela signifie tout ce qu'il y a dans ce dossier et en-dessous. Pour un fichier cela signifie juste ce fichier. Normalement, la liste des chemins montre tous les chemins qui sont affectés par la même livraison, mais en gris. Si la case est cochée, ces chemins sont cachés.

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 4.18.2, « Propriétés de projet TortoiseSVN » pour savoir comment utiliser cette propriété.

Pas de formatage des commentaires depuis l'explorateur de dépôt

Comme le formatage requiert l'accès aux propriétés de Subversion, vous n'en verrez les résultats qu'avec une copie de travail extraite. Récupérer des propriétés à distance est une opération lente, donc cette fonctionnalité n'est pas active dans le navigateur de dépôt.

4.10.9. 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 semaine par semaine et bien d'autres. Maintenant vous pouvez voir du premier coup d'œil qui a travaillé le plus dur et qui se relâche ;-)

4.10.9.1. Page des statistiques

Cette page vous donne toutes les statistiques que vous pouvez imaginer, en particulier la période et le nombre de révisions couvertes et quelques valeurs min/max/moyennes.

4.10.9.2. Page de livraisons par auteur

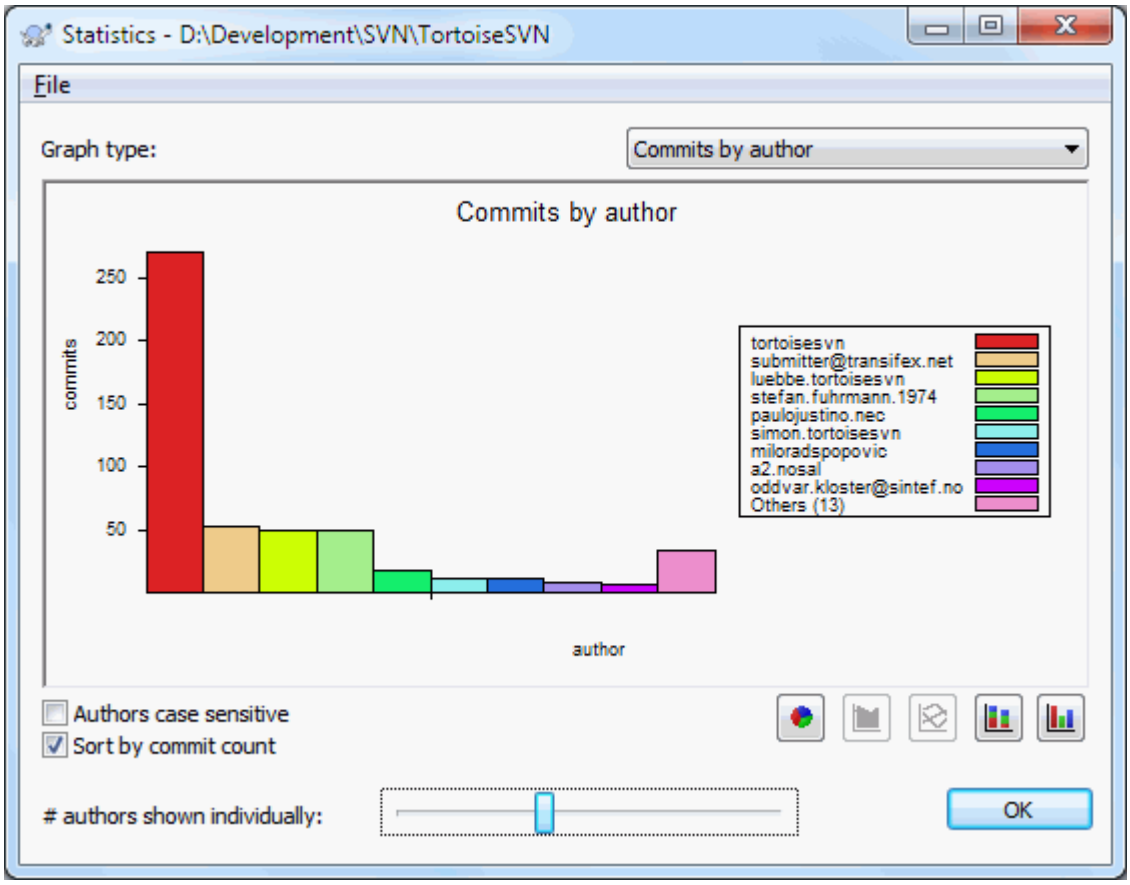


Figure 4.25. Histogramme de livraisons par auteur

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

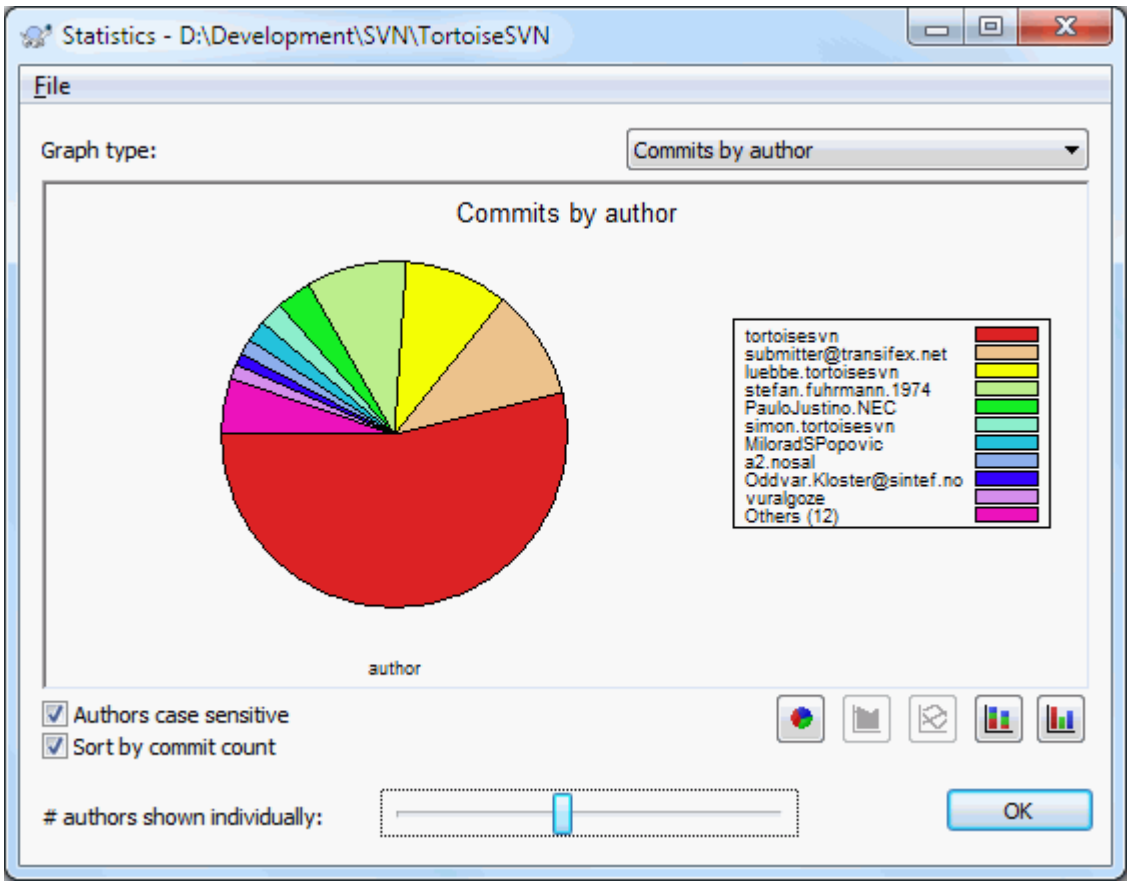


Figure 4.26. Camembert de livraisons par auteur

Quand il y a quelques auteurs principaux et beaucoup de contributeurs mineurs, le nombre de segments minuscules peut rendre le graphique plus difficile à lire. Le curseur de défilement en bas vous permet de définir un seuil (en pourcentage du total des livraisons) en-dessous duquel toutes les activités sont regroupées dans une catégorie *Autres*.

4.10.9.3. Page de livraisons par date

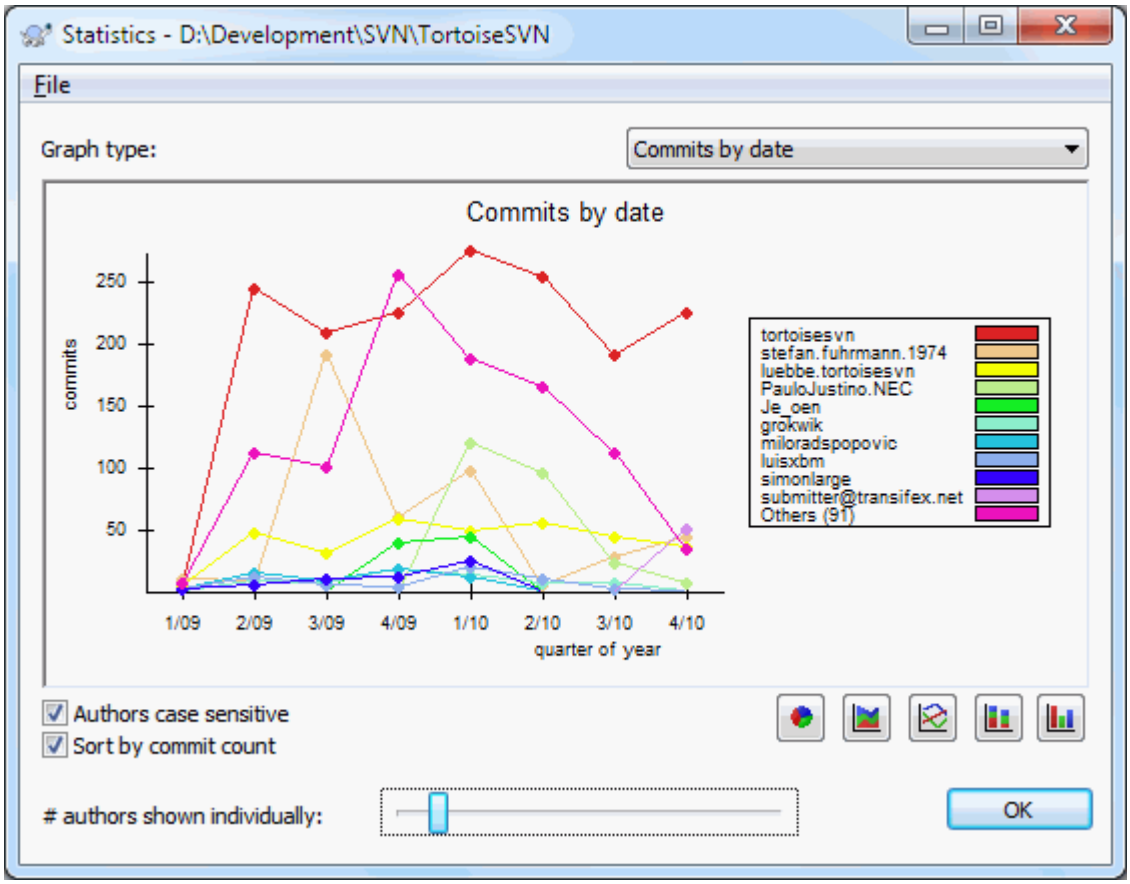


Figure 4.27. Graphique de livraisons par date

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 couvrent la même période que la boîte de dialogue de journal. Si cela ne concerne qu'une révision alors les statistiques ne vous diront pas grand-chose.

4.10.10. Mode hors ligne

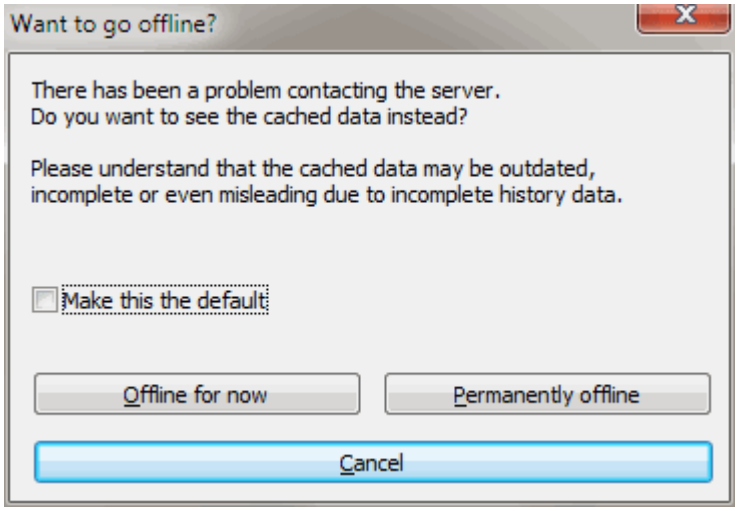


Figure 4.28. Fenêtre de mode hors ligne

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 de 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 :

Hors ligne pour cette fois
: Termine l'opération courante en mode hors ligne, mais réessaie avec le dépôt à la prochaine demande d'informations.

Hors ligne en permanence
: Reste en mode hors ligne jusqu'à la prochaine demande explicite de vérification du dépôt. Voir Section 4.10.11, « Rafraîchissement de l'affichage ».

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

La case à cocher Définir comme choix par défaut empêche cette boîte de dialogue de réapparaître et retient toujours l'option choisie par la suite. Vous pouvez toujours modifier (ou supprimer) cette valeur par défaut plus tard dans le menu TortoiseSVN → Paramètres.

4.10.11. Rafraîchissement de l'affichage

Si vous voulez vérifier si de nouveaux commentaires sont arrivés sur le serveur, vous pouvez simplement rafraîchir l'affichage en appuyant sur **F5**. Si vous utilisez le cache pour les commentaires (ce qui est le comportement par défaut), cette vérification se fera et seuls les nouveaux commentaires seront rapatriés. 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 **Maj-F5** ou **Ctrl-F5** pour récupérer à nouveau les messages affichés à partir du serveur et mettre à jour le cache du journal. Notez que ceci affecte uniquement les messages affichés à cet instant et que ceci n'invalide pas le cache complet pour ce dépôt.

4.11. 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 distincts. 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.

4.11.1. Différences de fichier

Changements locaux
: Si vous voulez voir quels changements *vous* avez fait dans votre copie de travail, utilisez simplement 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 trunk (si vous travaillez sur une branche) ou sur une branche spécifique (si vous travaillez sur le trunk), vous pouvez utiliser le menu contextuel de l'explorateur. Maintenez la touche **Maj** appuyée 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 dépôt avec laquelle vous voulez comparer votre fichier local.

  Vous pouvez aussi utiliser l'explorateur de dépôt et sélectionner deux arborescences à comparer, par exemple deux étiquettes, ou une branche/étiquette et le tronc. Le menu contextuel vous permet alors de les comparer en utilisant Comparer les révisions. Plus d'informations dans Section 4.11.3, « Comparer des dossiers ».

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 livrée et votre copie de travail, en supposant que la copie de travail n'ait pas été modifiée, faites simplement un clic droit sur le fichier. Ensuite, sélectionnez TortoiseSVN → Comparer avec la révision précédente. Ceci calculera les différences entre la révision avant la date de dernière livraison (comme sauvegardée dans votre copie de travail) et la BASE de travail. Ceci vous montre la dernière modification effectuée sur ce fichier et qui l'a amené à l'état dans lequel vous le voyez maintenant dans votre copie de travail. Ceci ne va pas montrer les modifications plus récentes que votre copie de travail.

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 modifiés dans ce dossier. Plus d'informations dans Section 4.11.3, « Comparer des dossiers ».

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 de 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**). Puis, à partir du menu contextuel de l'explorateur, sélectionnez TortoiseSVN → Voir les différences.

Si les fichiers à comparer ne sont pas situés dans le même dossier, utilisez la commande TortoiseSVN → Comparer ultérieurement pour marquer le premier fichier comme à comparer, puis naviguez jusqu'au second fichier et utilisez TortoiseSVN → Comparer avec "chemin/du/fichier/marqué". Pour retirer la marque sur le premier fichier, utilisez la commande TortoiseSVN → Comparer ultérieurement une nouvelle fois, mais gardez le modificateur **Ctrl** appuyé en cliquant dessus.

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 dépôt Subversion, vous pouvez le faire directement dans l'explorateur en sélectionnant le fichier puis en maintenant enfoncée 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 dossier 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 voir non seulement 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 4.24.2, « Annoter les différences » pour plus de détails.

Comparaison entre répertoires
: Les outils intégrés fournis avec TortoiseSVN ne supportent pas la visualisation de différences entre les hiérarchies de répertoire. Mais si vous avez un outil externe qui, lui, supporte cette fonctionnalité, vous pouvez l'utiliser à la place. Dans Section 4.11.6, « Outils de différenciation/fusion externes » nous vous parlerons de quelques outils que nous avons utilisés.

Si vous avez configuré un outil de diff alternatif, vous pouvez utiliser la touche **Maj** quand vous sélectionnez la commande Diff pour utiliser l'outil alternatif. Lisez Section 4.31.5, « Réglages des programmes externes » pour savoir comment configurer les autres outils de diff.

4.11.2. 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'indentation d'une section. Malheureusement ceci va marquer un grand nombre de lignes comme étant modifiées, bien qu'il n'y ait pas de changement à la signification du code. Ces options 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 TortoiseMerge.

Ignorer les fins de ligne exclut les modifications qui ne concernent que le style des caractères de fin de ligne.

Comparer les espacements inclut les modifications d'indentation et d'espaces à la liste des lignes ajoutées/supprimées.

Ignorer les changements d'espaces exclut les changements dus uniquement à une modification du nombre ou du type d'espaces, par exemple un changement d'indentation ou le passage des tabulations aux espaces. Ajouter un espace où il n'y en avait pas précédemment, ou supprimer complètement un espace est toujours indiqué comme un changement.

Ignorer tous les caractères d'espacement exclut toutes les modifications qui ne concernent que des caractères d'espacement.

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

4.11.3. Comparer des dossiers

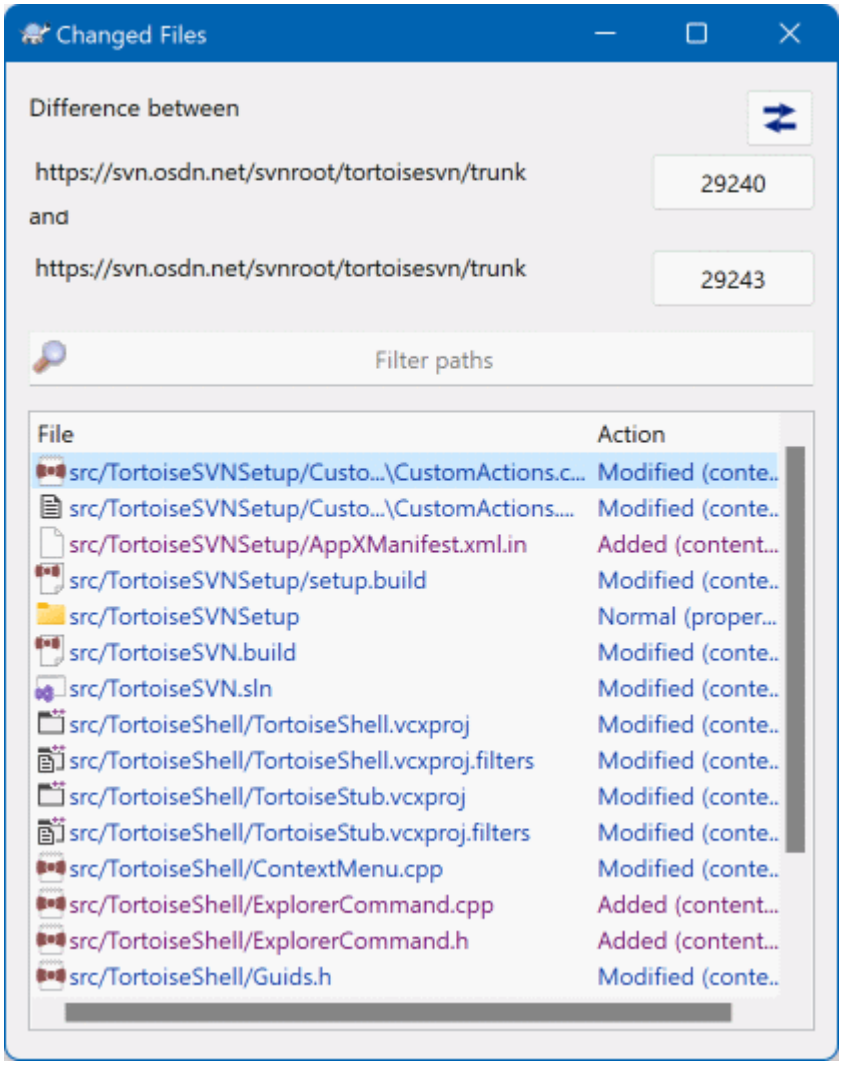


Figure 4.29. La boîte de dialogue Comparer les révisions

Quand vous sélectionnez deux arborescences dans l'explorateur de dépôt ou quand vous sélectionnez deux révisions d'un dossier dans la boîte de dialogue de journal, vous pouvez Menu contextuel → Comparer 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 choisir un emplacement où sauvegarder l'arbre des modifications.

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

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

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 limiter la liste aux noms de fichiers contenant un texte donné. Remarque : la boîte de dialogue utilise une recherche texte simple, donc si vous voulez restreindre la liste aux fichiers C vous devrez saisir `.c` et non `*.c`.

4.11.4. Comparaison d'images avec TortoiseIDiff

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

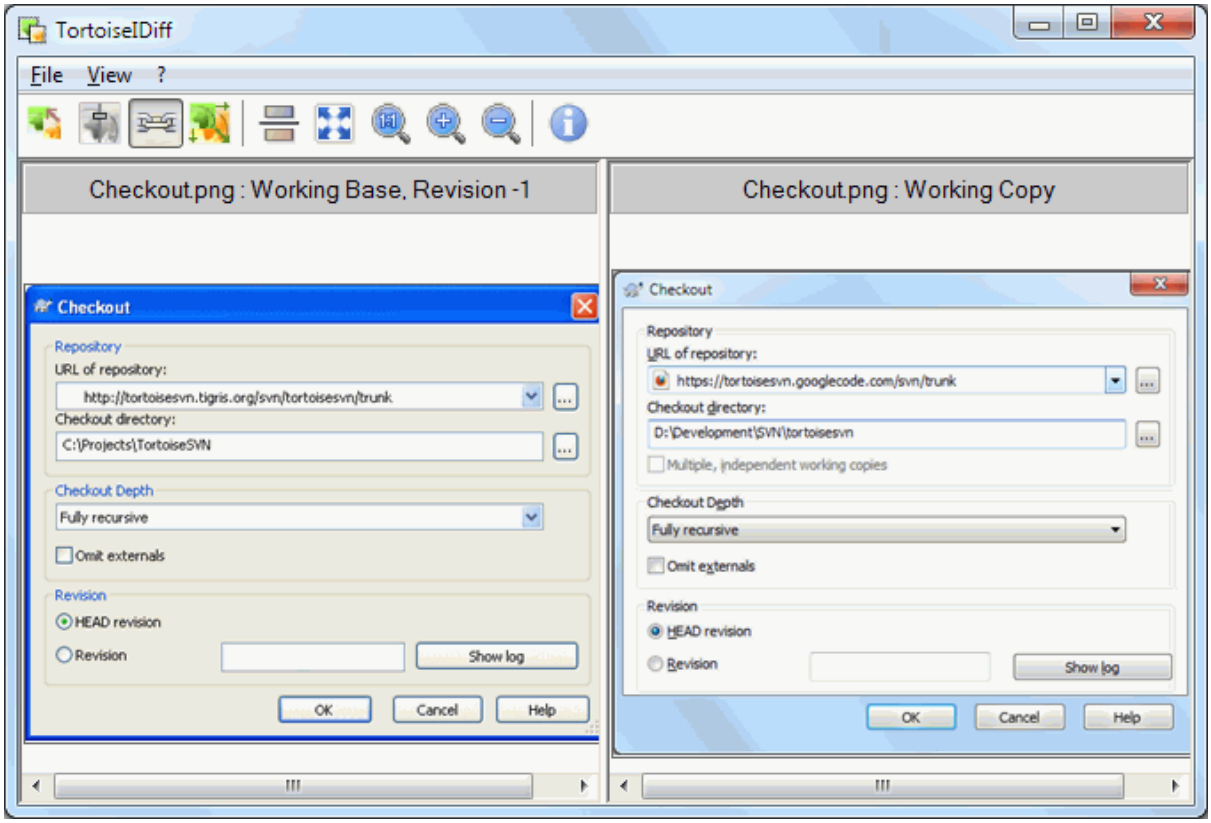


Figure 4.30. Le visualiseur de différences d'images

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 faire comme si vous compariez deux calques par transparence.

Naturellement vous pouvez aussi faire un zoom avant et arrière et vous déplacer sur l'image. Vous pouvez également effectuer un mouvement panoramique de l'image en la faisant glisser avec le bouton gauche. 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 vous gêne, 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 superposées, 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+Maj**-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. Cela peut être utile lorsque vous cherchez beaucoup de petites modifications.

Parfois vous voulez voir une différence plutôt qu'un mélange. Vous pourriez avoir les fichiers d'images de deux révisions d'un circuit imprimé et souhaiter voir quelles pistes ont changé. 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 blanches et les changements apparaîtront en couleur.

4.11.5. Comparaison de documents Office

Quand vous voulez comparer des documents binaires, vous devez normalement utiliser le logiciel que vous avez utilisé pour créer le document, puisqu'il comprend le format de fichier. Pour les suites bureautiques couramment utilisées Microsoft Office et Open Office, il y a effectivement des fonctionnalités d'affichage de différences, et TortoiseSVN inclut des scripts pour les appeler avec les bons paramètres quand vous comparez des fichiers avec les extensions connues. Vous pouvez vérifier quelles extensions de fichier sont supportées et en ajouter en allant dans TortoiseSVN → Configuration et en cliquant sur Avancé dans la section Visualiseur de différences.

Problèmes avec Office 2010

Si vous avez installé la version *Click-to-Run* d'Office 2010 et que vous essayez de comparer des documents, vous pouvez avoir un messsage d'erreur de Windows Script Host du type « le composant ActiveX ne peut pas créer l'objet word.Application ». Il semble qu'il faille utiliser la version MSI d'Office pour avoir la fonctionnalité de comparaison.

4.11.6. 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* [https://winmerge.org/] est un excellent outil de comparaison open source qui sait aussi manipuler les répertoires.

Perforce Merge
: Perforce est un logiciel de gestion de version commercial, mais vous pouvez télécharger l'outil de comparaison/fusion gratuitement. Vous pouvez obtenir plus d'informations de *Perforce* [https://www.perforce.com/perforce/products/merge.html].

KDiff3
: KDiff3 est un outil de comparaison gratuit qui est également capable de gérer les répertoires. Vous pouvez le télécharger *ici* [http://kdiff3.sf.net/].

SourceGear DiffMerge
: SourceGear Vault est un logiciel de gestion de version commercial, mais vous pouvez télécharger l'outil de comparaison/fusion gratuitement. Vous pouvez obtenir plus d'informations de *SourceGear* [https://www.sourcegear.com/diffmerge/].

ExamDiff
: ExamDiff Standard est un logiciel gratuit. Il peut gérer 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 Unicode. Vous pouvez les télécharger chez *PrestoSoft* [http://www.prestosoft.com/].

Beyond Compare
: Comme ExamDiff Pro, c'est un excellent outil de comparaison shareware qui est capable de gérer les comparaisons de répertoire et Unicode. Téléchargez-le chez *Scooter Software* [https://www.scootersoftware.com/].

Araxis Merge
: Araxis Merge est un outil commercial utile pour comparer et fusionner les fichiers et les dossiers. Il fait des comparaisons à trois éléments pour les fusions et a des liens de synchronisation à utiliser si vous avez changé l'ordre des fonctions. Téléchargez-le chez *Araxis* [https://www.araxis.com/merge/index.html].

Lisez Section 4.31.5, « Réglages des programmes externes » pour des informations sur la façon de configurer TortoiseSVN pour utiliser ces outils.

4.12. Ajouter de nouveaux fichiers et répertoires

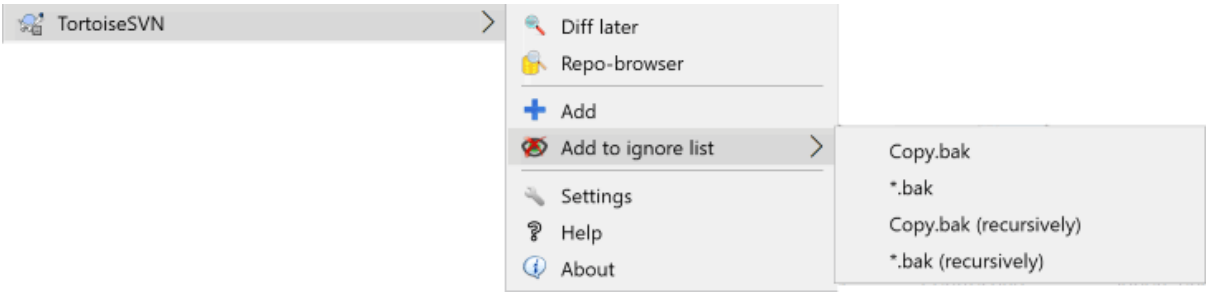


Figure 4.31. Menu contextuel de l'explorateur pour les fichiers non versionnés

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

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

Ajouts multiples

Vous pouvez aussi utiliser la commande Ajouter sur des dossiers 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. faites-les glisser avec le bouton droit vers le nouvel emplacement dans 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....

4.13. Copier/déplacer/renommer des fichiers et des dossiers

Il arrive souvent que les fichiers dont vous avez besoin se trouvent déjà dans un autre projet sur votre dépôt, et que vous vouliez les rajouter au vôtre. Vous pourriez simplement copier les fichiers et les ajouter en gestion de version, mais cela ne vous donnerait pas leur historique. Et si par la suite vous corrigez un bogue dans les fichiers originaux, vous ne pourrez fusionner automatiquement la correction que si la nouvelle copie est apparentée à l'original dans Subversion.

La façon la plus simple de copier des fichiers et des dossiers depuis une copie de travail est d'utiliser le menu du clic droit glissé. Quand vous faites glisser avec le bouton droit un fichier ou dossier d'une copie de travail à une autre, ou même dans le même dossier, un menu contextuel apparaît quand vous relâchez le bouton de la souris.

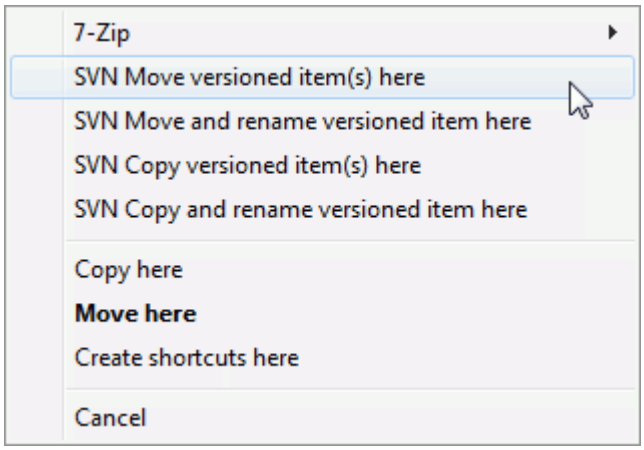


Figure 4.32. Menu pour un répertoire sous contrôle de version lors d'un glisser-déplacer avec le bouton droit

Vous pouvez maintenant copier du contenu versionné existant vers un nouvel emplacement, avec la possibilité de le renommer en même temps.

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 copier/coller familier. Utilisez Copier ou Couper pour copier un ou plusieurs éléments versionnés dans le presse-papiers. Si le presse-papiers contient des éléments versionnés, vous pouvez alors utiliser TortoiseSVN → Coller (remarque : il ne s'agit PAS de la commande Windows Coller habituelle) 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 dépôt en utilisant TortoiseSVN → Branche/étiquette. Lisez Section 4.20.1, « Créer une branche ou une étiquette » pour plus d'informations.

Vous pouvez localiser une ancienne version d'un fichier ou d'un répertoire dans la boîte de dialogue de journal et la copier directement à un autre endroit du dépôt en utilisant Menu contextuel → Créer une branche/étiquette depuis la révision. Lisez Section 4.10.3, « Obtenir des informations supplémentaires » pour plus d'informations.

Vous pouvez également utiliser l'explorateur de dépôt pour localiser du contenu, et le copier directement dans votre copie de travail, ou entre deux endroits du dépôt. Lisez Section 4.25, « L'explorateur de dépôt » pour plus d'informations.

Impossible de copier entre dépôts

Tandis que vous pouvez copier ou déplacer des fichiers et dossiers *à l'intérieur* d'un même dépôt, vous ne *pouvez pas* copier ou déplacer des fichiers ou des dossiers d'un dépôt à l'autre avec

TortoiseSVN tout en préservant l'historique. Ceci 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 dépôt.

Si vous n'êtes pas sûr de savoir si deux URL pointant sur le même serveur font référence à un même dépôt, utilisez l'explorateur de dépôt pour ouvrir une des deux et localiser sa racine. Si vous voyez les deux chemins dans la même fenêtre d'explorateur de dépôt, alors ils sont dans le même dépôt.

4.14. Ignorer des fichiers et des répertoires

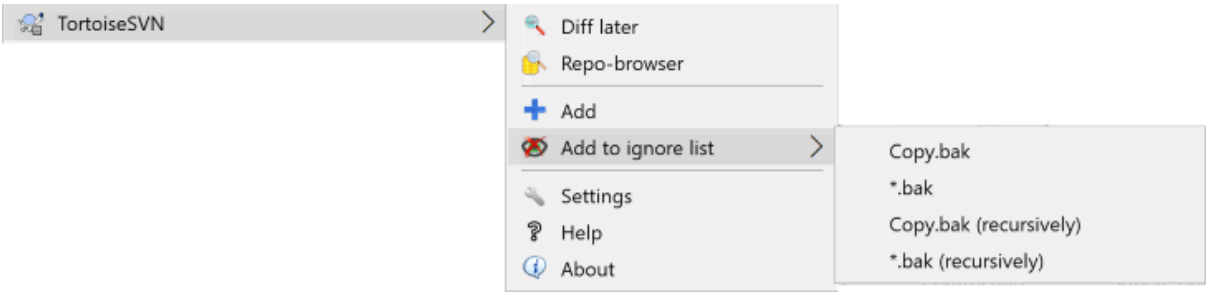


Figure 4.33. Menu contextuel de l'explorateur pour les fichiers non versionnés

Dans la plupart des projets, il y aura des fichiers et des dossiers qui ne devraient pas être en gestion de version. Par exemple, il peut s'agir de fichiers créés par le compilateur, `*.obj, *.lst`, peut-être un dossier de destination qui va recevoir l'exécutable. Chaque fois que vous livrez des changements, TSVN vous montre vos fichiers non versionnés, ce qui encombre la liste des fichiers dans la boîte de dialogue de livraison. Vous pouvez bien 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 générés 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 seront toujours signalés.

Si vous faites un clic droit sur un fichier non versionné, et que vous sélectionnez la commande TortoiseSVN → Ajouter à la liste des ignorés dans le menu contextuel, un sous-menu apparaît pour vous permettre de sélectionner uniquement ce fichier, ou tous les fichiers avec la même extension. Les deux éléments ont aussi chacun un élément

Si vous choisissez la version `(récursivement)` de la commande du menu contextuel, l'élément sera ignoré non seulement pour le dossier sélectionné mais également pour tous les sous-dossiers. Cependant, cela nécessite la version 1.8 ou supérieure du client SVN.

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 accéder directement à la propriété `svn:ignore` d'un dossier. Cela vous permet de spécifier des modèles plus généraux en utilisant des jokers, comme décrit dans la section ci-dessous. Lisez Section 4.18, « Configuration des projets » pour plus d'informations sur la définition directe des propriétés. Notez qu'il faut placer une règle de filtrage par ligne. Les séparer avec des espaces ne fonctionne 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 4.31.1, « 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 ».

4.14.1. Correspondance avec des modèles dans la liste des ignorés

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 avec des modèles est sensible à la casse, ce qui peut causer des problèmes sous Windows. Vous pouvez forcer l'insensibilité à la casse à la dure en appariant les caractères. Par exemple, pour ignorer `*.tmp` quelle que soit la casse, vous pouvez 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 ignorés

Vous ne devez pas inclure de chemin d'accès dans vos modèles. La correspondance avec les modèles est faite pour être utilisée à la place des noms de fichier ou de dossier. Si vous souhaitez ignorer tous les dossiers `CVS`, ajoutez juste `CVS` à la liste des ignorés. Il n'est pas nécessaire 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 fiable d'avoir ce type de règle de filtrage avec les modèles.

4.15. Supprimer, déplacer et renommer

Subversion permet le renommage et le déplacement de fichiers et de dossiers. Des entrées de menu existent pour supprimer et renommer dans le sous-menu TortoiseSVN.

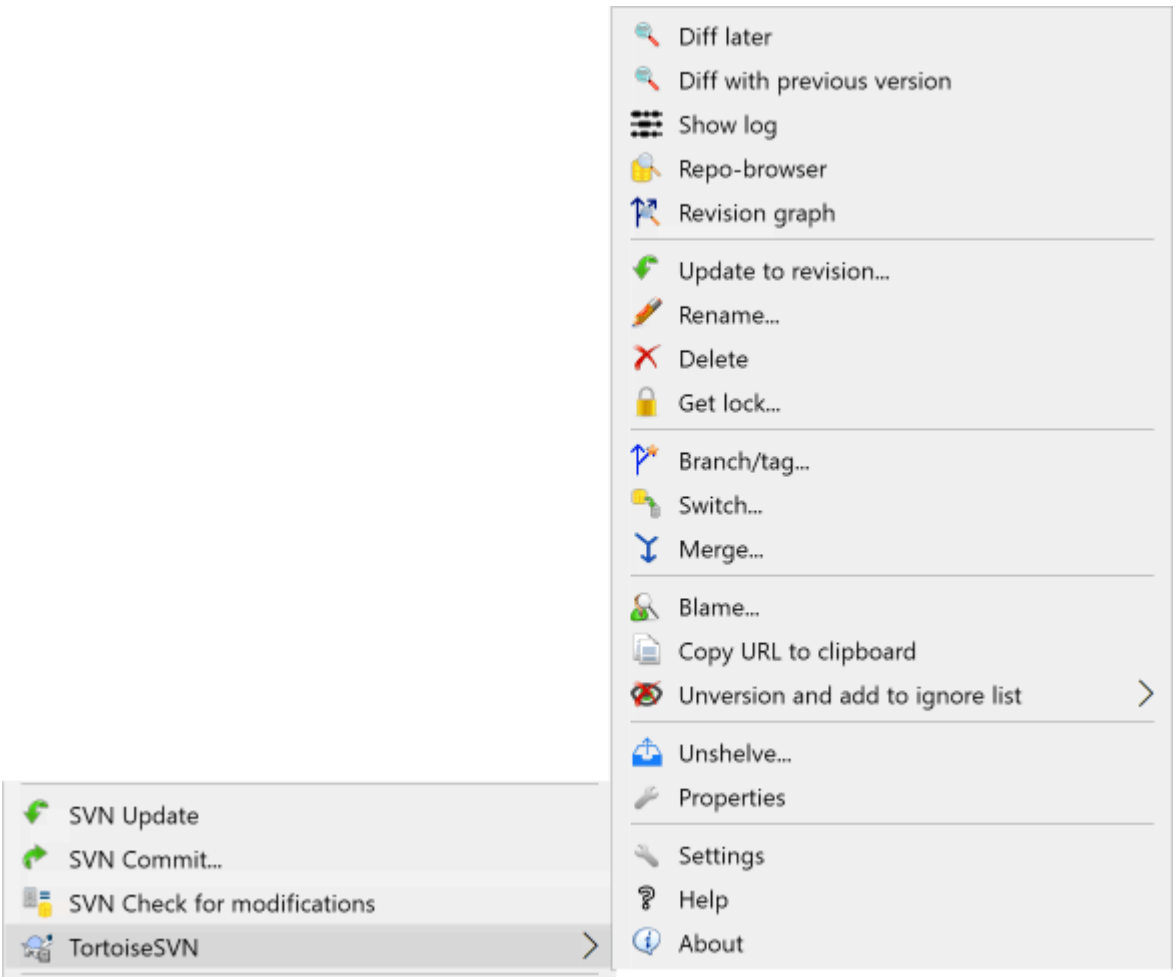


Figure 4.34. Menu contextuel de l'explorateur pour les fichiers versionnés

4.15.1. Supprimer des fichiers et des dossiers

Utilisez TortoiseSVN → Supprimer pour enlever des fichiers ou des dossiers de Subversion.

Quand vous faites TortoiseSVN → Supprimer sur un fichier ou un dossier, il est immédiatement effacé de votre copie de travail et marqué pour suppression dans le dépôt à la prochaine livraison. Le dossier parent de l'élément affiche l'icône superposée de l'état « modifié ». Tant que vous n'avez pas livré la modification, vous pouvez récupérer le fichier en utilisant TortoiseSVN → Revenir en arrière sur le répertoire parent.

Si vous voulez supprimer un élément dans le dépôt, mais le conserver localement comme un fichier/répertoire non versionné, utilisez Menu contextuel étendu → Supprimer (conserver localement). Vous devez tenir la touche **Maj** enfoncée tout en effectuant un clic droit avec la souris sur l'élément désiré dans le volet de l'explorateur (volet droit) afin de voir cette commande dans le menu contextuel étendu.

Si un élément est effacé depuis l'explorateur au lieu d'utiliser le menu contextuel de TortoiseSVN, la boîte de dialogue de livraison signale ces éléments comme manquants et vous permet de les retirer de la gestion de version avant la livraison. Cependant, si vous faites une mise à jour de votre copie de travail, Subversion détectera l'élément manquant et le remplacera avec la dernière version du dépôt. Si vous avez besoin de supprimer un fichier versionné, utilisez toujours TortoiseSVN → Supprimer pour que Subversion ne soit pas obligé de deviner ce que vous cherchez réellement à faire.

Récupérer un fichier ou un répertoire supprimé

Si vous avez supprimé un fichier ou un dossier et déjà livré cette opération de suppression dans le dépôt, un simple TortoiseSVN → Revenir à l'ancienne version ne pourra 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 (si vous ne la connaissez pas, utilisez la fenêtre de journal pour la retrouver) ouvrez l'explorateur de dépôt et basculez vers cette révision. Sélectionnez ensuite le fichier ou le dossier que vous avez supprimé, faites un clic droit et sélectionnez Menu contextuel → Copier vers... avec comme cible de l'opération de copie le chemin vers votre copie de travail.

4.15.2. Déplacer des fichiers et des dossiers

Si vous voulez faire un simple renommage d'un fichier ou d'un dossier sans changer son emplacement, utilisez Menu contextuel → Renommer... Entrez le nouveau nom de l'élément et vous avez terminé.

Si vous voulez déplacer des fichiers dans votre copie de travail, peut-être vers un sous-dossier différent, utilisez le glisser-déposer avec le bouton droit :

1. sélectionnez les fichiers ou les répertoires que vous voulez déplacer
2. faites-les glisser avec le bouton droit vers le nouvel emplacement dans la copie de travail
3. relâchez le bouton droit de la souris
4. dans le menu qui apparaît, sélectionnez Menu contextuel → SVN Déplacer les éléments versionnés ici

Livrer 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 suppression du renommage/déplacement apparaisse dans la boîte de dialogue Livrer. Si vous ne livrez pas la partie suppression du renommage/déplacement, il restera dans le dépôt et une mise à jour par vos collègues ne supprimera pas le vieux fichier. C'est-à-dire qu'ils auront *les deux* copies, la vieille et la nouvelle.

Vous *devez* livrer un renommage de dossier avant de changer l'un des fichiers de ce dossier, sinon votre copie de travail peut être vraiment perturbée.

Une autre façon de déplacer ou de copier des fichiers est d'utiliser les commandes copier/couper de Windows. Sélectionnez les fichiers que vous voulez copier, faites un clic droit et choisissez Menu contextuel → Copier

Vous pouvez aussi utiliser l'explorateur de dépôt pour déplacer des éléments. Lisez Section 4.25, « L'explorateur de dépôt » pour en savoir plus.

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 dépôt parent, en dérangeant probablement beaucoup d'autres personnes. Si vous devez déplacer un dossier externe, vous devriez utiliser un déplacement ordinaire, et ensuite ajuster les propriétés `svn:externals` des répertoires parents de la source et de la destination.

4.15.3. Gérer les conflits de casse dans les noms de fichier

Si le dépôt contient deux fichiers qui ont le même nom mais qui ne diffèrent que par la casse (par exemple `TEST.TXT` et `test.txt`), vous ne pourrez pas mettre à jour ou extraire le répertoire parent sous Windows. Bien que Subversion soit sensible à la casse dans les noms de fichiers, ce n'est pas le cas de Windows.

Cela arrive parfois lorsque deux personnes livrent, depuis des copies de travail séparées, des fichiers qui se trouvent avoir le même nom, mais avec une différence de casse. Cela peut également arriver quand les fichiers sont livrés depuis un système d'exploitation qui utilise un système de fichiers sensible à la casse, comme Linux.

Dans ce cas, vous devez décider lequel des deux vous voulez conserver et supprimer (ou renommer) l'autre du dépôt.

Éviter que deux fichiers aient le même nom

Il y a un script hook serveur disponible à l'adresse : https://svn.apache.org/repos/asf/subversion/trunk/contrib/hook-scripts/ qui bloque les livraisons qui créeraient des conflits de casse.

4.15.4. 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 de livrer vos modifications, Subversion peut voir l'ancien nom de fichier comme manquant et le nouveau comme un fichier non versionné. Vous pourriez simplement livrer le nouveau nom de fichier pour le rajouter dans le dépôt, mais on perdrait alors l'historique, car 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 dans les boîtes de dialogue 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 → Réparer le déplacement pour lier les deux fichiers en tant que renommage.

4.15.5. Supprimer les fichiers non versionnés

Généralement vous paramétrez votre liste des ignorés pour que tous les fichiers générés soient ignorés dans Subversion. Mais que faire si vous souhaitez effacer tous les éléments ignorés pour produire une génération propre ? Généralement vous définiriez cela dans votre makefile, mais si vous êtes en train de déboguer le makefile ou de changer le système de build, il est utile d'avoir un moyen de nettoyer la plate-forme.

TortoiseSVN fournit précisément une telle option avec Menu contextuel étendu → Supprimer les éléments non versionnés.... Vous devez tenir enfoncée la touche **Maj** tout en effectuant un clic droit sur un dossier dans le volet de l'explorateur (volet droit) afin de voir cette commande apparaître dans le menu contextuel étendu. Cela a pour effet d'ouvrir une boîte de dialogue qui répertorie tous les fichiers non versionnés où qu'ils soient 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.

4.16. 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, faire un clic droit pour faire apparaître le menu contextuel et sélectionner ensuite la commande TortoiseSVN → Revenir en arrière. Une boîte de dialogue apparaîtra avec la liste des fichiers que vous avez changés et que vous pouvez restaurer. Choisissez ceux que vous voulez restaurer et cliquez sur OK.

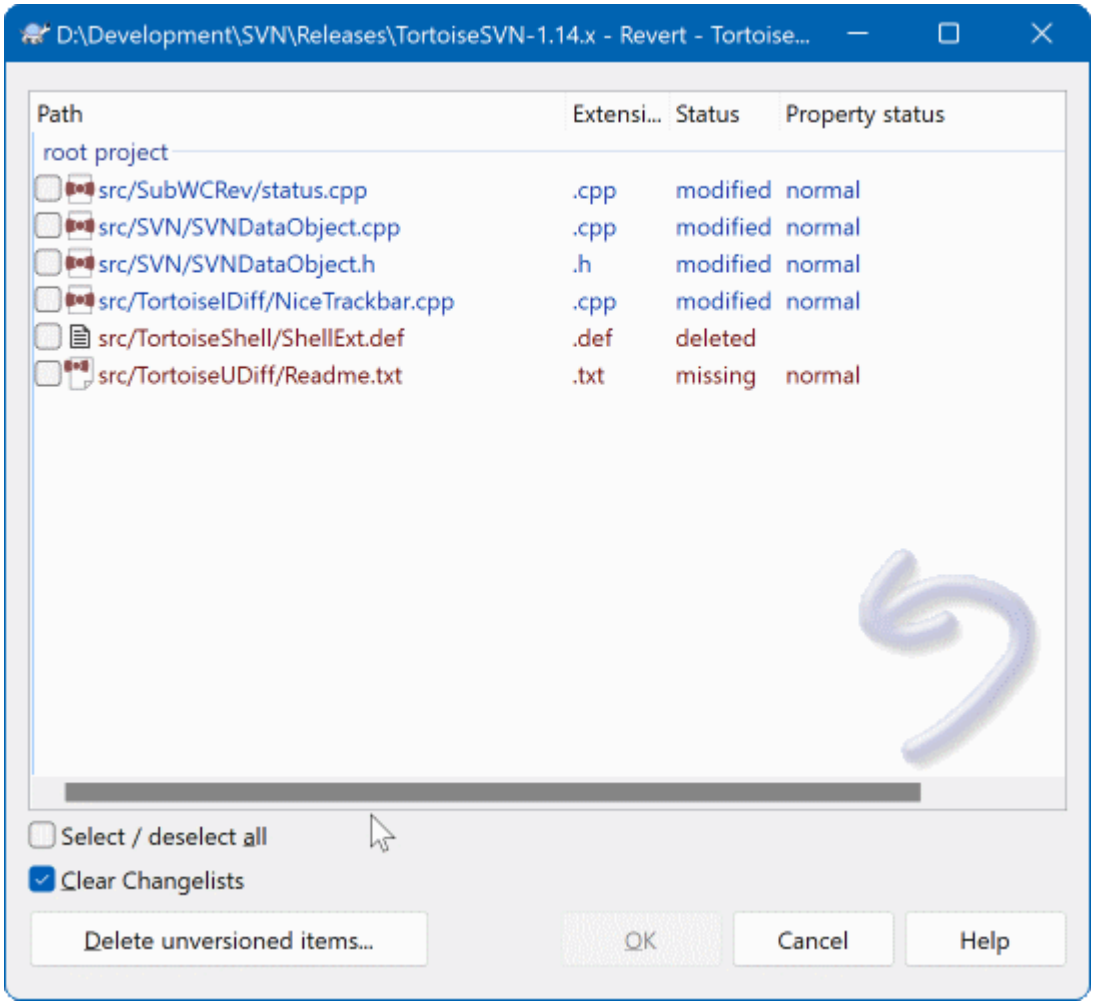


Figure 4.35. La boîte de dialogue Revenir en arrière

Si vous voulez également effacer toutes les listes de changements qui ont été définies, cochez la case en bas de la boîte de dialogue.

Si vous voulez annuler une suppression ou un renommage, vous devez utiliser Revenir en arrière sur le dossier parent, puisque vous ne pouvez plus faire de clic droit sur les éléments supprimés qui n'existent plus.

Si vous souhaitez annuler l'ajout d'un élément, cela apparaît dans le menu contextuel comme TortoiseSVN → Annuler l'ajout.... En réalité, cette commande effectuera aussi un retour en 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.

Comme le retour en arrière est parfois utilisé pour nettoyer une copie de travail, il y a un bouton supplémentaire qui vous permet de supprimer des éléments non versionnés. Lorsque vous cliquez sur ce bouton, une autre boite de dialogue apparaît avec la liste de tous les éléments non versionnés, que vous pouvez ensuite sélectionner pour suppression.

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 4.10, « La boîte de dialogue de journal de révision » pour plus d'informations.

Le retour en arrière est lent

Lorsque vous faites un retour en arrière sur des modifications, vous trouverez peut-être que l'opération prend beaucoup plus longtemps que ce à quoi vous vous attendiez. C'est parce que la version modifiée du fichier est envoyée à la corbeille, afin que vous puissiez récupérer vos modifications si vous faites un retour en arrière par erreur. Toutefois, si votre corbeille est pleine, Windows prend beaucoup de temps pour trouver un endroit où mettre le fichier. La solution est simple : soit vider la corbeille, soit désactiver l'option Utiliser la corbeille lors d'un retour en arrière dans les paramètres de configuration de TortoiseSVN.

4.17. Nettoyer

Si une commande Subversion ne parvient pas à s'exécuter correctement, peut-être à cause de problèmes du côté du serveur, votre copie de travail peut se retrouver dans un état incohérent. Dans ce cas, il vous faudra utiliser TortoiseSVN → Nettoyer sur le dossier. C'est généralement une bonne idée de faire cela à la racine de la copie de travail.

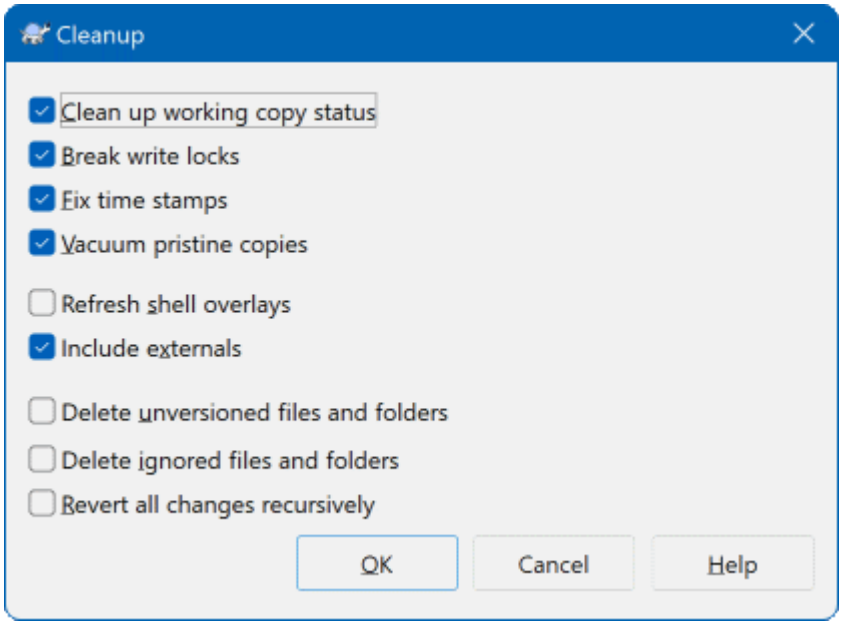


Figure 4.36. La boîte de dialogue Nettoyer

Dans la boîte de dialogue de nettoyage, il y a aussi d'autres options utiles pour obtenir la copie de travail dans un état `clean`.

Nettoyer le statut de la copie de travail
: Comme indiqué ci-dessus, cette option essaie de remettre une copie de travail incohérente en état de marche. Cela n'affecte pas les données que vous avez, mais uniquement les états internes de la base de données de la copie de travail. Il s'agit de la commande `cleanup` que vous connaissez dans les anciens clients TortoiseSVN ou d'autres clients SVN.

Ouvrir les verrous d'écriture
: Si cette case est cochée, tous les verrous d'écriture sont supprimés de la base de données de la copie de travail. Dans la majorité des cas, c'est nécessaire à l'efficacité du nettoyage !

  Ne décochez cette option que si la copie de travail est utilisée par d'autres utilisateurs ou clients à ce moment. Mais si alors le nettoyage échoue, vous devrez cocher cette option pour qu'il fonctionne.

Corriger les marqueurs temporels
: Ajuste les horodatages de tous les fichiers pour accélérer les vérifications de statut ultérieures. Cela peut permettre d'afficher plus vite toutes les boîtes de dialogue qui contiennent des listes de fichiers de la copie de travail, par exemple la boîte de dialogue de livraison.

Vider les copies du projet
: Supprime les copies primitives inutilisées et compresse toutes les copies primitives restante des fichiers de la copie de travail.

Rafraichir les icônes de recouvrement
: Parfois, les icônes superposées du shell, en particulier sur l'arborescence sur le côté gauche de l'explorateur, ne montrent pas l'état actuel, ou le cache d'état n'est pas parvenu à reconnaître les changements. Dans cette situation, vous pouvez utiliser cette commande pour forcer le rafraîchissement.

Inclure les externes
: Si ceci est coché, alors toutes les actions effectuées incluront les fichiers et les dossiers ayant la propriété `svn:externals`

Supprimer les fichiers et les dossiers non versionnés, Supprimer les fichiers et les dossiers ignorés
: C'est un moyen facile et rapide d'enlever tous les fichiers générés de votre copie de travail. Tous les fichiers et dossiers qui ne sont pas versionnés sont déplacés vers la corbeille.

  Remarque : vous pouvez aussi faire la même chose depuis le menu TortoiseSVN → Revenir en arrière. Cela vous donne également une liste de tous les fichiers et dossiers non versionnés que vous pouvez sélectionner pour suppression.

Annuler tous les changements récursivement
: Cette commande annule toutes vos modifications locales qui n'ont pas encore été livrées.

  Remarque : il est préférable d'utiliser à la place les commandes TortoiseSVN → Revenir en arrière, puisqu'il vous sera possible dans ce cas de voir puis de sélectionner les fichiers pour lesquels vous souhaitez annuler les changements.

4.18. Configuration des projets

4.18.1. Propriétés Subversion

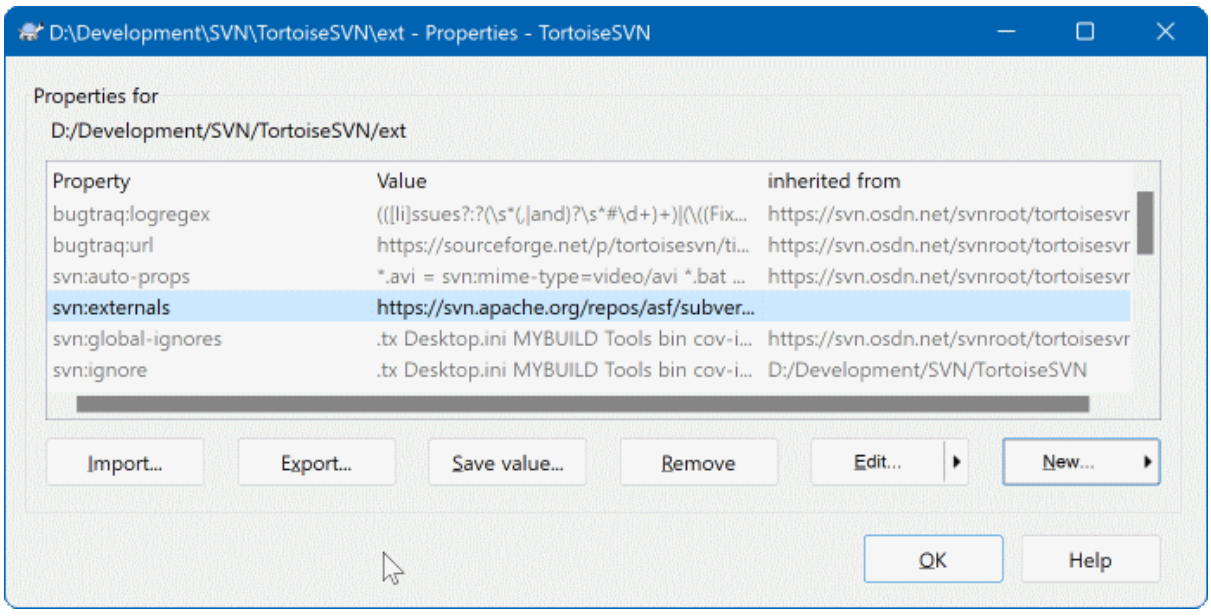


Figure 4.37. Page de propriété de subversion

Vous pouvez lire et définir les propriétés Subversion à partir de la boîte de dialogue de propriétés de Windows, mais aussi depuis TortoiseSVN → Propriétés et au sein des listes de statut de TortoiseSVN, depuis Menu contextuel → Propriétés.

Vous pouvez ajouter vos propres propriétés, ou des propriétés avec une signification spéciale pour Subversion. Celles-ci commencent par `svn:`. `svn:externals` est une de ces propriétés ; regardez comment manipuler les éléments externes dans Section 4.19, « Éléments externes ».

4.18.1.1. svn:keywords

Subversion supporte le développement de mots-clés, comme dans CVS, ce qui peut être utilisé pour intégrer des informations sur le nom de fichier et la révision dans le fichier lui-même. Les mots-clés actuellement supportés sont :

$Date$
: Dernière date de livraison connue. Cette information est obtenue quand vous mettez à jour votre copie de travail. On *ne vérifie pas* le dépôt pour trouver d'éventuels changements plus 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 dépôt.

Id
: Une combinaison raccourcie des quatre mots-clés précédents.

Pour savoir comment utiliser ces mots-clés, voyez la *section sur svn:keywords* [http://svnbook.red-bean.com/fr/1.8/svn.advanced.props.special.keywords.html] dans le livre de Subversion, qui fournit une description complète de ces mots-clés et de comment les activer et les utiliser.

Pour plus d'informations à propos des propriétés dans Subversion, voyez la section sur les *propriétés* [http://svnbook.red-bean.com/fr/1.8/svn.advanced.props.html].

4.18.1.2. Ajouter et modifier des propriétés

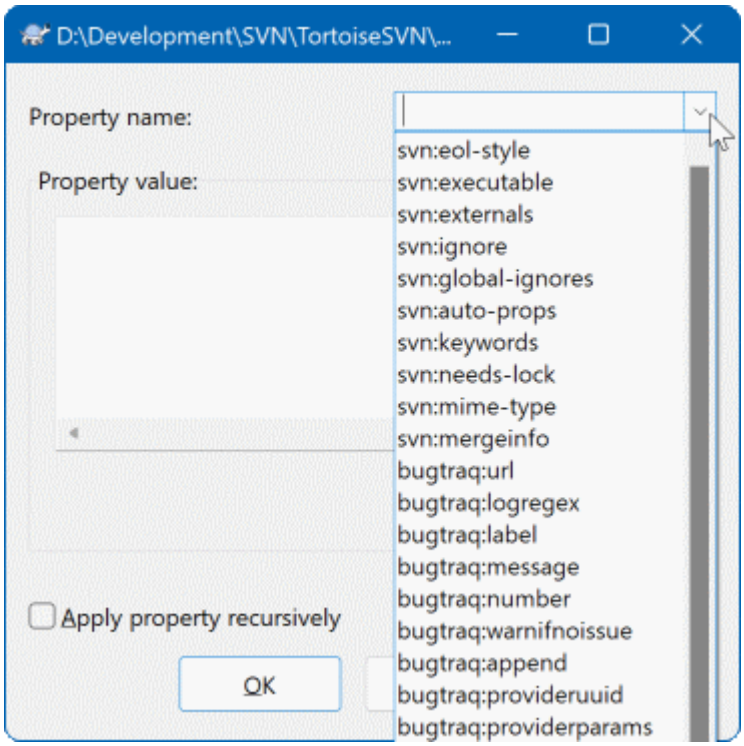


Figure 4.38. Ajouter des propriétés

Pour ajouter une nouvelle propriété, cliquez d'abord sur Nouveau.... Sélectionnez le nom de la propriété voulue à partir du menu, puis remplissez les informations requises dans la boîte de dialogue de propriété spécifique. Ces boîtes de dialogue de propriétés spécifiques sont décrites plus en détail dans Section 4.18.3, « Éditeurs de propriétés » .

Pour ajouter une propriété qui ne possède pas sa propre boîte de dialogue, choisissez Autre dans le menu Nouveau.... Ensuite, sélectionnez une propriété existante dans la liste déroulante ou saisissez un nom de propriété personnalisée.

Si vous voulez appliquer une propriété à plusieurs éléments à la fois, sélectionnez les fichiers/dossiers dans l'explorateur, puis sélectionnez 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.

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.

La propriété `svn:externals` peut être utilisée pour intégrer d'autres projets du même dépôt ou d'un dépôt complètement différent. Pour plus d'informations, lisez Section 4.19, « Éléments externes ».

Editer les propriétés de la révision HEAD

Parce que les propriétés sont versionnées, vous ne pouvez pas modifier les propriétés des révisions précédentes. Si vous regardez les propriétés dans la boîte de dialogue du journal (log) ou d'une

> révision non-HEAD dans le navigateur de dépôt, vous verrez une liste de propriétés et de valeurs, mais pas de contrôles d'édition.

4.18.1.3. Exporter et importer des propriétés

Souvent, vous vous retrouverez à appliquer le même ensemble de propriétés de nombreuses fois, par exemple `bugtraq:logregex`. Pour simplifier le processus de copie des propriétés d'un projet à l'autre, vous pouvez utiliser la fonctionnalité Exporter/Importer.

A partir du fichier ou du dossier où les propriétés sont déjà définies, utilisez TortoiseSVN → Propriétés, sélectionnez les propriétés que vous souhaitez exporter et cliquez sur Exporter.... Vous serez invité à entrer un nom de fichier où les noms et les valeurs des propriétés seront sauvegardés.

Dans le(s) dossier(s) où vous souhaitez appliquer ces propriétés, utilisez TortoiseSVN → propriétés et cliquez sur Importer.... Vous serez invité à entrer un nom de fichier à importer. Naviguez alors vers le lieu où vous avez enregistré précédemment le fichier d'exportation et sélectionnez-le. Les propriétés seront ajoutées aux dossiers de manière non récursive.

Si vous souhaitez ajouter des propriétés à une arborescence de manière récursive, suivez les étapes ci-dessus, puis dans la boîte de dialogue de propriété, sélectionnez chaque propriété, cliquez sur Modifier..., cochez la case Appliquer la propriété récursivement et cliquez sur OK.

Le format de fichier d'importation est binaire et propriétaire à TortoiseSVN. Son seul but est de transférer les propriétés en utilisant les commandes Importer et Exporter, de sorte qu'il n'est pas nécessaire de modifier ces fichiers.

4.18.1.4. Propriétés binaires

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.

4.18.1.5. Configuration automatique des propriétés

Vous pouvez configurer Subversion et TortoiseSVN pour définir automatiquement des propriétés sur les fichiers et sur les dossiers lorsqu'ils sont ajoutés au dépôt. Il y a deux manières de faire cela.

Vous pouvez éditer le fichier de configuration de Subversion pour activer cette fonctionnalité sur votre client. La page Général de la boîte de dialogue de paramètres de TortoiseSVN a un bouton Editer pour vous y amener directement. Le fichier de configuration est un simple fichier texte qui contrôle certains aspects du fonctionnement de Subversion. Vous devez modifier deux choses : premièrement, dans la section `miscellany`, décommentez la ligne `enable-auto-props = yes`. Deuxièmement, vous devez éditer la section suivante pour définir quelles propriétés vous voulez ajouter à quels types de fichier. Cette méthode est une fonctionnalité standard de Subversion et fonctionne avec tous les clients Subversion. Cependant, elle doit être définie individuellement sur chaque client — il n'est pas possible de propager ce paramétrage depuis le dépôt.

Une autre méthode consiste à définir la propriété `tsvn:autoprops` sur les dossiers, tel que décrit dans la section suivante. Cette méthode ne fonctionne que pour les clients TortoiseSVN, mais elle se propage à toutes les copies de travail à l'occasion d'une mise à jour.

Depuis Subversion1.8, vous pouvez aussi définir la propriété `svn:auto-props` sur le dossier racine. Tous les éléments fils héritent automatiquement de la valeur de la propriété.

Quelle que soit la méthode que vous choisissez, vous devez savoir que les propriétés automatiques sont uniquement appliquées aux fichiers au moment où ils sont ajoutés à la copie de travail. Les propriétés automatiques ne changeront jamais les propriétés de fichiers qui sont déjà versionnés.

Si vous voulez être absolument sûr que les nouveaux fichiers ont les bonnes propriétés appliquées, vous devez mettre en place un script hook pré-livraison sur le dépôt pour rejeter les livraisons quand les propriétés requises ne sont pas définies.

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 génère un fichier `.prej`. Supprimez ce fichier après avoir résolu le conflit.

4.18.2. Propriétés de 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 verrouillage. Si vous entrez un message plus court qu'indiqué ici, le verrouillage est désactivé. Cette fonctionnalité est très utile pour vous rappeler de fournir un message descriptif approprié à chaque fois que vous positionnez un verrou. Si cette propriété n'est pas définie, ou si la valeur est zéro, les commentaires de verrouillage 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 fonctionnalité ne marchera 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.

  Il ya aussi des modèles d'action spécifique que vous pouvez utiliser au lieu de `tsvn:logtemplate` . Les modèles d'actions spécifiques sont utilisés s'il sont activés, mais `tsvn:logtemplate` sera utilisé si aucun modèle d'action spécifique est défini.

  Les modèles d'action spécifiques sont les suivants:

  - `tsvn:logtemplatecommit` est utilisé pour toutes les livraisons à partir d'une copie de travail.
  - `tsvn:logtemplatebranch` est utilisé lorsque vous créez une branche / étiquette, ou lorsque vous copiez des fichiers ou des dossiers directement dans l'explorateur de dépôt.
  - `tsvn:logtemplateimport` est utilisé pour les importations.
  - `tsvn:logtemplatedelete` est utilisé lors de la suppression d'éléments directement dans l'explorateur de dépôt.
  - `tsvn:logtemplatemove` est utilisé lors du renommage ou du déplacement d'éléments dans l'explorateur de dépôt.

  - `tsvn:logtemplatemkdir` est utilisé lors de la création de répertoires dans l'explorateur de dépôt.
  - `tsvn:logtemplatepropset` est utilisé lors de la modification de propriétés dans l'explorateur de dépôt.
  - `tsvn:logtemplatelock` est utilisé lors de la pose d'un verrou.

- Subversion vous permet de positionner des « autoprops » qui seront appliquées aux fichiers nouvellement ajoutés ou importés, en fonction de l'extension du fichier. Cela requiert que chaque client ait positionné des autoprops adéquates dans leur fichier de configuration de Subversion. Des `tsvn:autoprops` peuvent être positionnées sur des dossiers, et elles seront fusionnées avec les autoprops locales de l'utilisateur au moment de l'importation ou de l'ajout de fichiers. Le format est le même que pour les autoprops de Subversion, par exemple `*.sh = svn:eol-style=native;svn:executable` positionne deux propriétés sur les fichiers qui ont l'extension `.sh`.

  S'il y a un conflit entre les autoprops locales et `tsvn:autoprops`, les propriétés de projet ont la priorité car elles sont spécifiques à ce projet.

  Depuis Subversion 1.8, vous devriez utiliser la propriété `svn:auto-props` à la place de `tsvn:autoprops`, car la fonctionnalité est identique, mais elle fonctionne avec tous les clients svn et n'est pas spécifique à TortoiseSVN.

- Dans la boîte de dialogue Livrer, vous avez une option pour coller la liste des fichiers changés, en incluant le statut de chaque fichier (ajouté, modifié, etc). `tsvn:logfilelistenglish` définit si le statut de fichier est inséré en anglais ou dans la langue locale. Si la propriété n'est pas définie, la valeur par défaut est `true`.

- TortoiseSVN peut utiliser un vérificateur d'orthographe. Sous Windows 10, le vérificateur d'orthographe du système d'exploitation est utilisé. Sous les versions antérieures de Windows, il peut utiliser les modules de vérification d'orthographe qui sont également utilisés par OpenOffice et Mozilla. Si vous les avez installés, cette propriété déterminera quel vérificateur d'orthographe utiliser, c'est-à-dire dans quelle langue les messages de commentaire de votre projet doivent être écrits. `tsvn:projectlanguage` définit le module de langue que le vérificateur d'orthographe doit utiliser quand vous saisissez un message de log. Vous pouvez trouver les valeurs de langue sur cette page : *MSDN: Language Identifiers* [https://docs.microsoft.com/en-us/windows/desktop/intl/language-identifier-constants-and-strings].

  Vous pouvez saisir cette valeur en décimal, ou en hexadécimal si elle est préfixée avec `0x`. Par exemple l'anglais (États-Unis) 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.

- La propriété `tsvn:logrevregex` définit une expression régulière qui compare les références aux révisions dans un message de log. Ceci est utilisé dans le dialogue de log pour passer ces références en liens qui, lorsqu'il sont cliqués, soit amènent à cette révision (si la révision est déjà affichée dans la boîte de dialogue du log, ou si elle est disponible à partir du cache de log) soit ouvrent une nouvelle boîte de dialogue montrant cette révision.

  L'expression régulière doit correspondre à la référence entière, pas seulement au numéro de révision. Le numéro de révision est extrait de la chaîne de référence automatiquement.

  Si cette propriété n'est pas définie, une expression régulière par défaut est utilisée pour lier les références de révision.

- Il existe plusieurs propriétés qui permettent de configurer des scripts hook côté client. Chaque propriété concerne un type spécifique de script hook.

Les propriétés / scripts disponibles sont

- tsvn:startcommithook
- tsvn:precommithook
- tsvn:postcommithook
- tsvn:startupdatehook
- tsvn:preupdatehook
- tsvn:postupdatehook
- tsvn:prelockhook
- tsvn:postlockhook

Les paramètres sont les mêmes que si vous configuriez les scripts hook dans la boîte de dialogue de paramètres. Voyez Section 4.31.8, « Scripts hook côté client » pour les détails.

Comme tous les utilisateurs n'ont pas extrait leur copie de travail au même emplacement avec le même nom, vous pouvez configurer un script ou un outil à exécuter situé dans votre copie de travail en spécifiant à la place l'URL dans le dépôt, en utilisant `%REPOROOT%` pour la partie de l'URL correspondant à la racine du dépôt. Par exemple, si votre script hook est dans votre copie de travail sous `contrib/hook-scripts/client-side/checkyear.js`, vous spécifieriez le chemin vers le script comme étant `%REPOROOT%/trunk/contrib/hook-scripts/client-side/checkyear.js`. De cette façon, même si vous déplacez votre dépôt sur un autre serveur, vous n'avez pas besoin de rectifier les propriétés du script hook.

Au lieu de `%REPOROOT%`, vous pouvez également indiquer `%REPOROOT+%`. Le `+` permet d'insérer dans le chemin autant de dossiers que nécessaire pour trouver le script. C'est utile si vous voulez définir votre script de telle sorte que si vous créez une branche, le script reste trouvable, alors même que l'URL de la copie de travail a changé. Dans l'exemple ci-dessus, vous définiriez le chemin vers le script comme `%REPOROOT+%/contrib/hook-scripts/client-side/checkyear.js`.

La capture d'écran suivante montre comment le script de vérification de l'année de copyright dans les en-têtes des fichiers sources est configuré pour TortoiseSVN.

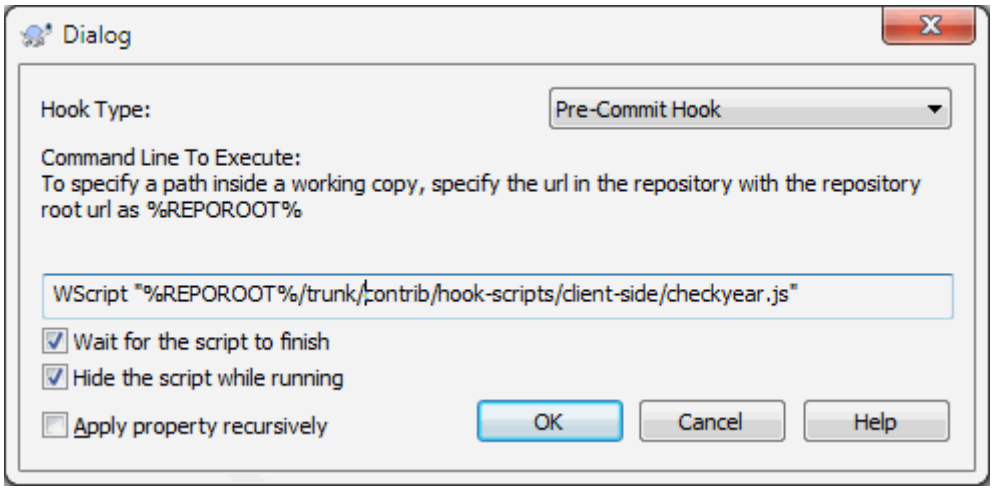


Figure 4.39. Boîte de dialogue de propriétés pour les scripts hook

- Lorsque vous souhaitez ajouter une nouvelle propriété, vous pouvez soit en choisir une dans la liste déroulante, soit ajouter n'importe quel nom de propriété qui vous convient. Si votre projet utilise certaines propriétés personnalisées, et que vous voulez que ces propriétés apparaissent dans la liste déroulante (pour éviter les fautes

de frappe lorsque vous entrez un nom de propriété), vous pouvez créer une liste de vos propriétés personnalisées à l'aide `tsvn:userfileproperties` et `tsvn:userdirproperties`. Appliquez ces propriétés à un dossier. Quand vous éditez les propriétés de n'importe quel élément enfant, vos propriétés personnalisées apparaissent dans la liste des propriétés prédéfinies.

Vous pouvez également spécifier si une boîte de dialogue personnalisée est utilisée pour ajouter ou éditer votre propriété. TortoiseSVN propose quatre modèles de boîtes de dialogue différents, en fonction du type de votre propriété.

bool
: Si votre propriété ne peut avoir que deux états, par exemple true (vrai) et false (faux), vous pouvez la configurer avec un type `booléen`.

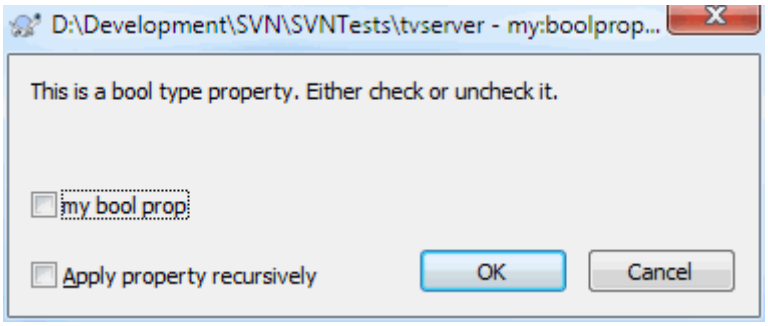


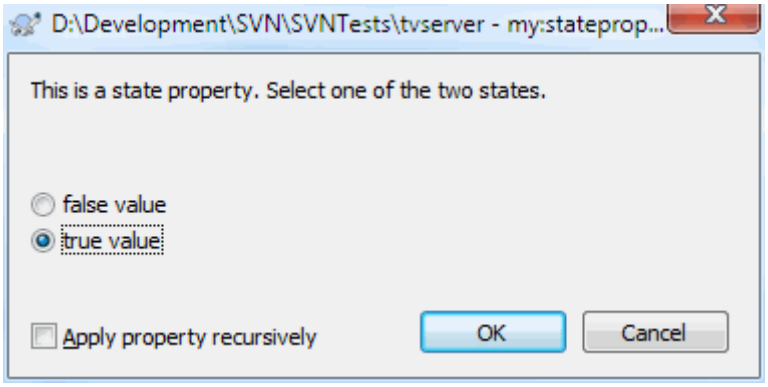
Figure 4.40. Boîte de dialogue de propriété utilisateur de type booléen

Spécifiez votre propriété comme ceci :

```
nom de propriété=bool;texte du label(VALEUROUI;VALEURNON;texte de la case à cocher)
```

Le `texte du label` est le texte affiché dans la boîte de dialogue au-dessus de la case à cocher, dans lequel vous pouvez expliquer l'objet de la propriété et son utilisation. Les autres paramètres devraient être auto-explicatifs.

state
: Si votre propriété représente un état parmi plusieurs possibles, par exemple `oui, non, peut-être`, alors vous pouvez configurer votre propriété avec un type `état`.



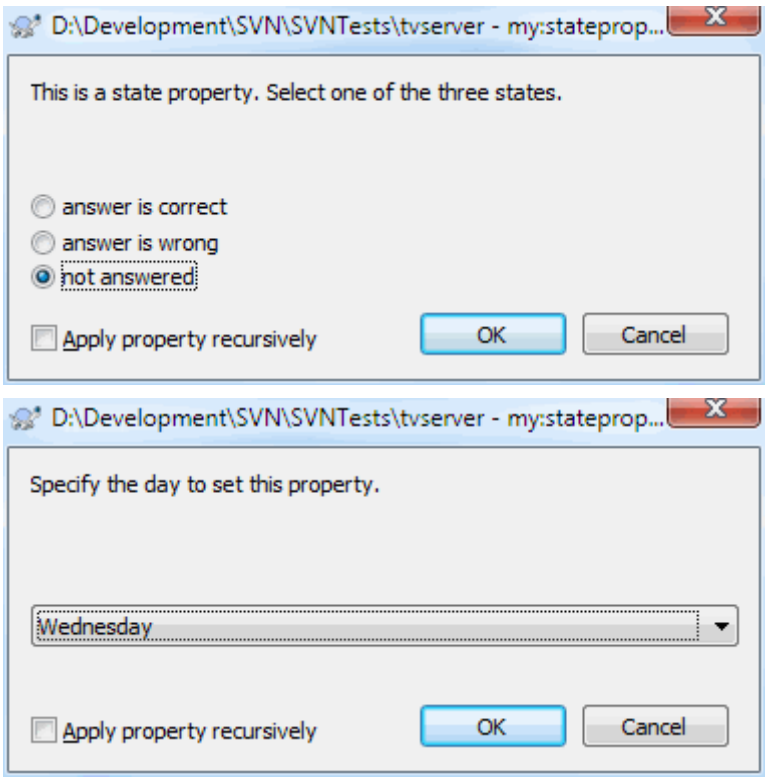


Figure 4.41. Boîtes de dialogue de propriété utilisateur de type état

Spécifiez votre propriété comme ceci :

```
nom de propriété=state;texte du label(VALDEF;VAL1;TEXTE1;VAL2;TEXTE2;VAL3;TEXTE3;...)
```

Les paramètres sont les mêmes que pour la propriété `booléenne`, et `VALDEF` est la valeur par défaut à utiliser si la propriété n'est pas encore définie ou a une valeur qui n'est pas configurée.

Jusqu'à trois valeurs différentes, la boîte de dialogue affiche des boutons radio. S'il y a plus de valeurs configurées, elle utilise une liste déroulante dans laquelle l'utilisateur peut sélectionner l'état désiré.

singleline
Pour les propriétés qui consistent en une seule ligne de texte, utilisez le type de propriété `ligne unique` :

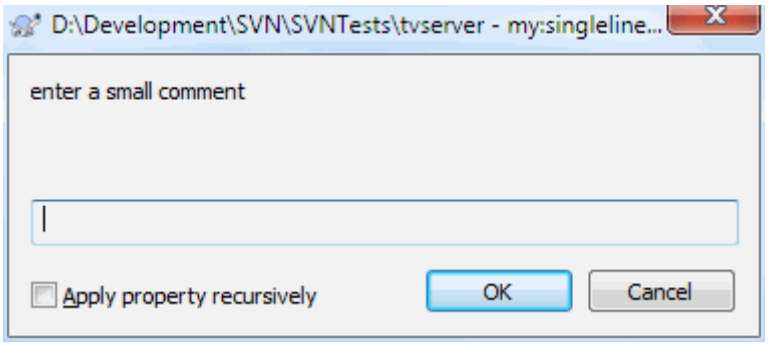


Figure 4.42. Boîte de dialogue de propriété utilisateur de type ligne unique

```
nom de propriété=singleline;texte du label(regex)
```

La `regex` spécifie une expression régulière qui est utilisée pour valider (par correspondance) le texte que l'utilisateur a saisi. Si le texte ne correspond pas avec la regex, alors une erreur est retournée à l'utilisateur et la propriété n'est pas positionnée.

multiline
Pour des propriétés qui consistent en plusieurs lignes de texte, utilisez le type de propriété `multi-lignes` :

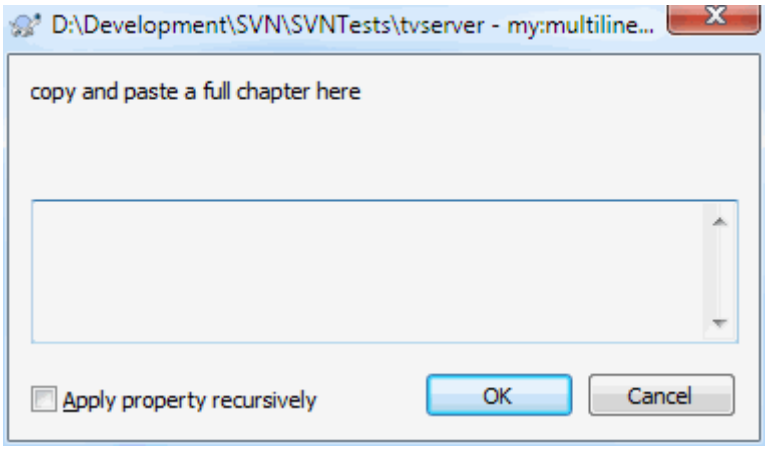


Figure 4.43. Boîte de dialogue de propriété utilisateur de type multi-lignes

```
nom de propriété=multiline;texte du label(regex)
```

La `regex` spécifie une expression régulière qui est utilisée pour valider (par correspondre) le texte que l'utilisateur a saisi. N'oubliez pas d'inclure le caractère de retour à la ligne (\n) dans la regex !

Les captures d'écran ci-dessus ont été faites avec les `tsvn:userdirproperties` suivantes :

```
my:boolprop=bool;This is a bool type property. Either check or uncheck it.(true;false;
my:stateprop1=state;This is a state property. Select one of the two states.(true;true;
my:stateprop2=state;This is a state property. Select one of the three states.(maybe;tr
my:stateprop3=state;Specify the day to set this property.(1;1;Monday;2;Tuesday;3;Wedne
my:singlelineprop=singleline;enter a small comment(.*)
my:multilineprop=multiline;copy and paste a full chapter here(.*)
```

TortoiseSVN peut s'intégrer avec certains outils de suivi de bogue. Cela utilise des propriétés qui commencent par `bugtraq:`. Lisez Section 4.29, « Intégration avec des systèmes de gestion de bug / gestion d'incidents » pour plus d'informations.

Il peut aussi s'intégrer avec certains explorateurs de dépôt en ligne, en utilsant des propriétés qui commencent par `webviewer:`. Lisez Section 4.30, « Intégration avec des visionneuses de dépôt web » pour plus d'informations.

Positionner les propriétés de projet dans les dossiers

Ces propriétés de projet spéciales doivent être appliquées aux *dossiers* pour que le système fonctionne. Lorsque vous utilisez une commande TortoiseSVN qui utilise ces propriétés, celles-ci sont lues dans le dossier où vous avez cliqué. S'il n'y trouve pas les propriétés, TortoiseSVN les cherchera en remontant l'arborescence jusqu'à ce qu'il atteigne un répertoire non versionné, ou la racine d'un lecteur (par exemple `C:\`). Si vous êtes certain que les utilisateurs extraient seulement à partir, par exemple, de `trunk/` et pas d'un sous-dossier, alors il suffit de positionner les propriétés sur `trunk/`. Au contraire, si vous n'êtes pas certain, il est recommandé de positionner les propriétés récursivement sur chaque sous-répertoire. Si vous positionnez la même propriété avec des valeurs

différentes à différents niveaux de la hiérarchie du projet, vous obtiendrez des résultats différents par rapport à l'emplacement dans la structure de répertoire.

Pour les propriétés de projet *uniquement*, c'est-à-dire `tsvn:`, `bugtraq:` et `webviewer:` vous pouvez utiliser la case à cocher Récursif pour appliquer la propriété à tous les sous-dossiers, sans avoir à la mettre aussi sur tous les fichiers.

Lorsque vous ajoutez un nouveau sous-dossier à une copie de travail via TortoiseSVN, toutes les propriétés de projet du dossier parent seront automatiquement ajoutées à ce nouveau dossier fils.

Limitations de l'utilisation de l'explorateur de dépôt

Récupérer des propriétés à distance est une opération lente, ainsi certaines des caractéristiques décrites ci-dessus ne fonctionnent pas dans l'explorateur de dépôt comme elles le font dans une copie de travail.

- Lorsque vous ajoutez une propriété en utilisant l'explorateur de dépôt, seules les propriétés standard `svn:` sont affichées dans la liste pré-définie. Tout autre nom de propriété doit être saisi manuellement.
- Les propriétés ne peuvent pas être renseignées ou supprimées récursivement via l'explorateur de dépôt.
- Les propriétés du projet *ne seront pas* propagées automatiquement quand un dossier enfant est ajouté en utilisant l'explorateur de dépôt.
- `tsvn:autoprops` *ne* renseignera *pas* les propriétés des fichiers ajoutés grâce à l'explorateur de dépôt.

Attention

Bien que les propriétés de projet TortoiseSVN soient extrêmement utiles, elles fonctionnent uniquement avec TortoiseSVN, et certaines ne fonctionnent que dans les versions les plus récentes de TortoiseSVN. Si les personnes qui travaillent sur votre projet utilisent une variété de clients Subversion, ou disposent peut-être d'anciennes versions de TortoiseSVN, vous devriez utiliser des hooks sur le dépôt pour faire appliquer les politiques du projet. Les propriétés de projet ne peuvent que contribuer à mettre en œuvre une politique, elles ne peuvent pas l'imposer.

4.18.3. Éditeurs de propriétés

Certaines propriétés doivent utiliser des valeurs spécifiques, ou être formatées d'une manière spécifique afin d'être utilisées pour l'automatisation. Pour obtenir plus facilement le formatage correct, TortoiseSVN présente pour certaines propriétés particulières des boîtes de dialogue d'édition qui montrent les valeurs possibles ou qui décomposent chaque composant de la propriété.

4.18.3.1. Contenu externe

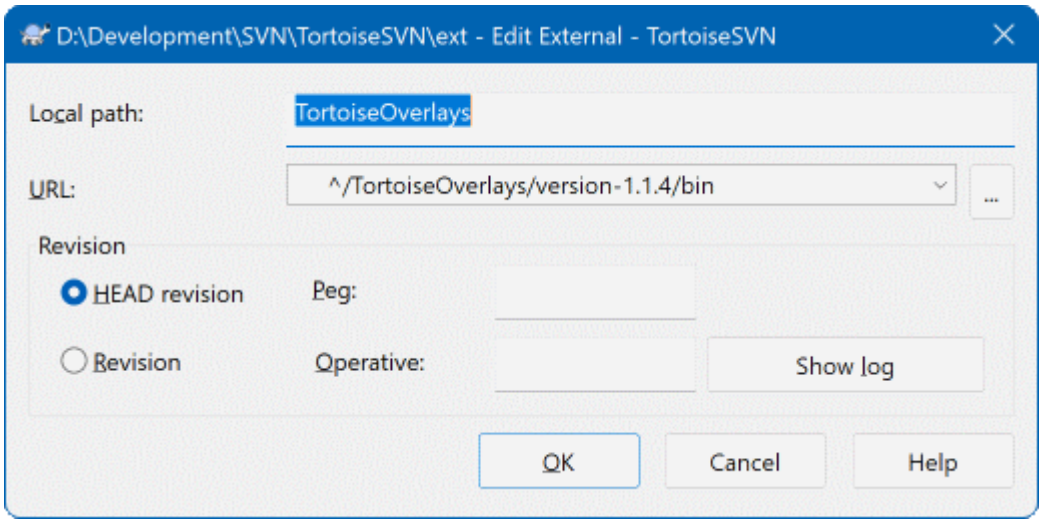


Figure 4.44. Page de propriété svn:externals

La propriété `svn:externals` peut être utilisée pour intégrer d'autres projets du même dépôt ou d'un dépôt complètement différent comme décrit dans Section 4.19, « Éléments externes » .

Vous devez définir le nom du sous-dossier sous lequel le dossier externe est extrait, et l'URL Subversion de l'élément externe. Vous pouvez extraire un externe à sa révision HEAD, de sorte que quand l'élément externe change dans le dépôt, votre copie de travail obtiendra ces changements à la mise à jour suivante. Cependant, si vous voulez que l'externe fasse référence à une version stable particulière, alors vous pouvez spécifier la révision spécifique à utiliser. Dans ce cas, vous voudrez peut-être aussi définir cette même révision comme révision pivot. Si l'élément externe est renommé ultérieurement, alors Subversion ne pourra pas mettre à jour cet élément dans votre copie de travail. En spécifiant une révision pivot, vous dites à Subversion de rechercher un élément qui portait ce nom à la révision pivot plutôt qu'à la révision HEAD.

Le bouton Trouver la révision HEAD récupère la révision HEAD de toutes les URL externes et affiche cette révision HEAD dans la colonne de droite. Une fois que la révision HEAD est connue, un simple clic droit sur un externe vous propose la commande pour définir leur révision HEAD explicite comme révision pivot des externes sélectionnés. Si la révision HEAD n'est pas encore connue, la commande avec le clic droit commencera par récupérer la révision HEAD.

4.18.3.2. Mots-clés SVN

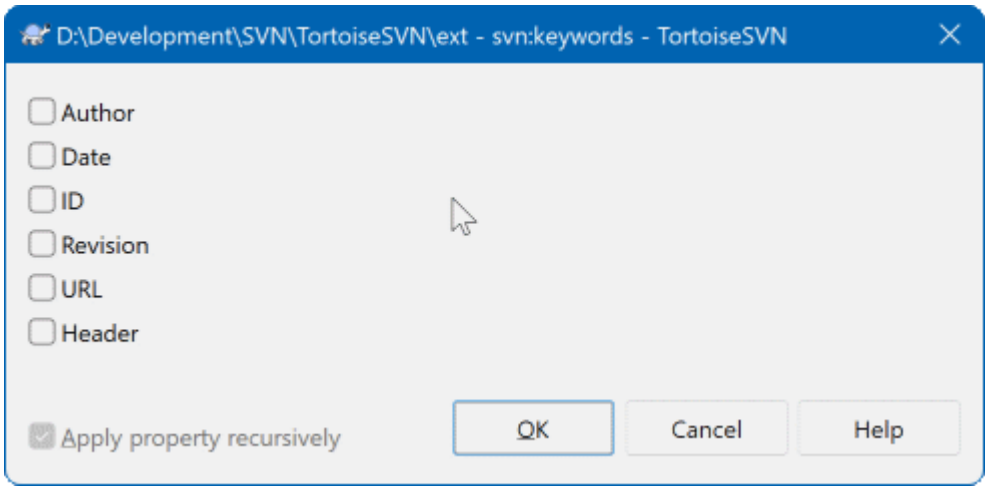


Figure 4.45. Page de propriété svn:keywords

Sélectionnez les mots-clés que vous voudriez voir développés dans votre fichier.

4.18.3.3. Style de caractère de fin de ligne

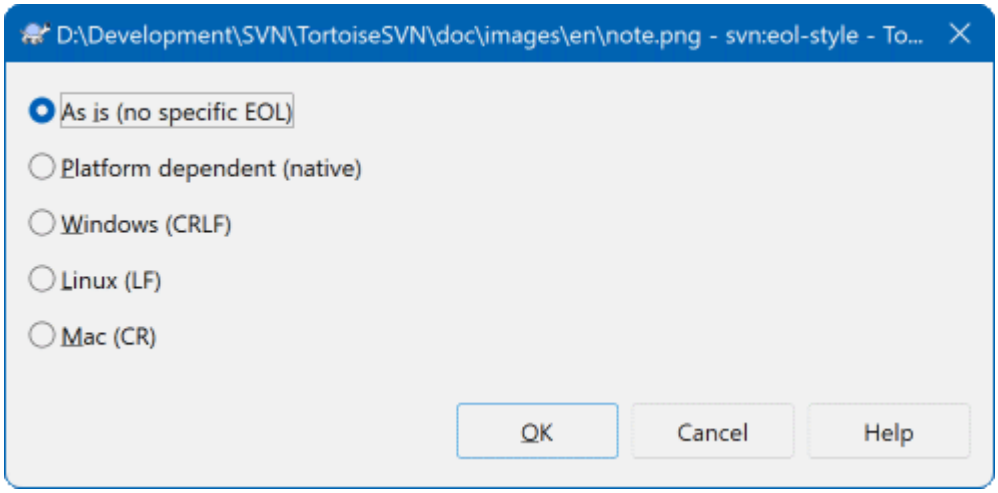


Figure 4.46. Page de propriété svn:eol-style

Sélectionnez le style de fin de ligne que vous souhaitez utiliser et TortoiseSVN utilisera la valeur de propriété correcte.

4.18.3.4. Intégration d'un gestionnaire d'incidents

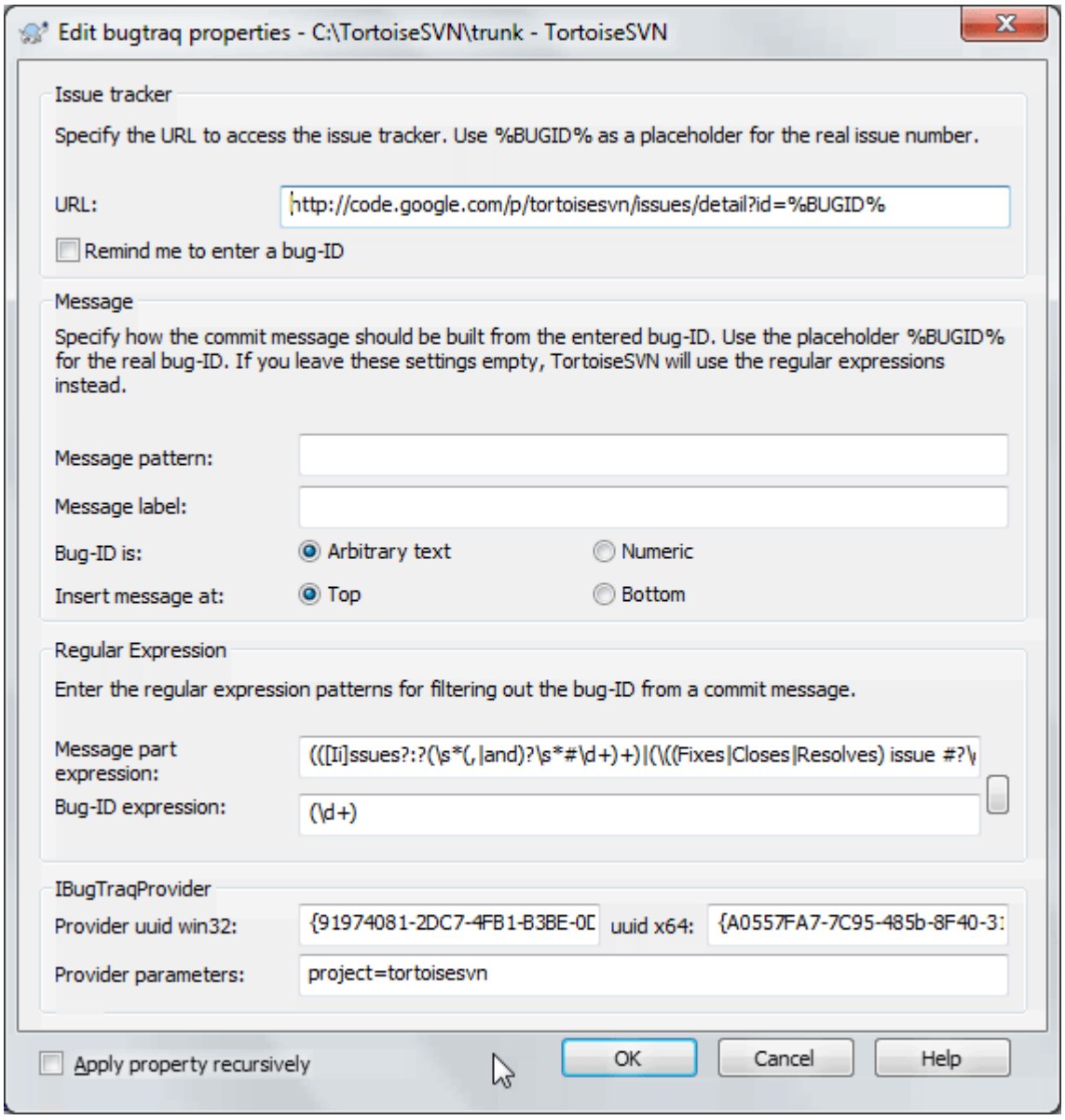


Figure 4.47. Page de propriété tsvn:bugtraq

4.18.3.5. Taille des messages de log

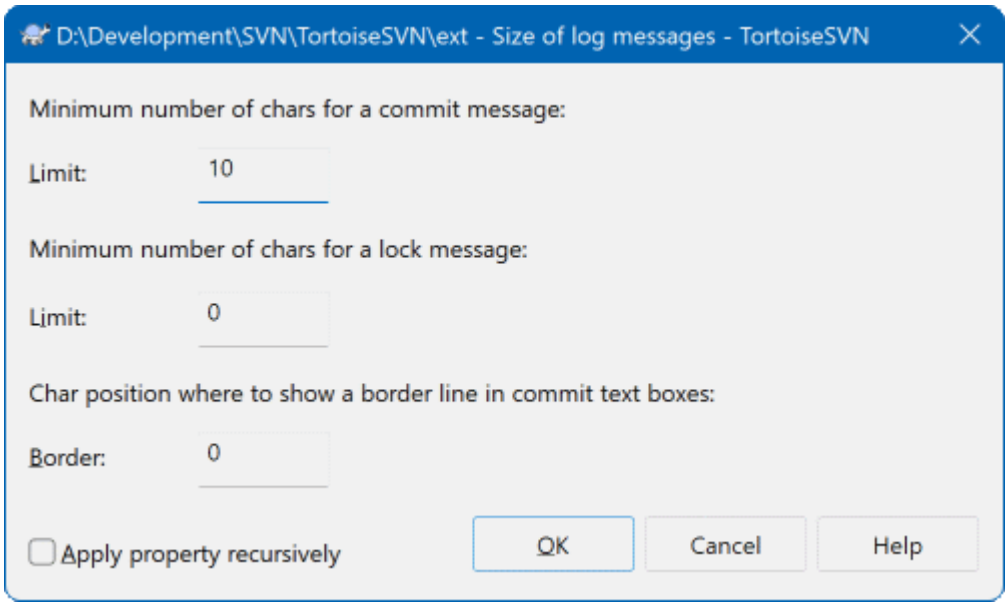


Figure 4.48. Page de propriété de taille des messages de log

Ces 3 propriétés contrôlent le format des messages de log. Les 2 premières désactivent le OK dans les boîtes de dialogue de livraison et de verrouillage jusqu'à ce que le message atteigne la taille minimum. La propriété de position de bordure affiche un marqueur à la largeur de colonne définie pour servir de guide aux projets qui ont des limites de largeur à leurs messages de log. Positionner une valeur à zéro effacera la propriété.

4.18.3.6. Langue de projet

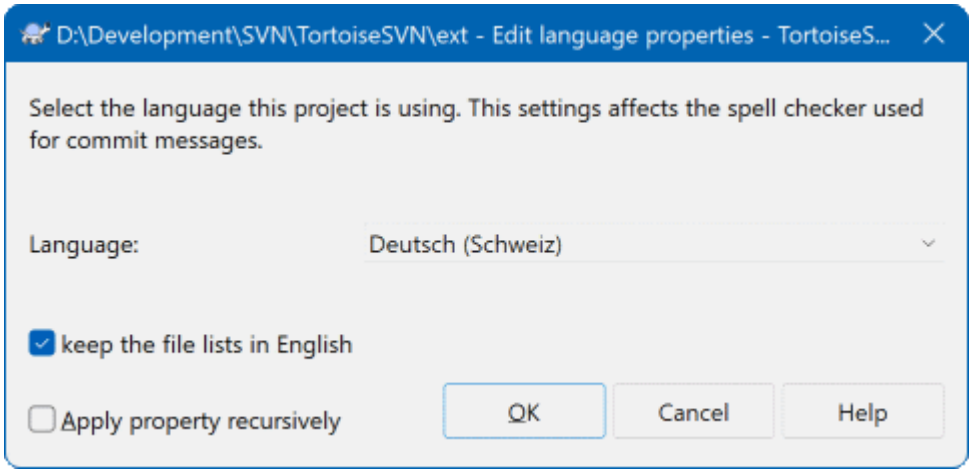


Figure 4.49. Page de propriété de langue

Choisissez la langue à utiliser pour vérifier l'orthographe des messages de log dans la boîte de dialogue de livraison. La case à cocher de listes de fichier entre en compte quand vous faites un clic droit sur le volet de message de log et que vous sélectionnez Coller la liste de fichiers. Par défaut, le statut Subversion sera affiché dans votre langue locale. Quand cette case est cochée, le statut est toujours fourni en anglais, pour les projets qui requièrent des messages de log uniquement en anglais.

4.18.3.7. Type MIME

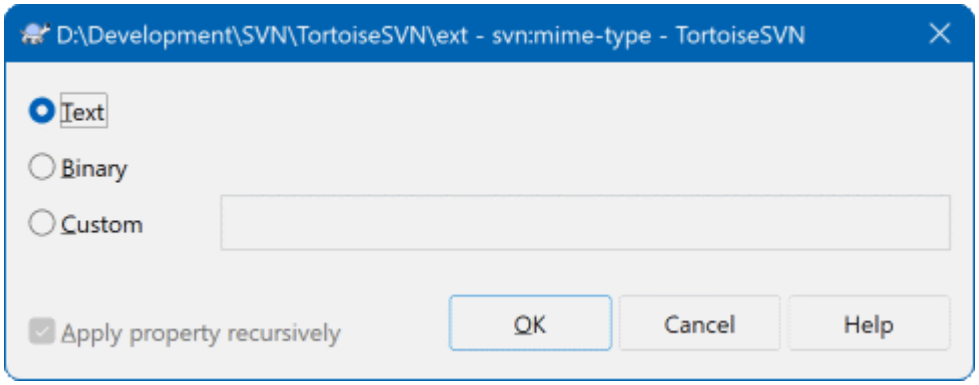


Figure 4.50. Page de propriété svn:mime-type

4.18.3.8. svn:needs-lock

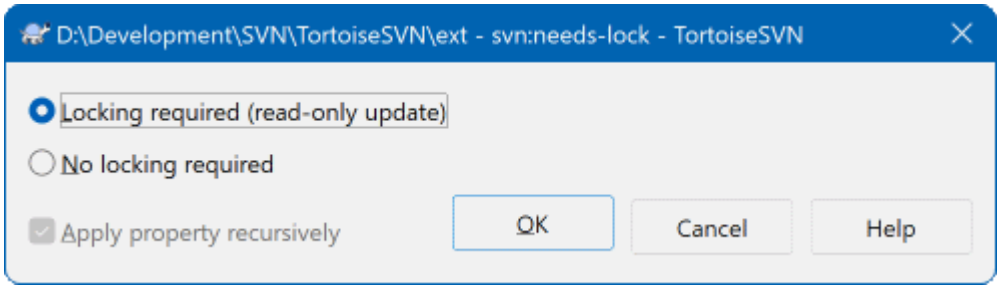


Figure 4.51. Page de propriété svn:needs-lock

Cette propriété contrôle simplement si un fichier sera extrait en mode lecture seule en l'absence d'un verrou pour ce fichier dans la copie de travail.

4.18.3.9. svn:executable

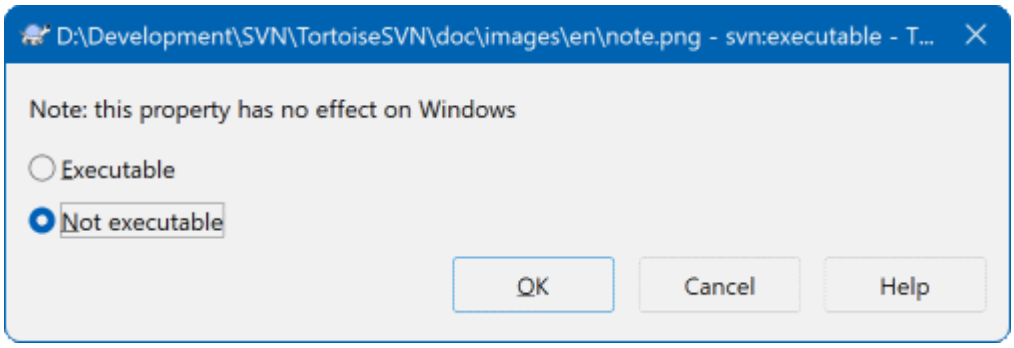


Figure 4.52. Page de propriété svn:executable

Cette propriété contrôle si le statut exécutable sera donné à un fichier lorsqu'il sera extrait sur un système Unix/Linux. Cela n'a pas d'effet sur une extraction sous Windows.

4.18.3.10. Fusionner les modèles de message de journal

Quand des révisions sont fusionnées dans une copie de travail, TortoiseSVN génère un message de log à partir de toutes les révisions fusionnées. Ces messages sont alors disponibles depuis le bouton Messages récents dans la boîte de dialogue de livraison.

Vous pouvez personnaliser le message généré avec les propriétés suivantes :

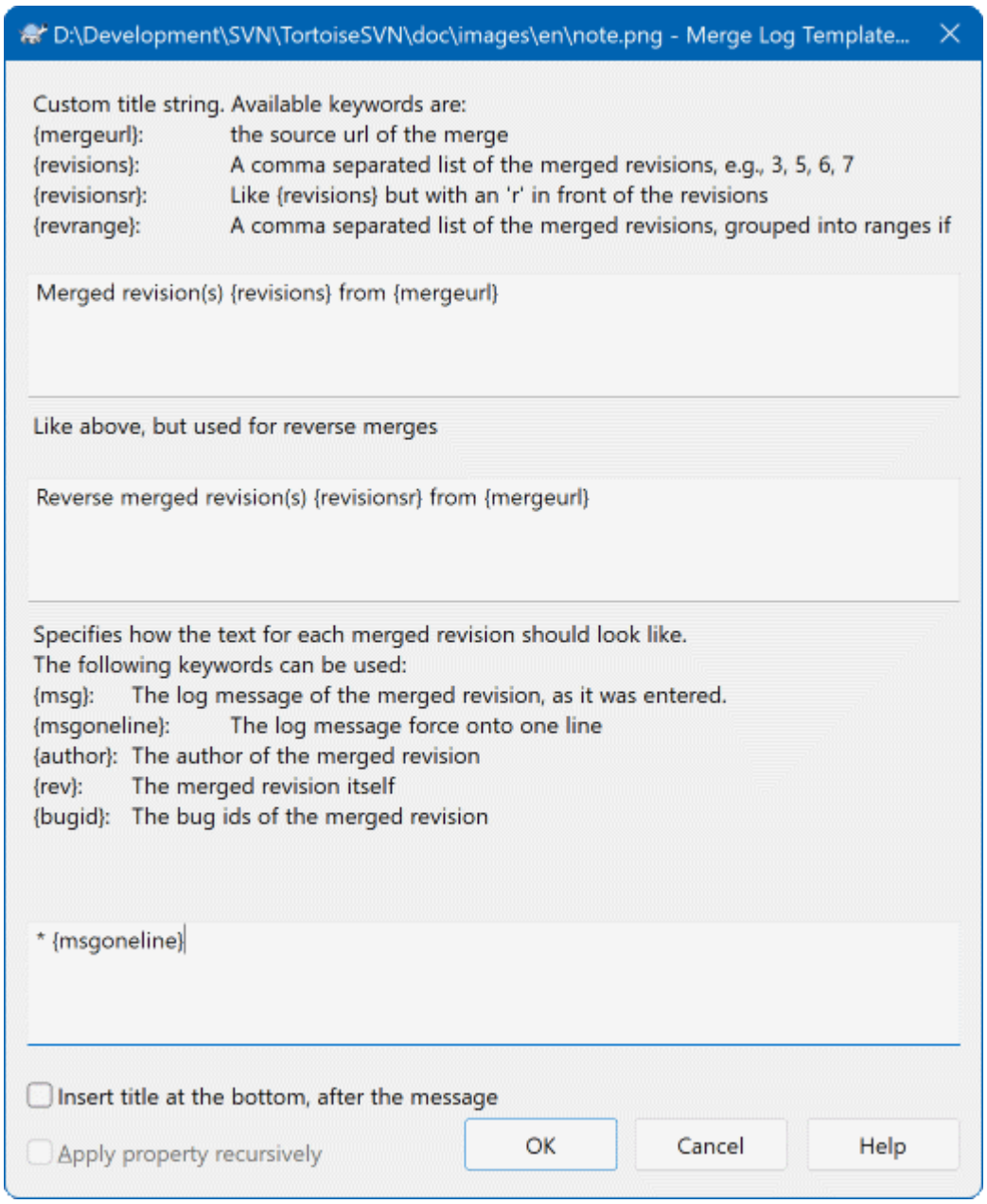


Figure 4.53. Boîte de dialogue de propriété de modèles de message de fusion

tsvn:mergelogtemplatetitle, tsvn:mergelogtemplatereversetitle
: Cette propriété spécifie la première partie du message de journal généré. On peut utiliser les mots-clés suivants :

{revisions}
: Une liste séparée par des virgules des révisions fusionnées, par exemple `3, 5, 6, 7`

{revisionsr}
: Comme `{revisions}`, mais avec chaque révision précédée d'un `r`, par exemple `r3, r5, r6, r7`

{revrange}
: Une liste séparée par des virgules des révisions fusionnées, regroupées dans des plages quand c'est possible, par exemple `3, 5-7`

{mergeurl}
: L'URL source de la fusion, c'est-à-dire à partir de laquelle les révisions sont fusionnées.

La valeur par défaut pour cette chaîne est `Merged revision(s) {revrange} from {mergeurl}:` avec un retour à la ligne à la fin.

tsvn:mergelogtemplatemsg
Cette propriété spécifie à quoi devrait ressembler le texte pour chaque révision fusionnée. On peut utiliser les mots-clés suivants :

{msg}
Le message de journal de la révision fusionnée, tel qu'il a été saisi.

{msgoneline}
Comme `{msg}`, mais tous les retours à la ligne sont remplacés par des espaces, de sorte que l'ensemble du message de journal apparaît sur une seule ligne.

{author}
L'auteur de la révision fusionnée.

{rev}
La révision fusionnée elle-même.

{bugids}
Les ID de bogues de la révision fusionnée, s'il en existe.

tsvn:mergelogtemplatemsgtitlebottom
Cette propriété spécifie la position de la chaîne titre spécifiée dans la propriété `tsvn:mergelogtemplatetitle` ou `tsvn:mergelogtemplatereversetitle`. Si la propriété est positionnée à `yes` (oui) ou `true` (vrai), alors la chaîne titre est ajoutée à la fin au lieu du début.

Important

Cela ne fonctionne que si les révisions fusionnées sont déjà dans le cache du journal. Si vous avez désactivé le cache du journal ou que vous n'avez pas affiché le journal avant la fusion, le message généré ne contiendra aucune information sur les révisions fusionnées.

4.19. Éléments externes

Il est parfois utile de construire une copie de travail à partir 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 dépôt, ou peut-être même de dépôts différents. Si vous voulez que chaque utilisateur ait la même organisation, vous pouvez définir les propriétés `svn:externals` de façon à positionner les ressources spécifiées dans les emplacements cibles souhaités.

4.19.1. Répertoires externes

Disons que vous avez extrait une copie de travail de `/projet1` dans `D:\dev\projet1`. Sélectionnez le dossier `D:\dev\projet1`, faites un clic droit et choisissez menu Windows → Propriétés dans le menu contextuel. La boîte de dialogue de Propriétés apparaît. Allez ensuite sur l'onglet Subversion. Ici, vous pouvez positionner des propriétés. Cliquez sur Propriétés.... Dans la boîte de dialogue de propriétés, double-cliquez sur le `svn:externals` s'il existe déjà, ou cliquez sur le bouton Nouveau... et sélectionnez `Références externes` dans le menu. Pour ajouter un nouvel externe, cliquez sur Nouveau... et saisissez les informations requises dans la boîte de dialogue.

Attention

Les URLs doivent être correctement échappées pour fonctionner, par exemple vous devez remplacer chaque espace par `%20`.

Si vous souhaitez que le chemin local contienne des espaces ou d'autres caractères spéciaux, vous pouvez les encadrer avec des guillemets doubles, ou vous pouvez utiliser le caractère `\` (antislash) comme caractère d'échappement avant un caractère spécial, comme dans le shell Unix. 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.

Utilisez des numéros de révision explicites

Vous devriez sérieusement envisager d'utiliser des numéros de révision explicite dans toutes vos définitions d'externes, comme décrit ci-dessus. En procédant de la sorte, c'est vous qui décidez quand récupérer un nouvel instantané des informations externes, et quel instantané précis récupérer. En plus du principe de bon sens de ne pas vous laisser surprendre par des changements dans des dépôts de tierce partie que vous ne contrôlez peut-être pas, l'utilisation de numéros de révision explicite signifie aussi que si vous voulez récupérer une ancienne révision de votre copie de travail, les définitions de vos externes reviendront elles aussi à ce qu'elles étaient dans cette ancienne révision, ce qui implique aussi que les copies de travail des externes seront mises à jour pour correspondre à l'état dans lequel *elles* étaient, à l'époque où votre dépôt était à cette ancienne révision. Pour des projets logiciels, cela peut représenter la différence entre la réussite et l'échec d'une compilation d'un instantané ancien d'une base de code complexe.

La boîte de dialogue d'édition des propriétés `svn:externals` vous permet de sélectionner les externes et de les positionner explicitement à la révision HEAD de manière automatique.

Si le projet externe est dans le même dépôt, 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 dépôt différent, les changements que vous faites au projet externe seront signalés quand vous livrerez le projet principal, mais vous devrez livrer ces changements externes séparément.

Si vous utilisez des URL absolues dans la valeur de l'option `svn:externals` and que vous relocalisez votre copie de travail (c'est-à-dire si l'URL de votre dépôt change), alors vos références externes ne seront pas mises à jour et risquent de ne plus fonctionner.

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 spécification d'URL relatives sont supportées. Dans les exemples suivants, considérez qu'il y a deux dépôts : un premier à l'adresse `http://example.com/svn/depot-1` et un second à l'adresse `http://example.com/svn/depot-2`. Nous avons une extraction de `http://example.com/svn/depot-1/projet/trunk` dans `C:\Travail` et la propriété `svn:externals` est positionnée sur le trunk.

Relatif au répertoire parent
: Ces URLs commencent toujours par la chaîne `../`, par exemple :

```
../../widgets/toto  commun/toto-widget
```

Cela extraira `http://example.com/svn/depot-1/widgets/toto` dans `C:\Travail\commun\toto-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 dépôt
: Ces URLs commencent toujours par la chaîne `^/`, par exemple :

```
^/widgets/toto  commun/toto-widget
```

Cela extraira `http://example.com/svn/depot-1/widgets/toto` dans `C:\Travail\commun\toto-widget`.

Vous pouvez facilement faire référence à d'autres dépôts avec le même `SVNParentPath` (un répertoire commun qui contient plusieurs dépôts). Par exemple :

```
^/../depot-2/hammers/claw  commun/claw-hammer
```

Cela extraira `http://example.com/svn/depot-2/hammers/claw` dans `C:\Travail\commun\claw-hammer`.

Relatif au schéma
: Les URLs qui commencent par la chaîne `//` copient uniquement la partie schéma de l'URL. C'est utile quand il faut accéder au même nom d'hôte avec des schémas différents suivant l'emplacement réseau ; par exemple, les clients dans l'intranet utilisent `http://` tandis que les clients externes utilisent `svn+ssh://`. Par exemple :

```
//example.com/svn/depot-1/widgets/toto  commun/toto-widget
```

Cela extraira `http://example.com/svn/depot-1/widgets/toto` ou `svn+ssh://example.com/svn/depot-1/widgets/toto` suivant la méthode utilisée pour extraire `C:\Working`.

Relatif au nom de domaine du serveur
: Les URLs qui commencent par la chaîne `/` copient les parties schéma et nom d'hôte de l'URL, par exemple :

```
/svn/depot-1/widgets/toto  commun/toto-widget
```

Cela extraira `http://example.com/svn/depot-1/widgets/toto` dans `C:\Travail\commun\toto-widget`. Mais si vous extrayez votre copie de travail depuis un autre serveur à l'adresse `svn+ssh://un.autre.miroir.net/svn/depot-1/projet1/trunk`, alors la référence externe extraira `svn+ssh://un.autre.miroir.net/svn/depot-1/widgets/toto`.

Si nécessaire, vous pouvez aussi spécifier une révision pivot et une révision opérationnelle pour l'URL. Pour plus d'informations sur les révisions pivot et les révisions opérationnelles, veuillez lire le *chapitre correspondant* [http://svnbook.red-bean.com/fr/1.8/svn.advanced.pegrevs.html] dans le livre de Subversion.

Important

Si vous spécifiez un sous-dossier comme dossier cible de l'externe comme dans les exemples ci-dessus, assurez-vous que *tous* les dossiers intermédiaires soient versionnés aussi. Dans les exemples ci-dessus, le dossier `commun` devrait donc être versionné !

Même si la référence externe fonctionnera correctement dans la plupart des situations quand les répertoires intermédiaires ne sont pas versionnés, il y a certaines opérations qui ne fonctionneront pas comme prévu. Les icônes superposées dans l'explorateur n'indiqueront pas non plus le bon statut.

Si vous avez besoin de plus d'informations sur la façon dont TortoiseSVN manipule les propriétés, lisez Section 4.18, « Configuration des projets ».

Pour découvrir des méthodes différentes d'accéder aux sous-projets communs, lisez Section B.6, « Inclure un sous-projet commun ».

4.19.2. Fichiers externes

À partir de Subversion 1.6, vous pouvez ajouter à votre copie de travail 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 externe doit être un fils immédiat du dossier où vous avez positionné la propriété `svn:externals`.
- L'URL pour un fichier externe doit se trouver dans le même dépôt que l'URL dans laquelle le fichier externe sera inséré ; les fichiers externes inter-dépôt ne sont pas supportés.

Par beaucoup d'aspects, les fichiers externes se comportent comme n'importe quel autre fichier versionné. Néanmoins, ils ne peuvent pas être déplacés ou supprimés avec les commandes standard ; il faut passer par la modification de la propriété `svn:externals`.

4.19.3. Création d'externe via glisser-déposer

Si vous avez déjà une copie de travail des fichiers ou des dossiers que vous voulez inclure en tant que références externes dans une autre copie de travail, vous pouvez simplement les ajouter en faisant un glisser-déposer depuis l'explorateur Windows.

Faites simplement glisser avec le bouton droit le fichier ou le dossier depuis une copie de travail jusqu'à l'emplacement où vous voulez les inclure en tant que références externes. Un menu contextuel apparaît quand vous relâchez le bouton de la souris. Si vous cliquez sur l'élément SVN Ajouter comme référence externe ici de ce menu contextuel, la propriété `svn:externals` est ajoutée automatiquement. Tout ce qu'il vous reste à faire après cela est de livrer les changements de propriété et de mettre à jour pour que ces références externes soient incluses correctement dans votre copie de travail.

4.20. 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 appelée une *branche*. Les branches servent souvent à expérimenter de nouvelles fonctionnalités sans perturber la ligne de développement principale avec des erreurs de compilation et des bogues. Dès que la nouvelle fonctionnalité est assez stable, alors la branche de développement est *fusionnée* vers la branche principale (trunk).

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 ce qu'on appelle des `copies peu coûteuses` à la place. Les copies peu coûteuses sont similaires aux hard links d'Unix, ce qui signifie qu'au lieu de faire une copie complète dans le dépôt, on crée un lien interne qui pointe vers un arbre ou une révision spécifique. En conséquence, les branches et les étiquettes sont très rapides à créer et ne prennent presque aucun espace supplémentaire dans le dépôt.

4.20.1. 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 :

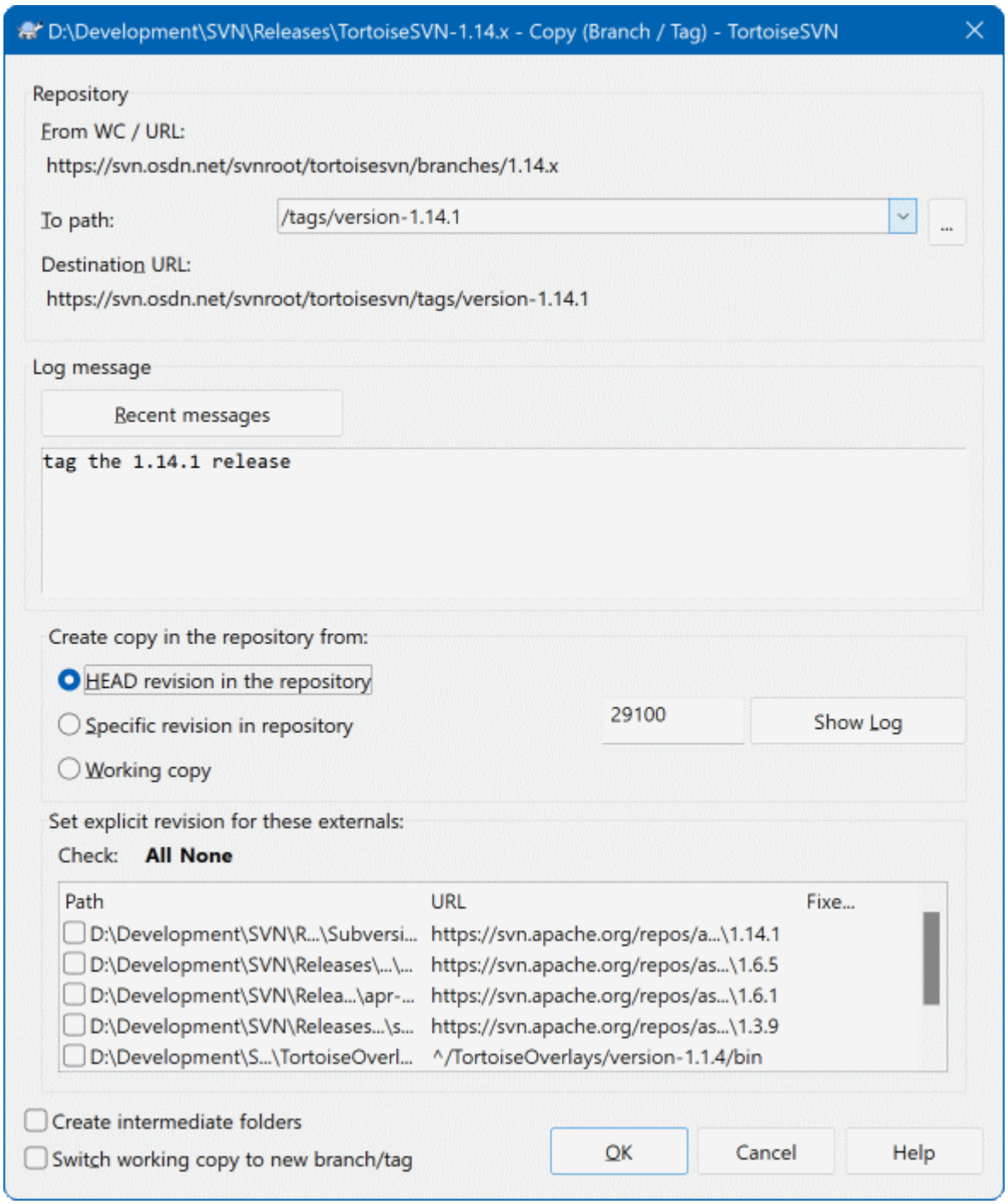


Figure 4.54. La boîte de dialogue Branche/étiquette

Sélectionnez le dossier de votre copie de travail que vous voulez copier dans une branche ou une étiquette, et choisissez ensuite la commande TortoiseSVN → Branche/étiquette....

Par défaut, l'URL de destination de la nouvelle branche sera l'URL source sur laquelle votre copie de travail est basée. Vous devrez éditer cette URL pour indiquer le nouveau chemin vers votre branche/étiquette. Ainsi, au lieu de

```
http://svn.collab.net/depot/NomDeProjet/trunk
```

vous allez peut-être passer à quelque chose comme

```
http://svn.collab.net/depot/NomDeProjet/tags/Release_1.10
```

Si vous ne vous souvenez pas de la convention de nommage que vous avez utilisée la dernière fois, cliquez sur le bouton à droite pour ouvrir l'explorateur de dépôt et voir la structure existante du dépôt.

Dossiers intermédiaires

Quand vous spécifiez l'URL cible, tous les dossiers jusqu'au père doivent déjà exister, ou vous aurez un message d'erreur. Dans l'exemple ci-dessus, l'URL `http://svn.collab.net/depot/NomDeProjet/tags/` doit exister pour pouvoir créer l'étiquette `Release_1.10`.

Cependant, si vous voulez créer une branche/étiquette à une URL dont les dossiers intermédiaires n'existent pas encore, vous pouvez cocher l'option `Créer les dossiers intermédiaires` en bas de la boîte de dialogue. Si cette option est activée, tous les dossiers intermédiaires sont créés automatiquement.

Remarquez que cette option est désactivée par défaut pour éviter les fautes de frappe. Par exemple, si vous tapez comme URL cible `http://svn.collab.net/depot/NomDeProjet/Tags/Release_1.10` au lieu de `http://svn.collab.net/depot/NomDeProjet/tags/Release_1.10`, vous aurez une erreur avec l'option désactivée, mais avec l'option activée un dossier `Tags` sera créé automatiquement, et vous vous retrouverez avec un dossier `Tags` et un dossier `tags`.

Maintenant vous devez choisir la source de la copie. Ici vous avez trois options :

Révision HEAD dans le dépôt
: La nouvelle branche est copiée directement dans le dépôt 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 dépôt
: La nouvelle branche est copiée directement dans le dépôt mais vous pouvez choisir une révision plus ancienne. C'est utile si vous avez oublié de faire une étiquette quand vous avez sorti votre projet la semaine dernière. Si vous ne souvenez pas du numéro de révision, cliquez sur le bouton à droite pour afficher le journal de révision et sélectionner le numéro de révision à partir de là. Là encore, aucune donnée n'est transférée depuis 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 dépôt 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.

Si votre copie de travail contient d'autres projets incluant des propriétés `svn:externals`, ces ressources externes seront énumérées au bas de la fenêtre de branche/étiquette. Pour chaque ressource externe, le chemin cible et l'URL source sont affichés.

Si vous voulez vous assurer que la nouvelle étiquette est toujours dans un état cohérent, vérifiez toutes les références externes pour que leurs révisions soient épinglées. Si vous ne vérifiez pas les références externes et que ces références externes pointent vers une révision HEAD qui peut changer dans l'avenir, extraire la nouvelle étiquette extraira cette révision HEAD de la référence externe et votre étiquette pourrait ne plus compiler. Donc c'est toujours une bonne idée de positionner les références externes à une révision explicite en créant une étiquette.

Les éléments externes sont automatiquement épinglé soit à la révision courante de tête soit à la révision de travail de base, selon la source de la branche/tag :

Copier la source	Révision épinglée
Révision HEAD dans le dépôt	révision HEAD du dépôt externe
Révision spécifique dans le dépôt	révision HEAD du dépôt externe
Copie de travail	révision de BASE de la CdT de la référence externe

Tableau 4.1. Révision épinglée

Références externes avec références externes

Si un projet qui est inclus comme référence externe a lui-même des références externes, alors celles-ci ne seront pas étiquetées ! Seules les références externes qui sont les enfants directs peuvent être étiquetées.

Appuyez sur OK pour livrer la nouvelle copie dans le dépôt. N'oubliez pas de mettre un commentaire. Notez que la copie est créée *dans le dépôt*.

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 trunk, donc votre CdT peut toujours être marquée comme modifiée par rapport au trunk.

4.20.2. Autres manières de créer une branche ou une étiquette

Vous pouvez aussi créer une branche ou une étiquette sans avoir de copie de travail. Pour ce faire, ouvrez l'explorateur de dépôt. Vous pouvez faire glisser des répertoires vers un nouvel endroit. Pour créer une copie, vous devez garder la touche **Ctrl** appuyée lorsque vous faites glisser le répertoire, sinon celui-ci n'est pas copié, mais déplacé.

Vous pouvez également faire glisser un dossier avec le bouton droit de la souris. Dès que vous relâchez le bouton, un menu contextuel s'affiche pour vous permettre de choisir entre la copie ou le déplacement du dossier. Bien sûr, pour créer une branche ou une étiquette, vous devez copier le dossier, pas le déplacer.

Il y a une autre façon de faire dans la boîte de dialogue de journal. Vous pouvez afficher la boîte de dialogue de journal, par exemple du trunk, sélectionner une révision (soit la révision HEAD tout en haut, soit une révision antérieure), faire un clic droit et choisir Créer une branche/étiquette depuis une révision.

4.20.3. Extraire ou aller sur...

... telle (n')est (pas) la question. Tandis qu'une extraction extrait tout de la branche désirée du dépôt dans votre répertoire de travail, TortoiseSVN → Aller sur... transfère seulement les données modifiées dans votre copie de travail. Bon pour la charge réseau, bon pour votre patience. :-)

Pour pouvoir travailler avec votre branche ou votre étiquette récemment générée, vous avez plusieurs méthodes. Vous pouvez :

- TortoiseSVN → Extraire pour faire une extraction propre dans un dossier vide. Vous pouvez extraire vers n'importe quel emplacement sur votre disque local et vous pouvez créer autant de copies de travail de votre dépôt que vous le souhaitez.
- Basculer votre copie de travail courante vers la copie nouvellement créée dans le dépôt. 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 trunk et une branche, mais sans le coût d'une nouvelle extraction, vous pouvez utiliser l'explorateur Windows pour copier votre extraction du trunk dans un autre dossier, puis utiliser la commande TortoiseSVN → Aller sur... sur cette copie pour en faire la branche.

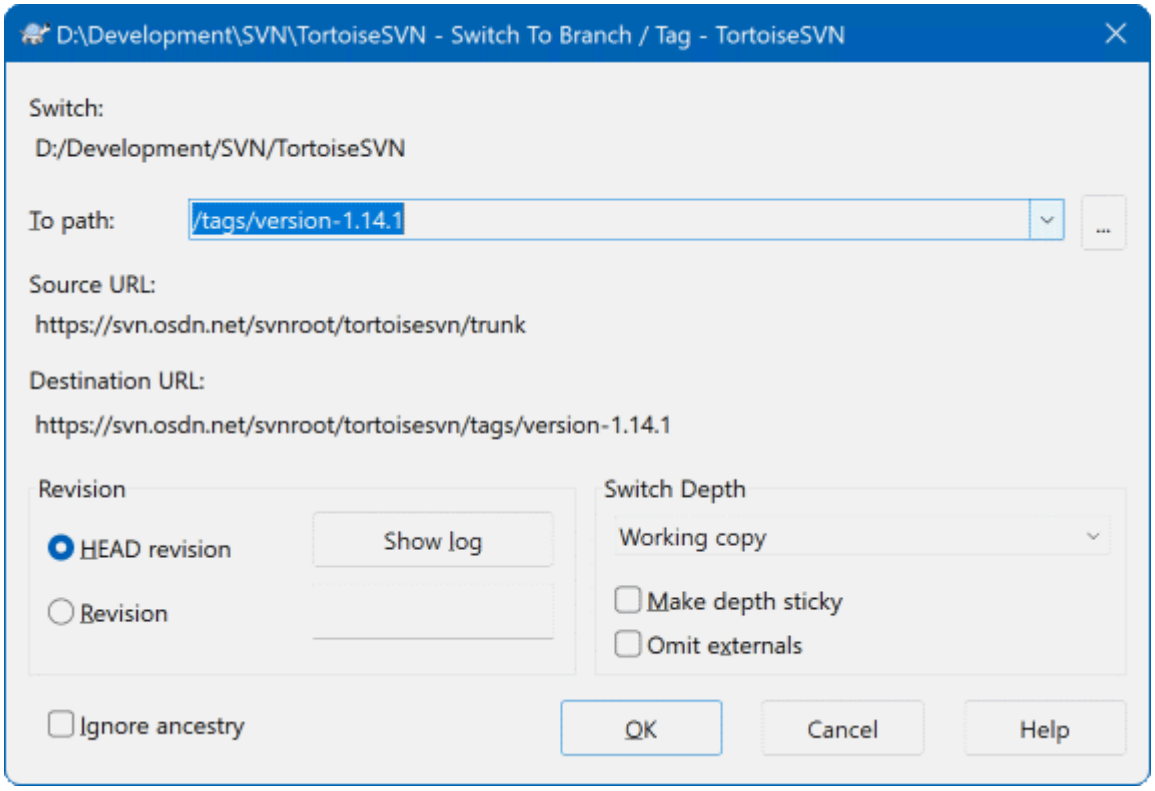


Figure 4.55. La boîte de dialogue Aller sur

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 instantané du projet à une étape particulière. Elles ne sont normalement pas utilisées pour du développement — c'est ce à quoi servent les branches, et c'est la raison pour laquelle nous avons recommandé la structure de dépôt `/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 qui vous empêche de le faire accidentellement. Cependant, si vous essayez de livrer vers un chemin dans le dépôt 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 livrés sur la nouvelle branche et *pas* sur le trunk. Seules les modifications sont stockées. Le reste demeure une copie peu coûteuse.

4.21. Fusionner

Quand vous utilisez des branches pour maintenir des lignes de développement séparées, il y aura un moment où vous voudrez fusionner les changements faits sur une branche vers le trunk, ou vice versa.

Il est important de comprendre comment les embranchements et les fusions fonctionnent dans Subversion avant de commencer à les utiliser, car ils peuvent devenir assez complexes. Il est fortement recommandé de lire le chapitre *Gestion des branches* [http://svnbook.red-bean.com/fr/1.8/svn.branchmerge.html] dans le livre de Subversion, qui donne une description complète et de nombreux exemples sur la façon de l'utiliser.

Il faut aussi noter que fusionner se fait *toujours* à l'intérieur d'une copie de travail. Si vous voulez fusionner des modifications *vers* une branche, vous devez avoir une copie de travail extraite pour la branche, 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 y compris 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 trunk) et vous voulez reporter ces changements vers une autre branche.

  Ce que vous demandez à Subversion de faire est : « Calcule les changements nécessaires pour passer [DE] la révision 1 de la branche A [À] la révision 7 de la branche A, et applique ces changements à ma copie de travail (du trunk ou de la branche B) ».

  Si vous laissez vide la plage de révisions, Subversion utilise les fonctions de suivi de fusion pour calculer la bonne plage de révisions à utiliser. Cela s'appelle une réintégration ou une fusion automatique.

Fusionner deux arborescences différentes
: C'est une généralisation de la méthode de réintégration. Ce que vous demandez à Subversion de faire est : « Calcule les changements nécessaires pour passer [DE] la révision HEAD du trunk [À] la révision HEAD de la branche, et applique ces changements à ma copie de travail (du trunk) ». Le résultat est que le trunk a maintenant la même apparence que la branche.

  Si votre serveur ou votre dépôt ne supporte pas le suivi de fusion, alors c'est la seule façon de fusionner une branche vers le trunk. Un autre cas d'utilisation se produit quand vous utilisez les branches d'un fournisseur et que vous avez besoin de fusionner les changements dans le code de votre trunk à la suite d'une nouvelle publication du fournisseur. Pour plus d'informations, lisez le chapitre sur les *branches fournisseurs* [http://svnbook.red-bean.com/fr/1.8/svn.advanced.vendorbr.html] dans le livre de Subversion.

4.21.1. Fusionner une plage de révisions

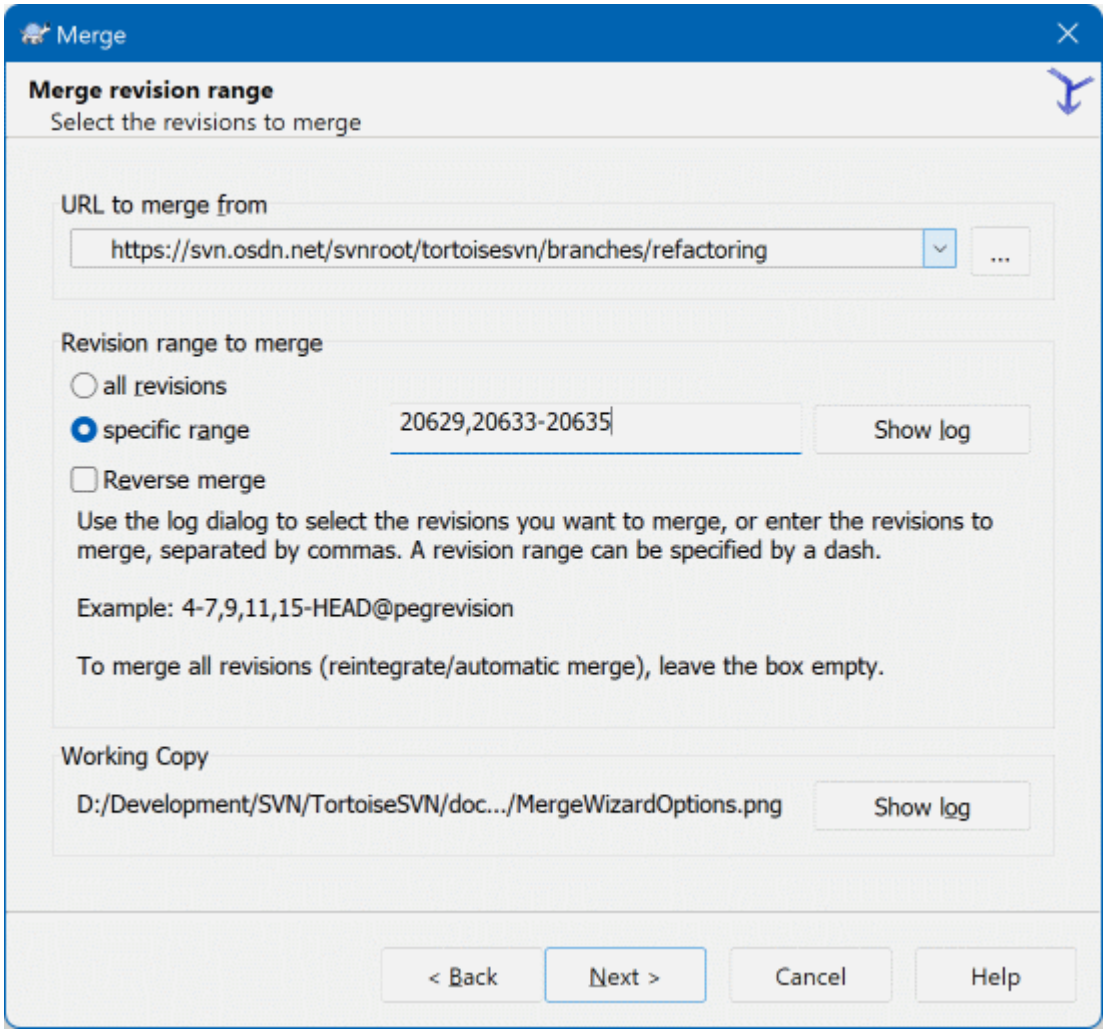


Figure 4.56. Assistant de fusion — Sélectionner une plage de révisions

Dans le champ De : entrez l'URL complète du dossier de la branche ou de l'étiquette contenant les changements que vous voulez reporter dans votre copie de travail. Vous pouvez aussi cliquer sur ... pour parcourir le dépôt et trouver la branche désirée. Si vous avez déjà fusionné depuis cette branche, alors utilisez simplement la liste déroulante contenant l'historique des URLs déjà utilisées.

Si vous fusionnez depuis une branche renommée ou effacée, alors vous devrez retourner à une révision où cette branche existait encore. Dans ce cas, vous devez aussi spécifier cette révision comme révision pivot dans la plage des révisions fusionnées (voir ci-dessous), sinon la fusion échouera quand elle ne pourra pas trouver ce chemin à la révision HEAD.

Dans le champ texte Plage de révisions à fusionner entrez la liste des révisions à fusionner. Ce peut être une seule révision, une liste de révisions spécifiques séparées par des virgules, une plage de révisions séparées par un tiret, ou une combinaison de ces différentes notations.

Si vous avez besoin de spécifier une révision pivot pour la fusion, ajoutez la révision pivot à la fin des révisions, ex. `:5-7,10@3`. Dans l'exemple ci-dessus, les révisions 5,6,7 et 10 seront fusionnées, avec 3 comme révision pivot.

Important

Dans TortoiseSVN, il y a une différence importante dans la manière dont est spécifiée une plage de révisions par rapport au client en ligne de commande. La façon la plus simple de visualiser cela est de penser à une clôture avec des poteaux et des panneaux.

> Avec le client en ligne de commande, vous spécifiez les modifications à fusionner avec deux révisions « poteaux de clôture » qui spécifient les bornes *avant* et *après*.
>
> Avec TortoiseSVN, vous spécifiez la liste des modifications à fusionner en utilisant des « panneaux de clôture ». La raison devient claire lorsque vous utilisez la boîte de dialogue de journal pour spécifier les révisions à fusionner, où chaque révision apparaît comme une liste de modifications.
>
> Si vous fusionnez des révisions par lots, la méthode décrite dans le livre de Subversion vous fera fusionner 100–200 cette fois et 200–300 la prochaine. Avec TortoiseSVN, vous fusionneriez 100–200 cette fois et 201–300 la prochaine.
>
> Cette différence a fait couler beaucoup d'encre dans les listes de diffusion. Nous reconnaissons les différences avec le client en ligne de commande, mais nous pensons que pour la majorité des utilisateurs de l'interface graphique, il est plus facile de comprendre la notation que nous avons implémentée.

La façon la plus facile de choisir la plage de révisions dont vous avez besoin est de cliquer sur Voir le journal, puisqu'ainsi les changements récents seront affichés avec les commentaires associés. Si vous voulez fusionner les changements d'une seule révision, sélectionnez juste cette révision. Si vous voulez fusionner les changements de plusieurs révisions, alors sélectionnez cette plage (en utilisant comme de coutume la touche **Maj**). Cliquez sur OK et les numéros de révision seront automatiquement insérés.

Si vous voulez *revenir* sur des modifications déjà fusionnées dans votre copie de travail, sélectionnez les révisions à annuler et vérifiez bien que la case Fusion inversée est cochée.

Si vous avez déjà fusionné des changements de cette branche, avec bon espoir vous aurez fait une note de la dernière révision fusionnée dans le commentaire quand vous avez livré la modification. Dans ce cas, vous pouvez utiliser Voir le journal dans la copie de travail pour retrouver ce commentaire. Dans la mesure où nous considérons les révisions comme des listes de modifications, vous devriez utiliser la révision de fin de la dernière fusion comme point de départ pour celle ci. Par exemple, si vous avez fusionné les révisions 37 à 39 la dernière fois, alors le point de départ pour cette fusion devrait être la révision 40.

Si vous vous servez de la fonctionnalité de suivi de fusion de Subversion, vous n'avez pas besoin de vous souvenir des révisions déjà fusionnées — Subversion les enregistrera pour vous. Si vous laissez le champ plage de révisions vide, toutes les révisions qui n'ont pas encore été fusionnées y seront listées. Lisez Section 4.21.5, « Suivi de fusion » pour en savoir plus.

Quand le suivi de fusion est utilisé, la boîte de dialogue de journal montrera les révisions précédemment fusionnées, et les révisions antérieures à l'ancêtre commun, c'est-à-dire avant la copie de la branche, en grisé. La case à cocher Cacher les révisions ne pouvant être fusionnées vous permet de filtrer complètement ces révisions, afin que vous ne voyiez que les révisions qui *peuvent* être fusionnées.

Si d'autres personnes peuvent livrer des changements alors soyez prudents en utilisant 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.

Si vous laissez vide la plage de révisions ou que vous sélectionnez le bouton radio toutes les révisions, alors Subversion fusionne toutes les révisions qui ne l'ont pas encore été. On appelle cela une réintégration ou une fusion automatique.

Il y a quelques prérequis pour pouvoir faire une réintégration. Premièrement, le serveur doit supporter le suivi de fusion. La copie de travail doit contenir toute la profondeur complète de l'arborescence (pas d'extractions partielles), et elle ne doit pas avoir de modifications locales, d'éléments déplacés ou mis à jour à une révision autre que HEAD. Tous les changements faits sur le trunk pendant la phase de développement de la branche doivent avoir été intégrés à la branche (ou marqués comme ayant été fusionnés). La plage de révisions à fusionner sera calculée automatiquement.

Cliquez sur Suivant et aller à Section 4.21.3, « Options de fusion ».

4.21.2. Fusionner deux arborescences différentes

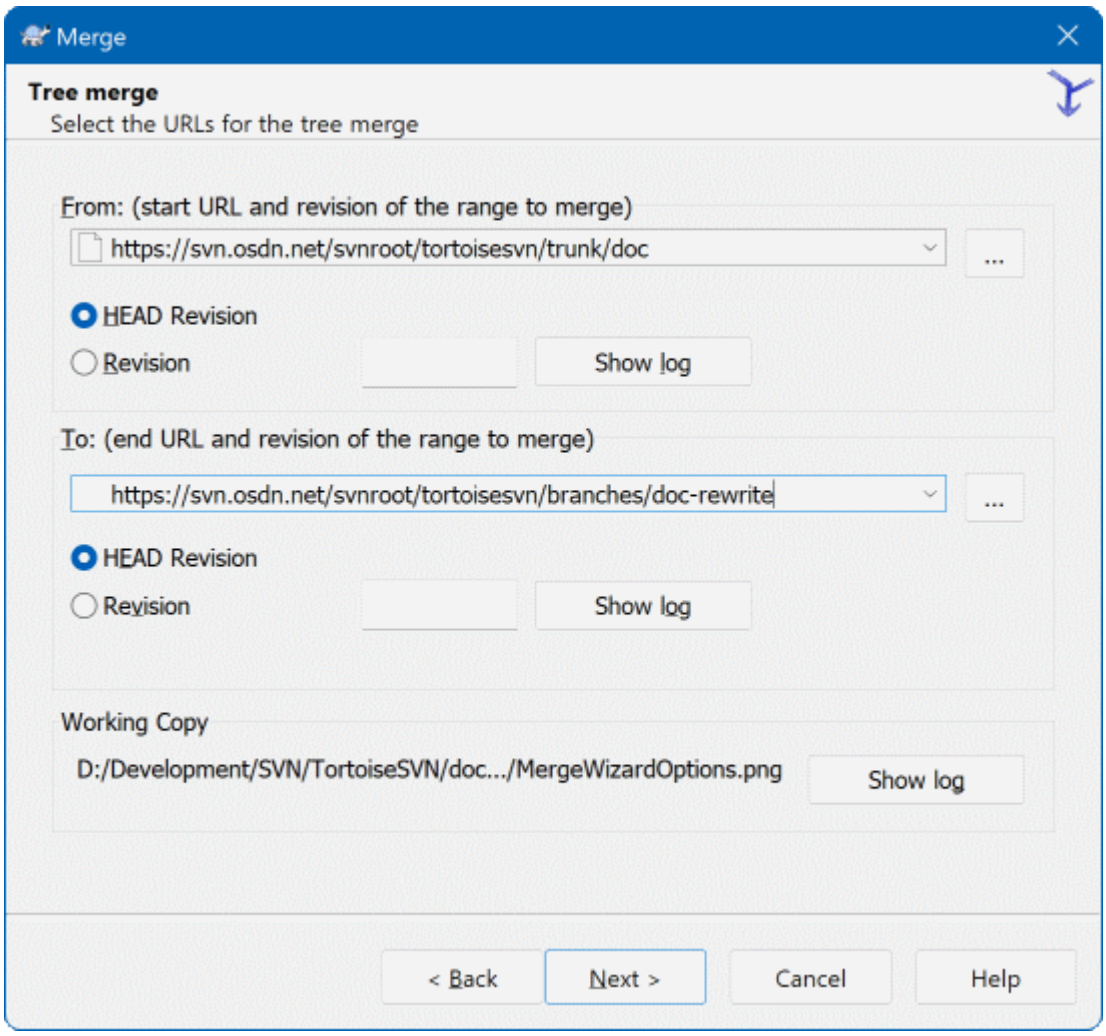


Figure 4.57. Assistant de fusion — fusion d'arborescence

Si vous utilisez cette méthode pour fusionner une branche avec le trunk, vous devez démarrer l'assistant de fusion depuis une copie de travail du trunk.

Dans le champ De : entrez l'URL complète du dossier du *trunk*. Cela peut sembler erroné, mais souvenez-vous que le trunk est l'endroit auquel vous souhaitez ajouter les modifications de la branche. Vous pouvez aussi cliquer sur ... pour parcourir le dépôt.

Remplissez le champ À : avec 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 livraison vous pouvez utiliser la révision HEAD dans les deux cas. S'il y a une chance que quelqu'un d'autre ait fait une livraison depuis cette synchronisation, utilisez le numéro spécifique de la révision pour éviter de perdre des livraisons plus récentes.

Vous pouvez également utiliser Voir le journal pour sélectionner la révision.

4.21.3. Options de fusion

Cette page de l'assistant vous permet de spécifier des options avancées avant de commencer le processus de fusion. La plupart du temps vous pouvez vous contenter d'utiliser les options par défaut.

Vous pouvez donner la profondeur de fusion, c'est-à-dire à quelle profondeur dans la copier de travail la fusion doit aller. Les termes de profondeur utilisés sont décrits dans Section 4.3.1, « Profondeur d'extraction ». La profondeur par défaut est Copie de travail, qui utilise le paramètre existant de profondeur, et c'est presque toujours ce dont vous avez besoin.

La plupart du temps, vous voulez que la fusion prenne en compte l'historique du fichier, afin que les changements relatifs à un ancêtre commun soient fusionnés. Parfois vous pouvez avoir besoin de fusionner des fichiers qui sont apparentés, mais pas dans votre dépôt. Par exemple, vous pouvez avoir importé les versions 1 et 2 d'une bibliothèque tierce partie dans deux répertoires différents. Bien qu'ils soient apparentés logiquement, Subversion ne le sait pas parce qu'il ne voit que les archives que vous avez importées. Si vous essayiez de fusionner les différences entre ces deux arborescences, vous verriez une suppression complète suivi d'un ajout complet. Pour dire à Subversion de n'utiliser que les différences basées sur le chemin plutôt que les différences basées sur l'historique, cochez la case Ignorer l'ascendance. Vous pouvez en lire plus sur ce sujet dans le livre de Subversion, *Prise en compte ou non de l'ascendance* [http://svnbook.red-bean.com/fr/1.8/svn.branchmerge.advanced.html#svn.branchmerge.advanced.ancestry].

Vous pouvez spécifier la gestion des changements de caractères de fin de ligne et d'espacement. Ces options sont décrites dans Section 4.11.2, « Options de fins de ligne et d'espacement ». Le comportement par défaut est de considérer toutes les différences d'espaces et de fins de lignes comme des modifications à part entière, qu'il faut livrer.

La case à cocher Forcer la fusion est utilisée pour éviter un conflit d'arborescence lors de la suppression d'un fichier qui a été modifié localement ou qui n'est pas sous contrôle de version. Si le fichier est supprimé, il n'y a aucun moyen de le récupérer, ce qui explique que par défaut cette case ne soit pas cochée.

Si vous utilisez la fonction de suivi de fusion et que vous souhaitez marquer une révision comme ayant été fusionnée, sans vraiment avoir fait de fusion ici, cochez la case Enregistrer uniquement la fusion. Il y a deux raisons pour lesquelles vous pourriez être amenés à faire ceci. La fusion à effectuer est trop compliquée pour les algorithmes de fusion, vous faites donc la fusion à la main et marquez les modifications comme ayant été fusionnées de manière à ce que le suivi de fusion le sache. Ou vous voulez éviter qu'une révision particulière soit fusionnée. La marquer comme ayant été fusionnée empêchera les clients supportant le suivi de fusion de faire effectivement la fusion.

A présent que tout est correctement configuré, tout ce que vous avez à faire est de cliquer sur le bouton Fusionner. Si vous voulez avoir un aperçu du résultat, le bouton Tester la fusion simulera la fusion *sans* modifier la CdT. La liste des fichiers qui seront changés par une vraie fusion s'affichera, ainsi que les endroits où il *pourrait* y avoir des conflits. Comme le suivi de fusion rend le processus de fusion beaucoup plus complexe, il n'est pas possible de savoir à l'avance avec certitude si la fusion sera réalisée sans conflit, donc les fichiers marqués comme étant en conflit dans un test de fusion pourraient en fait fusionner sans aucun problème.

La boîte de dialogue de progression de la fusion vous montre chaque étape de la fusion, avec les plages de révisions concernées. Elle peut indiquer une révision de plus que ce à quoi vous vous attendiez. Par exemple, si vous avez demandé de fusionner la révision 123, la boîte de dialogue de progression affichera « Fusion des révisions 122 à 123 ». Pour le comprendre, il faut vous souvenir que la fusion est apparentée à la comparaison. Le processus de fusion génère une liste de différences entre deux points dans le dépôts, puis applique ces différences à votre copie de travail. La boîte de dialogue de progression vous montre simplement les points de départ et d'arrivée de la comparaison.

4.21.4. Vérifier les résultats de la fusion

La fusion est à présent effectuée. C'est une bonne idée d'aller regarder le résultat et de vérifier que tout est conforme à vos attentes. Fusionner est normalement assez compliqué. Il y a souvent des conflits si la branche s'est beaucoup éloignée du trunk.

Astuce

Quand des révisions sont fusionnées dans une copie de travail, TortoiseSVN génère un message de log à partir de toutes les révisions fusionnées. Ces messages sont alors disponibles depuis le bouton Messages récents dans la boîte de dialogue de livraison.

Pour personnaliser ce message généré, positionnez les propriétés de projet correspondantes sur votre copie de travail. Voir Section 4.18.3.10, « Fusionner les modèles de message de journal »

Pour les clients et serveurs Subversion antérieurs à la version 1.5, aucune information de fusion n'est conservée et il faut aller rechercher manuellement les révisions à fusionner. Une fois que vous avez testé les changements et que vous allez livrer cette révision, le commentaire de livraison devrait *toujours* inclure les numéros de révision inclus dans la fusion. Si vous voulez faire une autre fusion plus tard, vous aurez besoin de savoir ce que vous avez déjà fusionné pour ne pas appliquer vos modifications plusieurs fois. Pour plus d'informations, consultez *Best Practices for Merging* [http://svnbook.red-bean.com/en/1.4/svn.branchmerge.copychanges.html#svn.branchmerge.copychanges.bestprac] dans le livre de Subversion [NdT : version 1.4 non traduite en français].

Pour des clients et serveurs Subversion en version 1.5 ou postérieure, la fonctionnalité de suivi de fusion enregistrera les révisions fusionnées et empêchera de fusionner une même révision plusieurs fois. Cela vous simplifie considérablement la tâche, dans la mesure où vous pouvez simplement fusionner toute la plage de révisions à chaque fois en étant sûr que seules les nouvelles révisions seront vraiment fusionnées.

La gestion de branches est importante. Si vous voulez conserver une branche synchronisée avec le tronc, vous devrez vous assurer de fusionner souvent de sorte que la branche et le tronc ne soient pas trop éloignés. Bien sûr, vous devez toujours éviter de fusionner plusieurs fois les mêmes modifications comme expliqué ci-dessus.

Astuce

Si vous venez de fusionner avec le trunk une branche créée pour le développement d'une fonctionnalité, le trunk contient à présent le code de la nouvelle fontionnalité, et la branche est obsolète. Vous pouvez donc la supprimer du dépôt si besoin est.

Important

Subversion ne peut pas fusionner un fichier avec un dossier et vice versa — dossiers entre eux et fichiers entre eux uniquement. Si vous cliquez sur un fichier et ouvrez la boîte de dialogue de fusion, vous devez alors donner un chemin vers un fichier dans cette boîte de dialogue. Si vous choisissez un dossier et affichez la boîte de dialogue, vous devez alors spécifier une URL de dossier pour la fusion.

4.21.5. Suivi de fusion

La version 1.5 de Subversion a introduit des aides au suivi de fusion. Lorsque vous fusionnez des modifications depuis une arborescence vers une autre, les numéros de révision sont stockés et cette information peut être utilisée à plusieurs fins.

- Vous pouvez éviter le 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 tronc, la fenêtre de commentaires peut vous montrer les livraisons de la branche comme faisant partie des commentaires du tronc, donnant une meilleure traçabilité des modifications.
- Lorsque vous affichez la fenêtre de commentaires depuis la fenêtre de fusion, les révisions déjà fusionnées sont grisées.
- Quand l'annotation 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.

Les informations de suivi de fusion sont stockées dans la propriété `svn:mergeinfo` par le client lorsqu'il effectue la fusion. Au moment de la livraison de cette fusion, le serveur stocke l'information dans la base de données, et quand vous souhaitez fusionner, commenter ou annoter, le serveur peut répondre de manière appropriée. Afin que le système fonctionne correctement, vous devez vous assurer que le serveur, le dépôt et les

clients sont à jour. Les clients plus anciens ne stockeront pas la propriété `svn:mergeinfo` et les serveurs plus anciens ne pourront pas fournir l'information demandée par les clients récents.

Apprenez-en plus sur le suivi des fusions dans la *Documentation du suivi des fusions* [http://svn.apache.org/repos/asf/Subversion/trunk/notes/merge-tracking/index.html] de Subversion.

4.21.6. Gérer les conflits après une fusion

Important

Le texte des boîtes de dialogue de résolution de conflit est fourni par la bibliothèque SVN, et la traduction peut donc ne pas (encore) être synchronisée avec celle des boîtes de dialogue TortoiseSVN. Nous en sommes désolés.

Une fusion ne se passe pas toujours très bien. Il y a parfois un conflit. TortoiseSVN vous aide à suivre ce processus en vous montrant la boîte de dialogue de *conflit de fusion*.

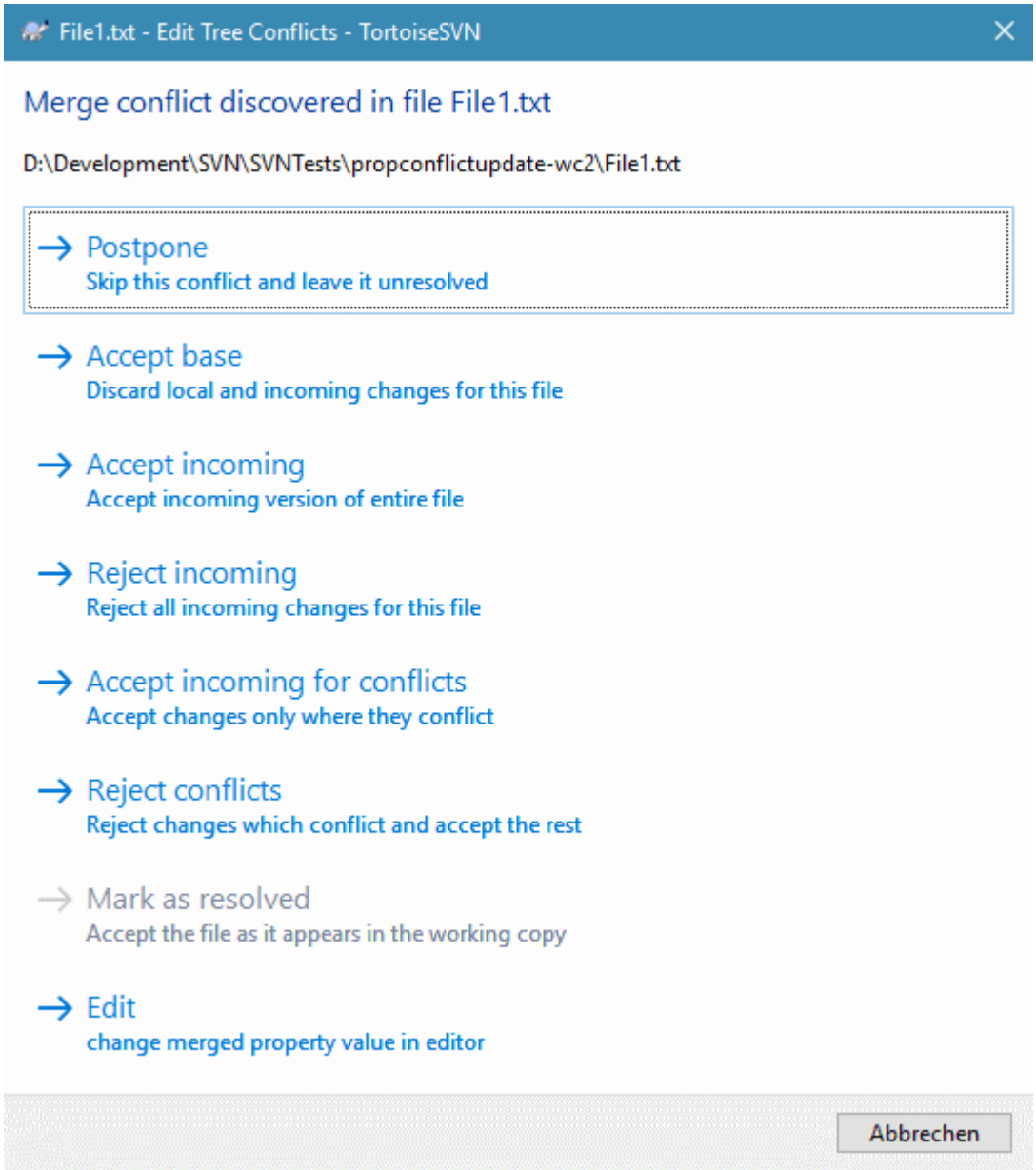


Figure 4.58. La boîte de dialogue de conflit de fusion

Il est probable que certains changements auront été fusionnés sans difficulté, mais que d'autres changements locaux seront en conflit avec les changements déjà livrés dans le dépôt. Tous les changements qui peuvent être fusionnés le sont. La boîte de dialogue de conflit de fusion vous donne plusieurs façons de gérer les lignes qui sont en conflit.

Pour les conflits normaux qui se produisent à cause de changements dans le contenu du fichier ou dans ses propriétés, la boîte de dialogue contient des boutons qui vous permettent de choisir ce que vous voulez garder ou rejeter parmi les parties en conflit.

S'il y a un conflit d'arborescence, veuillez d'abord consulter Section 4.6.3, « Conflits d'arborescence » à propos des différents types de conflits d'arborescence et de comment et pourquoi ils se produisent.

Pour résoudre les conflits d'arborescence après une fusion, une boîte de dialogue est affichée avec différentes options sur comment résoudre le conflit :

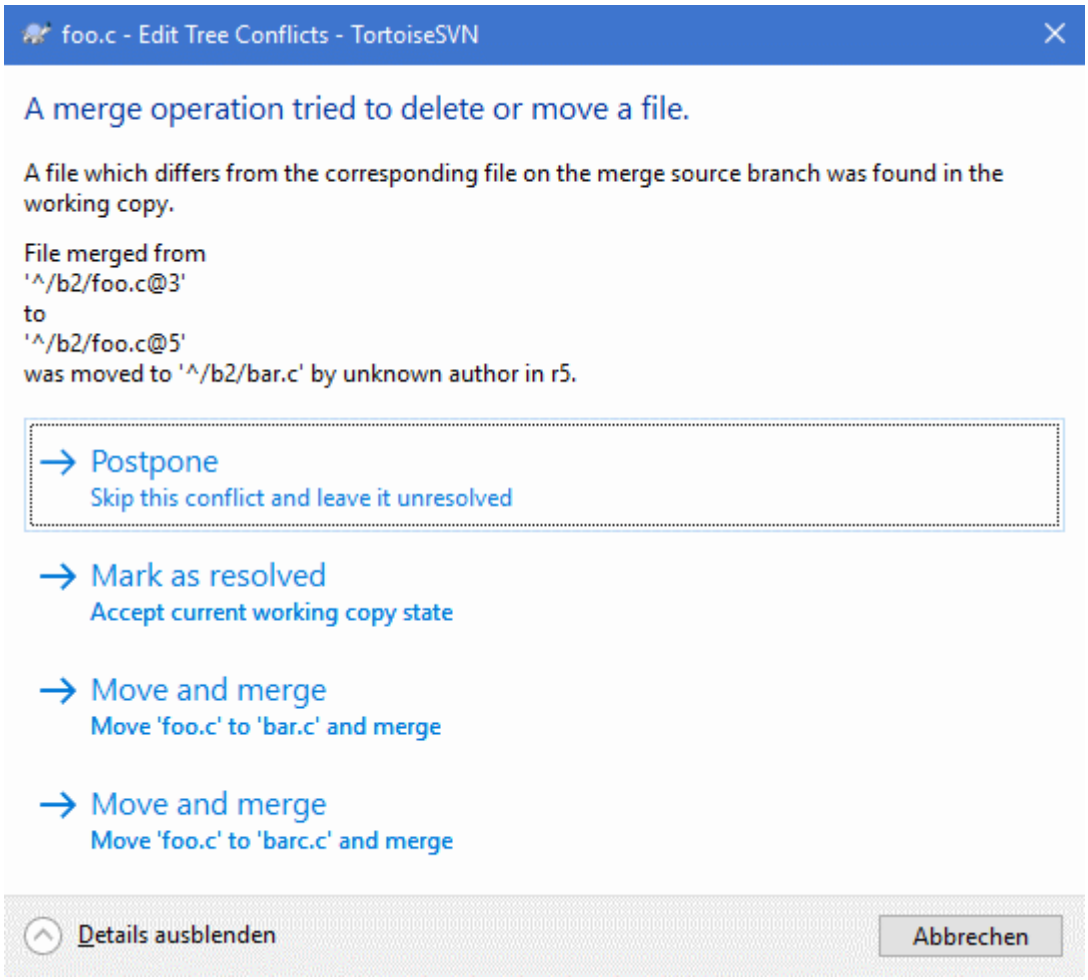


Figure 4.59. La boîte de dialogue de fusion de conflit d'arborescence

Comme il y a plusieurs situations possibles de conflit d'arborescence, les boutons affichés dans la boîte de dialogue pour résoudre ces situations dépendront du conflit concerné. Les textes et libellés des boutons expliquent ce que fait l'option pour résoudre le conflit. Si vous n'êtes pas sûr de vous, vous pouvez soit annuler la boîte de dialogue, soit utiliser le bouton Report pour résoudre le conflit plus tard.

4.21.7. Branche de maintenance d'une fonctionnalité

Quand vous développez une nouvelle fonctionnalité dans une branche séparée, c'est une bonne pratique de prévoir une politique de réintégration de cette branche lorsque la nouvelle fonctionnalité sera prête. Si la branche principale

— `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 adopter 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 plage de révisions, la plage de révisions couvrant la période de développement de la branche.

Dans le cas où la fonctionnalité prend plus de temps et que vous avez besoin de prendre en compte les évolutions du `trunk`, alors vous devez garder votre branche synchronisée. Cela signifie simplement que vous allez fusionner de temps en temps les évolutions du trunk dans votre branche, de manière à ce que votre branche contienne toutes les modifications du trunk *en plus* de la nouvelle fonctionnalité. Le processus de synchronisation utilise la commande Fusionner une plage de révisions. Quand le développement de la fonctionnalité est achevé vous pouvez alors fusionner la branche dans le `trunk` en utilisant soit Réintégrer une branche soit Fusionner deux arborescences.

Une autre façon (rapide) de fusionner tous les changements du trunk dans la branche de la fonctionnalité est d'utiliser la commande TortoiseSVN → Fusionner tout... dans le menu contextuel étendu (gardez appuyée la touche **Maj** en faisant un clic droit sur le fichier).

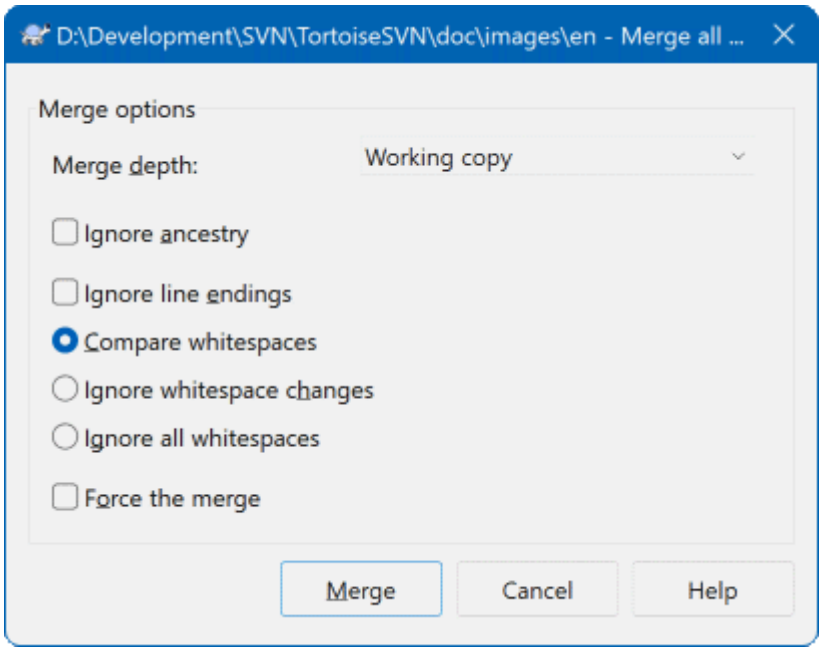


Figure 4.60. La boîte de dialogue Fusionner tout

Cette boîte de dialogue est très simple. Tout ce que vous avez à faire est de positionner les options pour la fusion, comme décrit dans Section 4.21.3, « Options de fusion ». TortoiseSVN s'occupe automatiquement du reste avec le suivi de fusion.

4.22. Verrouiller

Subversion fonctionne généralement mieux sans verrouillage, en utilisant les méthodes « Copier-Modifier-Fusionner » décrites précédemment dans Section 2.2.3, « La solution Copier-Modifier-Fusionner ». Cependant il y a quelques cas où vous pouvez devoir appliquer 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'une d'entre elles perdra ses modifications.
- Votre société a toujours utilisé un système de contrôle de version avec des verrous par le passé et le management a décidé 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 antérieures ne supportent pas du tout le verrouillage. Si vous utilisez un accès de type `file://`, alors bien sûr seul votre client doit être mis à jour.

Les trois significations de « verrou »

Dans cette section, et presque partout dans ce livre, les mots « verrou » et « verrouillage » décrivent un mécanisme d'exclusion mutuelle entre utilisateur afin d'éviter des collisions de livraison. Malheureusement, il y a deux autres types de « verrou » auxquels Subversion, et donc ce livre, doivent parfois s'intéresser.

Le deuxième type est les `verrous de copie de travail`, utilisés en interne par Subversion pour empêcher les collisions entre plusieurs clients Subversion opérant sur la même copie de travail. Ces verrous apparaissent généralement quand une commande de mise à jour, de livraison, etc., est interrompue à la suite d'une erreur. Ces verrous peuvent être retirés en lançant la commande Nettoyer sur la copie de travail, comme décrit dans Section 4.17, « Nettoyer ».

Et troisièmement, les fichiers et les dossiers peuvent être verrouillés s'ils sont en cours d'utilisation par un autre processus. Par exemple, si vous avez un document word ouvert dans Word, ce fichier est verrouillé et TortoiseSVN ne peut pas y accéder.

Vous pouvez généralement oublier ces autres types de verrou jusqu'à ce que quelque chose se passe mal et vous oblige à vous y intéresser. Dans ce livre, « verrou » signifie le premier type, à moins que le contraire ne soit clair d'après le contexte ou précisé explicitement.

4.22.1. Comment le verrouillage fonctionne dans Subversion

Par défaut, rien n'est verrouillé et quiconque a un accès en livraison peut livrer des modifications de n'importe quel fichier à tout moment. Les autres mettront à jour leurs copies de travail périodiquement et les modifications du dépôt seront fusionnées avec les modifications locales.

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 retiriez le verrou. Un fichier verrouillé ne peut être modifié d'aucune façon dans le dépôt, il ne peut donc pas non plus être supprimé ou renommé, sauf par le propriétaire du verrou.

Important

Un verrou n'est pas assigné à un utilisateur spécifique, mais à un utilisateur spécifique et une copie de travail. Avoir un verrou sur une copie de travail empêche également le même utilisateur d'effectuer des livraisons des fichiers verrouillés depuis d'autres copies de travail.

Par exemple, imaginez que l'utilisateur Jon a une copie de travail sur le PC de son bureau. Il commence à travailler sur une image, et acquiert donc un verrou sur ce fichier. Quand il quitte le bureau, il n'en a pas encore terminé avec ce fichier, donc il ne libère pas ce verrou. De retour chez lui, Jon a aussi une copie de travail et décide de continuer à travailler un peu sur le projet. Mais il ne peut pas modifier ou livrer ce même fichier image, parce que le verrou sur ce fichier réside sur sa copie de travail au bureau.

Toutefois, les autres utilisateurs ne sauront pas nécessairement que vous avez posé un verrou. À moins qu'ils ne vérifient le verrouillage régulièrement, ils ne le découvriront que 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* cette copie de travail ne détienne un verrou pour le fichier. Cela agit comme un avertissement de ne pas éditer ce fichier à moins que vous n'ayez d'abord posé un verrou. Les fichiers qui sont versionnés et en lecture seule sont identifiés par une coloration spéciale dans TortoiseSVN, pour indiquer que vous devez poser un verrou avant l'édition.

Les verrous sont enregistrés par emplacement de copie de travail et par propriétaire. Si vous avez plusieurs copies de travail (à la maison, au travail) alors vous ne pouvez détenir un verrou que dans *une* de ces copies de travail.

Si l'un de vos collègues pose un verrou et part ensuite en vacances sans le retirer, que faites-vous ? Subversion fournit un moyen de forcer les verrous. Retirer un verrou détenu par quelqu'un d'autre s'appelle *casser* le verrou et prendre de force 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 en bons termes avec vos collègues.

Les verrous sont enregistrés dans le dépôt 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 dépôt est toujours la référence absolue.

4.22.2. Obtenir un verrou

Sélectionnez les fichiers de votre copie de travail pour lesquels vous voulez un verrou, puis sélectionnez la commande TortoiseSVN → Obtenir un verrou....

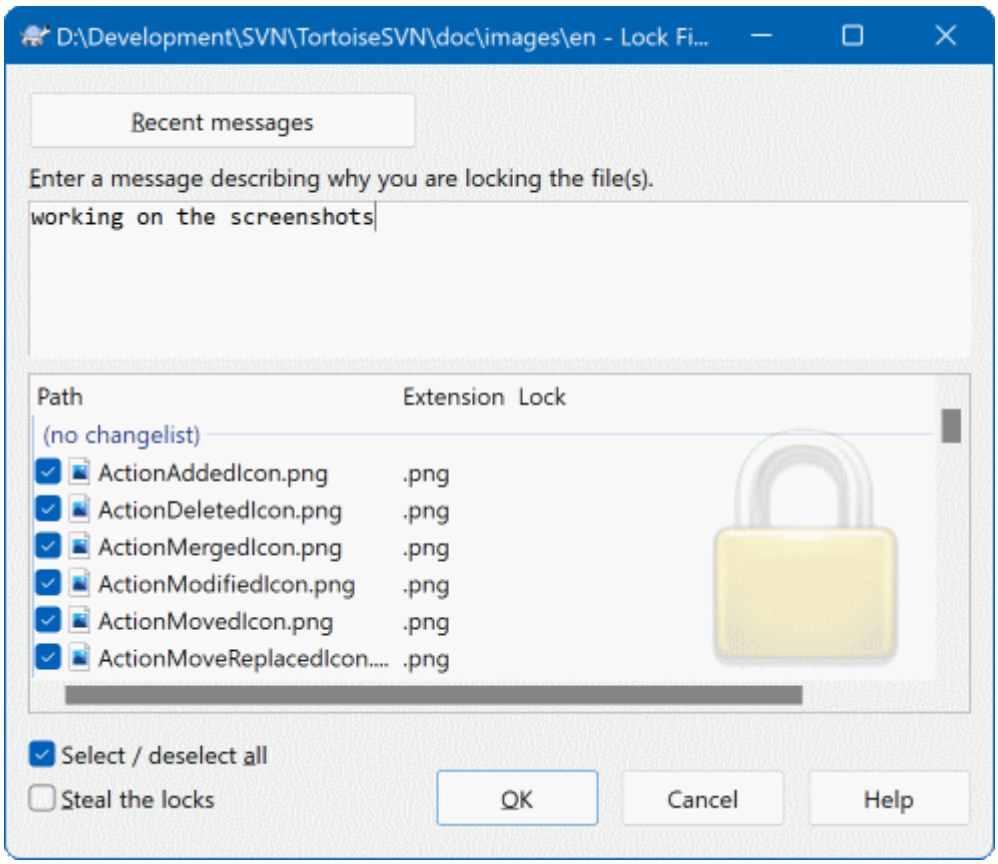


Figure 4.61. La boîte de dialogue Verrouiller

Une boîte de dialogue apparaît, vous permettant de saisir un commentaire, pour que les autres puissent voir pourquoi vous avez verrouillé le fichier. Le commentaire est facultatif et actuellement utilisé avec les seuls dépôts basés sur Svnserve. Si (et *seulement* si) vous devez voler le verrou de quelqu'un d'autre, cochez la case Voler les verrous, puis cliquez sur OK.

Vous pouvez positionner la propriété de projet `tsvn:logtemplatelock` pour fournir un modèle de message de verrouillage à compléter par les utilisateurs. Voyez Section 4.18, « Configuration des projets » pour des instructions sur comment positionner les propriétés.

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 sélectionnés pour être verrouillés. C'est la bonne

manière de faire si vous voulez vraiment verrouiller une arborescence complète, cependant vous pourriez devenir très impopulaire auprès de vos collègues si vous les empêchez d'accéder au projet. À utiliser avec parcimonie...

4.22.3. Retirer un verrou

Pour vous assurer de ne pas oublier de retirer 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 retirer 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 de livrer vos changements.

Pour retirer un verrou manuellement, sélectionnez les fichiers de votre copie de travail dont vous voulez retirer les verrous, choisissez ensuite la commande TortoiseSVN → Retirer un verrou Il n'y a rien d'autre préciser, et TortoiseSVN va communiquer avec le dépôt et retirer les verrous. Vous pouvez aussi utiliser cette commande sur un dossier pour retirer tous les verrous récursivement.

4.22.4. Vérifier le statut des verrous

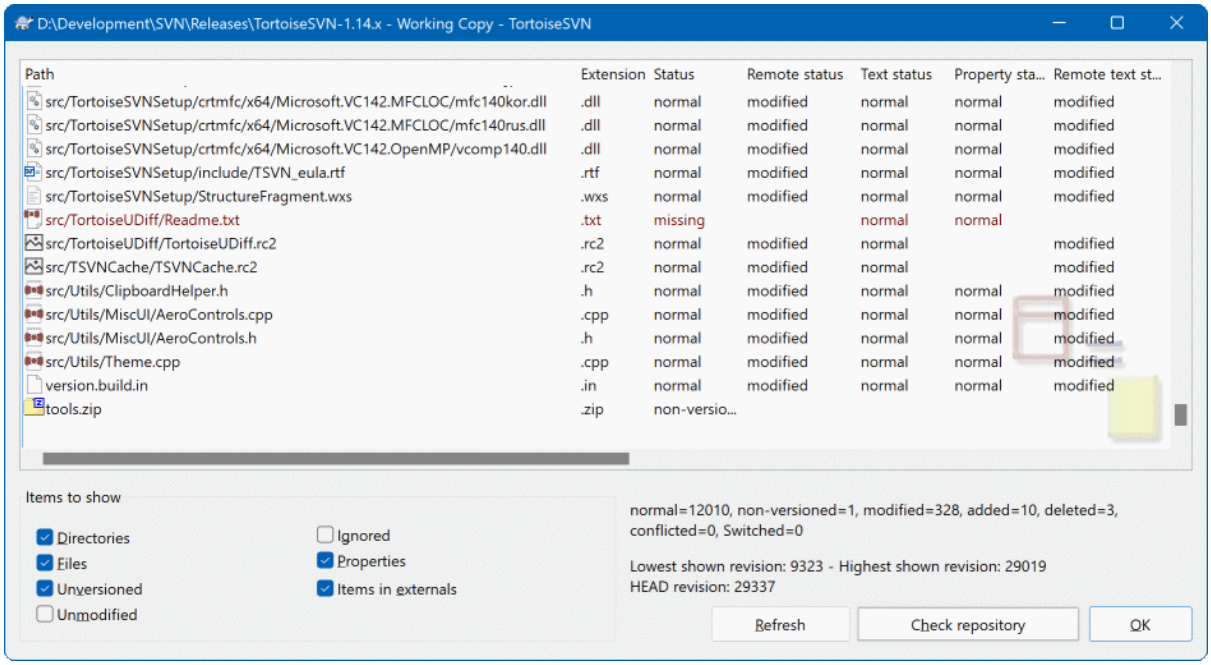


Figure 4.62. La boîte de dialogue Vérifier les modifications

Pour voir quels verrous les autres et vous détenez, vous pouvez utiliser TortoiseSVN → Vérifier les modifications.... Les jetons de verrouillage 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 dépôt.

À partir du menu contextuel accessible ici, vous pouvez aussi bien obtenir et retirer 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 de 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 retirez le verrou, il est libre de livrer ses modifications et d'écraser les vôtres. Subversion ne perd pas de données, mais vous avez perdu la protection de travail d'équipe que le verrouillage vous avait donnée.

4.22.5. 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 4.18, « Configuration des projets » pour les instructions sur le mode d'activation des propriétés. Les fichiers avec cette propriété activée seront toujours extraits et mis à jour avec l'attribut Lecture seule activé à moins que votre copie de travaill ne détienne un verrou.

Pour vous le rappeler, TortoiseSVN utilise une coloration spéciale pour l'indiquer.

Si vous appliquez une politique où tout fichier doit être verrouillé, vous pouvez trouver alors plus facile d'utiliser la fonctionnalité auto-props de Subversion pour activer la propriété automatiquement à chaque fois vous ajoutez de nouveaux fichiers. Lisez Section 4.31, « Configuration de TortoiseSVN » pour plus d'informations.

4.22.6. Les scripts hook de verrouillage

Quand vous créez un nouveau dépôt avec Subversion 1.2 ou supérieur, quatre modèles de hooks sont créés dans le répertoire `hooks` du dépôt. Ceux-ci sont appelés avant et après l'obtention d'un verrou et avant et après le retrait 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 d'installé, tous vos utilisateurs peuvent être informés si quelqu'un verrouille/déverrouille un fichier. Vous pouvez trouver un script hook d'exemple `hooks/post-lock.tmpl` dans le dossier de votre dépôt.

Vous pourriez aussi utiliser des hooks pour ne pas permettre la casse ou le vol de verrous, ou peut-être les limiter à un administrateur nommé. Ou peut-être vous voulez envoyer un email au propriétaire quand l'un de ses verrous est cassé ou volé.

Lisez Section 3.3, « Scripts hook côté serveur » pour en savoir plus.

4.23. Créer et appliquer des patches

Pour les projets open-source (comme celui-ci), tout le monde a accès au dépôt 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 dépôt 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.

4.23.1. 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...

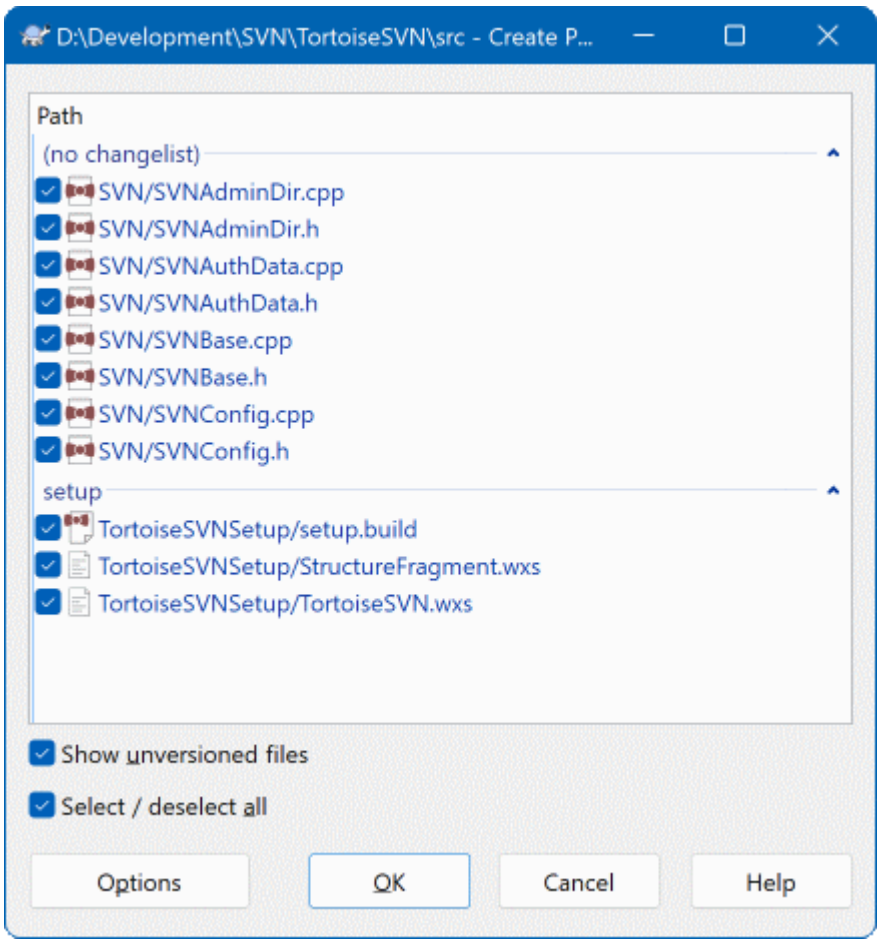


Figure 4.63. La boîte de dialogue Créer un patch

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 dépôt.

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.

En cliquant sur le bouton Options, vous pouvez spécifier comment le patch est créé. Par exemple, vous pouvez spécifier que les changements de caractères de fin de ligne ou d'espacement ne sont pas inclus dans le fichier de patch final.

Vous pouvez produire des patches séparés contenant des changements sur des jeux différents de fichiers. Bien sûr, si vous créez un fichier patch, faites d'autres changements dans les *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.

Astuce

Ne sauvegardez pas le fichier patch avec une extension `.txt` si vous avez l'intention de l'envoyer par e-mail à quelqu'un d'autre. Les fichiers texte sont souvent abîmés par les logiciels de messagerie et les caractères d'espacement et de fin de ligne sont souvent convertis et compressés automatiquement. Si cela se produit, il y aura des problèmes pour appliquer le patch. Donc utilisez `.patch` ou `.diff` comme extension quand vous sauvegardez le fichier de patch.

Vous pouvez aussi sauvegarder le patch dans le presse-papier plutôt que dans un fichier. Cela permet, par exemple, de le coller dans un mail afin de le transmettre à d'autres personnes pour une revue. Ou alors, si vous avez deux copies de travail sur une machine et que vous vouliez reporter des changements de l'une à l'autre, copier le patch dans le presse-papier est un moyen commode pour le faire.

Si vous préférez, vous pouvez créer un fichier de patch depuis les boîtes de dialogue Livrer ou Vérifier les modifications. Sélectionnez les fichiers et utilisez l'élément du menu contextuel pour créer un patch à partir de ces fichiers. Si vous voulez voir la boîte de dialogue Options, vous devez garder la touche **Maj** appuyée en faisant un clic droit.

4.23.2. Appliquer un patch

Les fichiers patch sont appliqués sur votre copie de travail. Cela doit être fait au même niveau de dossier qui a été utilisé pour créer le patch. Si vous n'êtes pas sûr duquel il s'agit, regardez simplement la première ligne du fichier de patch. Par exemple, si le premier fichier sur lequel vous avez travaillé était `doc/source/francais/chapitre1.xml` et que la première ligne dans le fichier de patch est `Index: francais/chapitre1.xml` alors vous devez appliquer le patch sur le dossier `doc/source/`. Cependant, du moment que vous êtes dans la bonne copie de travail, si vous choisissez le mauvais niveau de dossier, TSVN le remarquera et suggérera le niveau correct.

Pour appliquer un patch à votre copie de travail, vous devez avoir au moins l'accès en lecture pour le dépôt. 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.

Depuis le menu contextuel de ce dossier, cliquez sur TortoiseSVN → Appliquer le patch... Cela affichera une boîte de dialogue Ouvrir un fichier vous permettant de choisir le fichier patch à appliquer. Par défaut, seuls les fichiers `.patch` ou `.diff` sont affichés, mais vous pouvez opter pour "Tous les fichiers ". Si vous avez précédemment sauvegardé le patch dans le presse-papier, vous pouvez utiliser le bouton Ouvrir depuis le presse-papier dans le dialogue d'ouverture de fichiers. Notez que cette option est visible dans le cas où vous avez sauvegardé votre patch dans le presse-papier via le menu TortoiseSVN → Créer un patch.... La copie d'un patch du presse-papier vers une autre application ne fera pas appaître ce bouton.

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 invité 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 dépôt.

4.24. Qui a changé quelle ligne ?

Parfois, vous avez besoin de savoir non seulement quelles lignes ont changé, mais aussi qui exactement a changé des lignes spécifiques dans un fichier. C'est à ce moment que la commande TortoiseSVN → Annoter..., parfois aussi mentionnée sous son nom anglais de *blame*, devient pratique.

Cette commande liste, pour chaque ligne d'un fichier, l'auteur et la révision où la ligne a été changée.

4.24.1. Annoter pour les fichiers

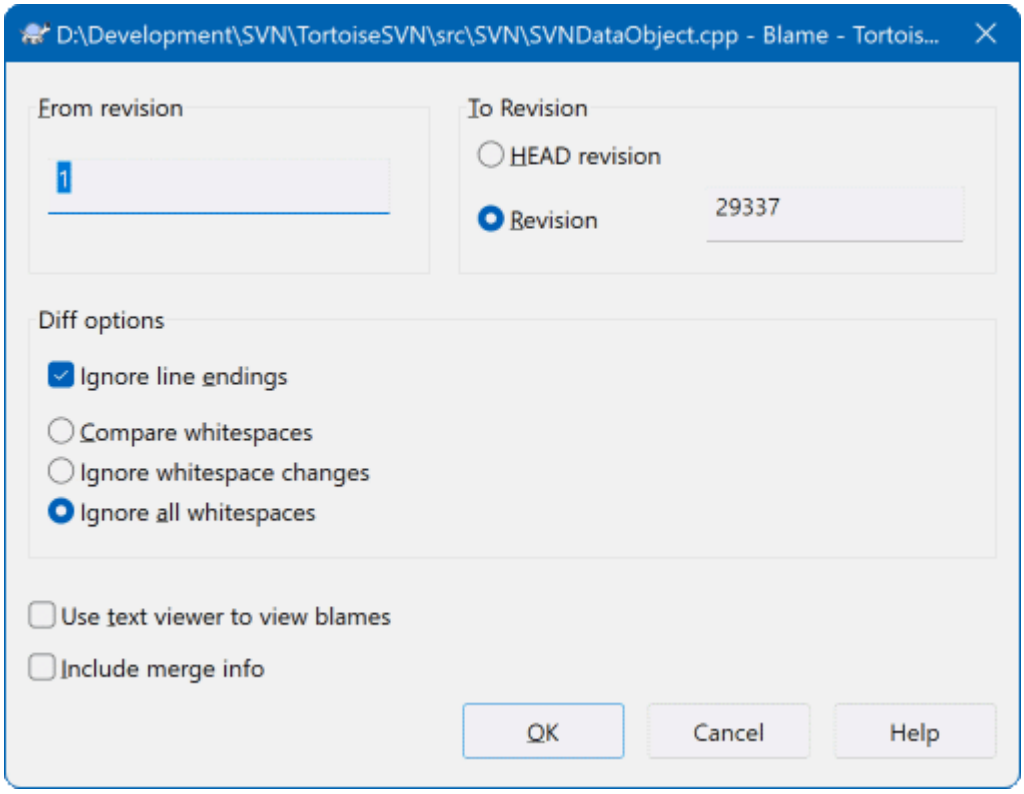


Figure 4.64. La boîte de dialogue Annoter

Si vous n'êtes pas intéressé par les changements des révisions précédentes, vous pouvez définir la révision d'où l'annotation devrait commencer. Mettez ce champ à 1 si vous voulez l'annotation pour *toutes* les révisions.

Par défaut, le fichier d'annotation est visualisé avec *TortoiseBlame*, qui surligne les différentes révisions pour le rendre plus facile à lire. Si vous voulez imprimer ou éditer le fichier d'annotation, cochez Utiliser la visionneuse texte pour voir les annotations.

Vous pouvez spécifier la manière de gérer les changements de caractères de fin de ligne et d'espacement. Ces options sont décrites dans Section 4.11.2, « Options de fins de ligne et d'espacement ». Par défaut, toutes les modifications de caractères de fin de ligne et d'espacement sont considérés comme des changements à part entière, mais si vous souhaitez ignorer les changements d'indentation et trouver l'auteur original, vous pouvez sélectionner l'option appropriée ici.

Vous pouvez également inclure les informations de fusion si vous le souhaitez, bien que cette option puisse prendre nettement plus longtemps à récupérer depuis le serveur. Quand les lignes sont fusionnées depuis une autre source, l'information d'annotation montre la révision à laquelle le changement a été fait dans la source originale ainsi que la révision à laquelle il a été fusionné dans ce fichier.

Une fois que vous appuyez sur OK, TortoiseSVN commence à récupérer les données pour créer le fichier d'annotation. Une fois que le processus d'annotation est terminé, le résultat est écrit dans un fichier temporaire et vous pouvez visualiser les résultats.

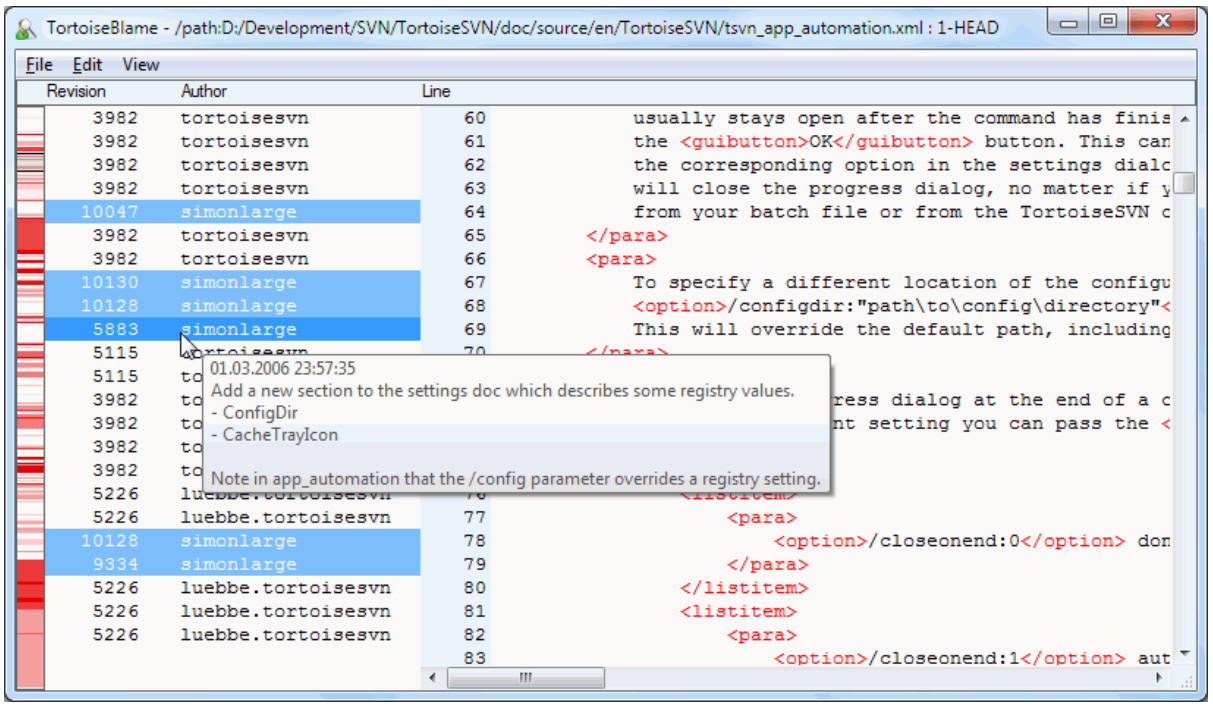


Figure 4.65. TortoiseBlame

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 devez utiliser la boîte de dialogue du journal pour faire une recherche dedans.

Vous pouvez aussi aller à une ligne particulière en utilisant Editer → Aller à la ligne....

Lorsque la souris survole la colonne annotation, un menu contextuel s'affiche vous permettant de comparer les révision et de consulter l'historique, en prenant comme référence le numéro de ligne sous le curseur de la souris. Menu contextuel → Annoter la révision précédente crée une annotation sur ce même fichier, mais en utilisant la révision précédente comme limite supérieure. Cela vous donne le rapport d'annotation du fichier dans l'état où il était juste avant que la ligne que vous survolez soit modifiée pour la dernière fois. Menu contextuel → Montrer les modifications démarre votre visionneuse de différences, vous montrant ce qui a changé à la révision référencée. Menu contextuel → Montrer les commentaires affiche la boîte de dialogue du journal des révisions, en commençant par la révision référencée.

Si vous souhaitez un meilleur indicateur visuel de l'emplacement des modifications les plus récentes et les plus anciennes, sélectionnez Voir → Colorer l'âge des lignes. Cette option utilisera un gradient de couleur pour

afficher les lignes les plus récentes en rouge et les plus anciennes en bleu. La coloration par défaut est assez légère, mais vous pouvez la modifier en utilisant les préférences de TortoiseBlame.

Si vous utilisez le suivi de fusion et que vous avez demandé les informations de fusion en lançant l'annotation, les lignes fusionnées sont affichées d'une façon légèrement différente. Quand une ligne a changé à la suite d'une fusion depuis un autre chemin, TortoiseBlame affichera la révision et l'auteur du dernier changement dans le fichier original plutôt que la révision où la fusion a eu lieu. Ces lignes sont indiquées en affichant la révision et l'auteur en italique. La révision où la fusion a eu lieu est affichée séparément dans l'infobulle quand la souris survole les colonnes d'information d'annotation. Si vous ne voulez pas que les lignes fusionnées soient affichées de cette façon, décochez la case Inclure les informations de fusion au moment de commencer l'annotation.

Si vous voulez voir les chemins concernés par la fusion, sélectionnez Voir → Fusionner les chemins. Cela affiche le chemin où la ligne a été modifiée pour la dernière fois, à l'exclusion des changements qui résultent d'une fusion.

La révision affichée dans l'information d'annotation représente la dernière révision où le contenu de cette ligne a changé. Si le fichier a été créé en copiant un autre fichier, alors, jusqu'à ce que vous changiez une ligne, sa révision d'annotation affichera le dernier changement dans le fichier source original, pas la révision où la copie a été effectuée. Cela s'applique également aux chemins affichés avec les informations de fusion. Le chemin affiche l'emplacement du dépôt où le dernier changement a été fait sur cette ligne.

On peut accéder aux paramètres de TortoiseBlame en sélectionnant TortoiseSVN → Paramètres... dans l'onglet TortoiseBlame. Voir Section 4.31.9, « Configuration de TortoiseBlame ».

4.24.2. Annoter les différences

Une des limitations du rapport d'annotation est qu'il ne montre que le fichier tel qu'il était dans une révision donnée, et la dernière personne à changer chaque ligne. Parfois vous voulez savoir ce qui a été changé, ainsi que qui l'a changé. Si vous faites un clic droit sur une ligne dans TortoiseBlame, vous avez un élément de menu contextuel pour afficher les changements effectués dans cette révision. Mais si voulez voir les changements *et* les informations d'annotation en même temps, alors vous avez besoin d'une combinaison des rapports de comparaison et d'annotation.

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 récupèrera les données d'annotation pour les 2 révisions, puis utilisera le visualiseur de différences pour comparer les deux fichiers d'annotation.

Annoter les modifications
: Sélectionnez une révision dans le panneau supérieur, puis choisissez un fichier dans le panneau inférieur et sélectionnez Menu contextuel → Annoter les modifications. Cela récupèrera les données d'annotation pour la révision sélectionnée et la révision précédente, puis utilisera le visualiseur de différences pour comparer les deux fichiers d'annotations.

Comparer avec la BASE de travail et annoter
: Affichez le journal pour un seul fichier et, dans le panneau supérieur, choisissez une seule révision, puis sélectionnez Menu contextuel → Comparer avec la BASE de travail et annoter. Cela récupèrera les données d'annotation pour la révision choisie et pour le fichier dans la BASE de travail, puis utilisera le visualiseur de différences pour comparer les deux fichiers d'annotations.

4.25. L'explorateur de dépôt

Parfois, vous devez travailler directement sur le dépôt, sans avoir de copie de travail. C'est à cela que sert l'*explorateur de dépôt*. Comme l'explorateur et les recouvrements d'icône vous permettent de voir votre copie de travail, l'explorateur de dépôt vous permet de voir la structure et le statut du dépôt.

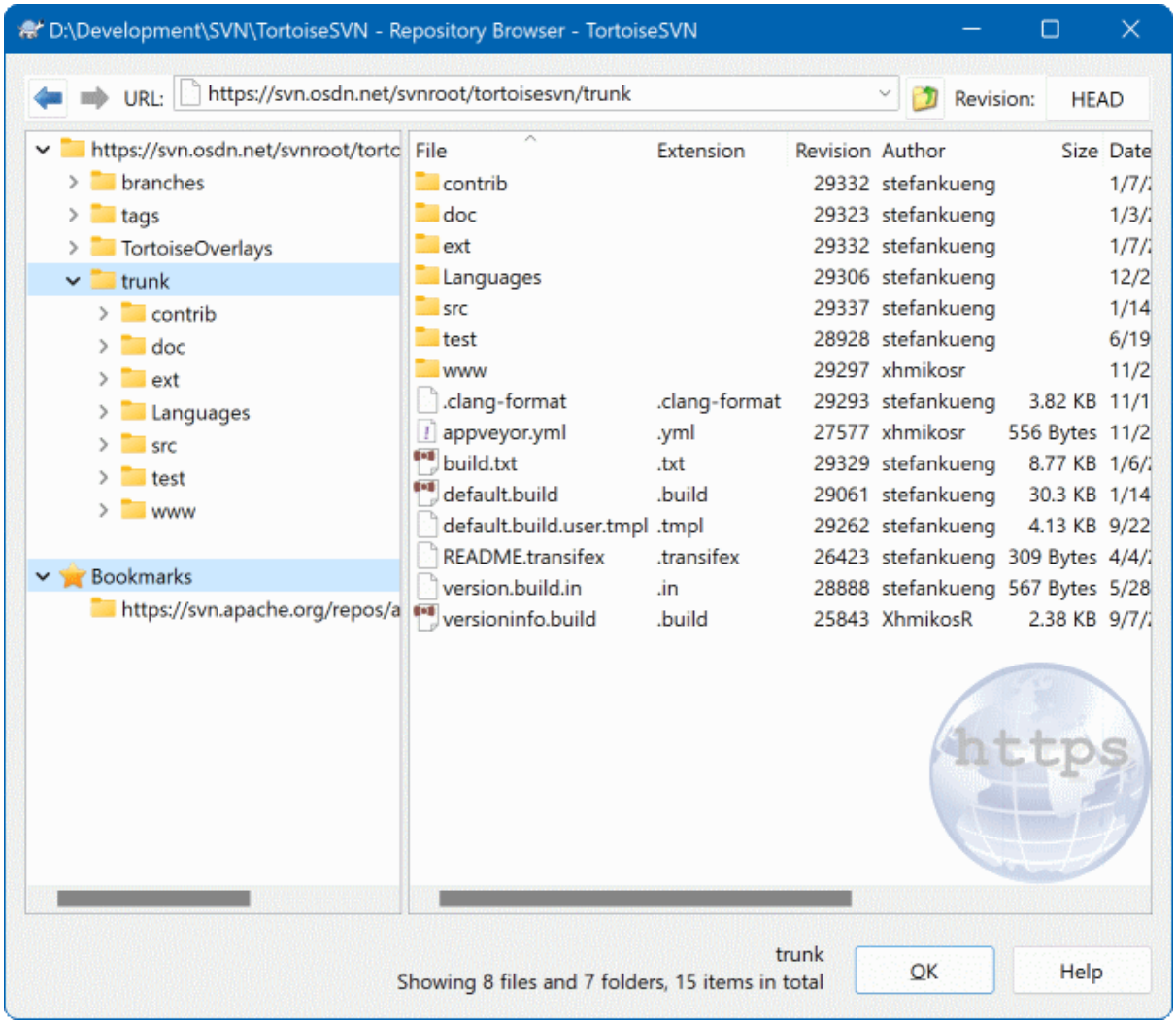


Figure 4.66. L'explorateur de dépôt

Avec l'explorateur de dépôt vous pouvez exécuter des commandes comme copier, déplacer, renommer... directement sur le dépôt.

L'explorateur de dépôt est très similaire à l'explorateur Windows, excepté le fait qu'il affiche le contenu du dépôt à une révision donnée au lieu de fichiers sur votre ordinateur. Dans le panneau de gauche vous pouvez voir l'arborescence des répertoires, et dans le panneau de droite le contenu du répertoire sélectionné. En haut de la fenêtre de l'explorateur de dépôt vous pouvez entrer l'URL du dépôt et la révision que vous souhaiter afficher.

Les dossiers inclus avec la propriété `svn:externals` sont également affichés dans l'explorateur de dépôt. Ces dossiers sont affichés avec une petite flèche sur l'icône pour indiquer qu'ils ne font pas partie de la structure du dépôt, mais que ce sont juste des liens.

Tout comme dans l'explorateur Windows, vous pouvez cliquer sur les en-têtes des colonnes dans le panneau de droite si vous voulez modifier le tri. Et comme dans l'explorateur il y a des menus contextuels dans 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.
- Éditer le fichier sélectionné. Cela extraira une copie de travail temporaire et lancera l'éditeur par défaut pour ce type de fichier. Quand vous fermez l'éditeur, si les changements ont été sauvegardés, alors une boîte de dialogue de livraison apparaît pour vous permettre de saisir un commentaire et de livrer le changement.

- Afficher les commentaires de révision de ce fichier, ou afficher le graphique de toutes les révisions de manière à voir d'où vient le fichier.
- Annoter le fichier, pour voir qui a changé quelle ligne et quand.
- Extraire un seul fichier. Cela crée une copie de travail « partielle » qui ne contient que ce seul fichier.
- Supprimer ou renommer le fichier.
- Récupérer une copie non versionnée du fichier sur votre disque dur.
- Copie l'URL affichée dans la barre d'adresse vers le presse-papiers.
- Faire une copie du fichier, soit à un autre endroit dans le dépôt, soit dans une copie de travail basée sur le même dépôt.
- Voir/éditer les propriétés du fichier.
- Créer un raccourci pour que vous puissiez rapidement redémarrer l'explorateur de dépôt, ouvert directement à cet emplacement.

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.
- Créer un nouveau répertoire dans le dépôt.
- Ajouter directement au dépôt des fichiers ou dossiers non versionnés. C'est effectivement l'opération d'importation de Subversion.
- Supprimer ou renommer le répertoire.
- Faire une copie du répertoire, dans un autre endroit du dépôt ou dans une copie de travail hébergée dans le même dépôt. Cette option peut aussi être utilisée pour créer une branche ou un tag sans pour autant avoir à récupérer une copie de travail.
- Voir/Modifier les propriétés du dossier.
- Marquer le dossier pour la comparaison. Le nom d'un dossier marqué est en gras.
- Comparer le dossier avec un dossier précédemment marqué, soit sous forme de fichier de différences unifiées, soit comme une liste de fichiers modifiés qui peuvent ensuite être comparés visuellement en utilisant l'outil de comparaison par défaut. Cela peut être particulièrement utile pour comparer deux tags, ou le trunk et une branche pour voir ce qui a changé.

Si vous choisissez deux éléments dans le panneau de droite, vous pouvez voir les différences soit sous forme de fichier de différences unifiées, soit comme une liste des fichiers qui peuvent être comparés visuellement en utilisant l'outil de comparaison par défaut.

Si vous sélectionnez plusieurs répertoires dans le panneau de droite, vous pouvez les extraire tous simultanément dans un dossier parent commun.

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.

Les éléments externes (référencés avec `svn:externals`) sont également affichés dans l'explorateur de dépôt, et vous pouvez même parcourir le contenu du dossier. Les éléments externes sont marqués avec une flèche rouge sur l'icône de l'élément.

Vous pouvez utiliser **F5** pour actualiser l'affichage. Cela mettra à jour tout ce qui est affiché. Si vous souhaitez récupérer ou actualiser les informations pour les nœuds qui n'ont pas encore été ouverts, utilisez **Ctrl-F5**. Par la suite, ouvrir n'importe quel nœud se fera instantanément, sans délai dû au réseau, lors de l'affichage de l'information.

Vous pouvez aussi utiliser l'explorateur de dépôt pour les opérations de glisser-déplacer. Si vous glissez un dossier de l'explorateur dans l'explorateur de dépôt, il sera importé dans le dépôt. Notez que si vous glissez plusieurs éléments, ils seront importés dans des livraisons séparées.

Si vous voulez déplacer un élément au sein du dépôt, 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.

Si vous voulez copier/déplacer un fichier ou un dossier vers un autre emplacement et lui donner en même temps un nouveau nom, vous pouvez le glisser avec le bouton droit ou le glisser avec le bouton droit en maintenant la touche **Ctrl** appuyée au lieu de le glisser avec le bouton gauche. Dans ce cas, une boîte de dialogue de renommage s'affiche, où vous pouvez saisir un nouveau nom pour le fichier ou le dossier.

Quand vous faites des changements dans le dépôt 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 opportunité d'annuler l'action.

Parfois quand vous essaierez d'ouvrir un chemin, vous obtiendrez un message d'erreur à la place des détails de l'élément. Cela pourrait arriver si vous spécifiez une URL invalide, ou si vous n'avez pas d'autorisation d'accès, ou s'il y a d'autres problèmes serveur. Si vous devez copier ce message pour l'inclure dans un e-mail, faites simplement un clic droit dessus et utilisez Menu contextuel → Copier le message d'erreur dans le presse-papiers, ou utilisez simplement **Ctrl+C**.

Les URLs/dépôts favoris sont affichés sous les dossier du dépôt courant dans l'arborescence du panneau de gauche. Vous pouvez y ajouter des entrées en faisant un clic droit sur un fichier ou dossier et en sélectionnant Menu contextuel → Ajouter aux favoris. Cliquer sur un favori naviguera vers ce dépôt et fichier/dossier.

4.26. Graphiques de révision

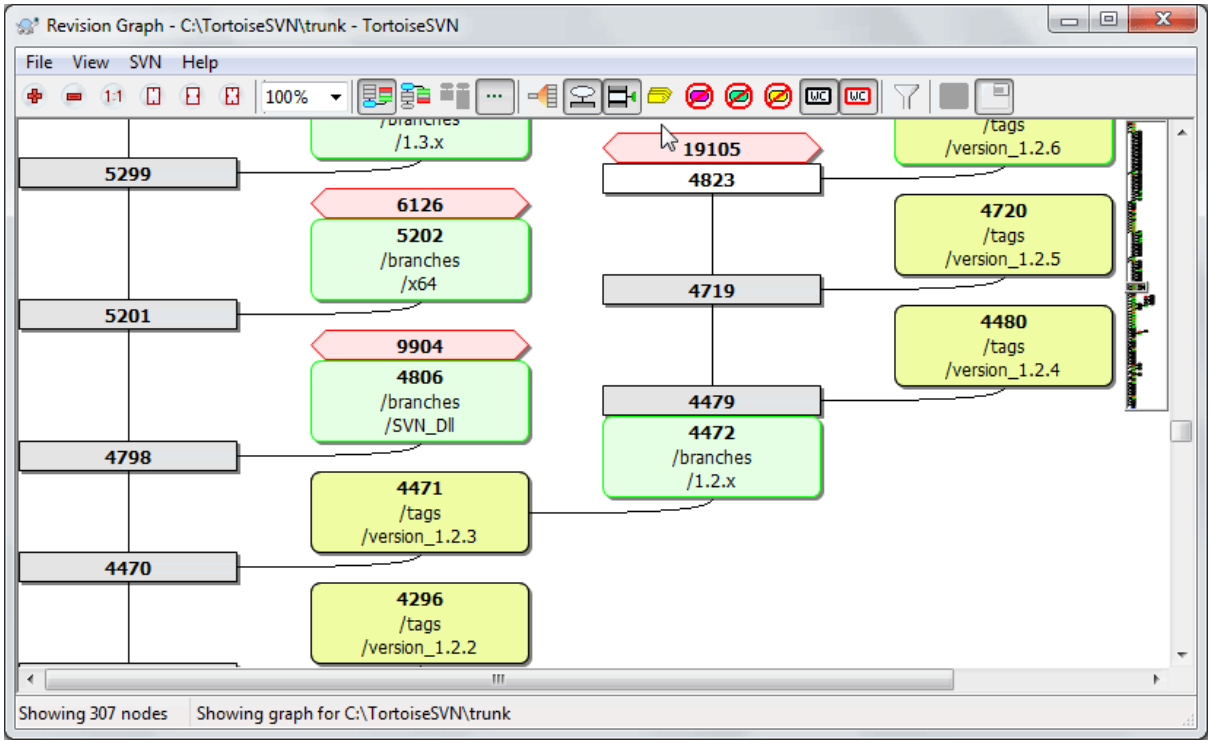


Figure 4.67. Un graphique de révision

Parfois vous avez besoin de savoir à quel moment les branches et les étiquettes ont été séparées du trunk, et la meilleure façon de visualiser ce genre d'informations est sous la forme d'un graphique ou d'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

Pour générer le graphique, TortoiseSVN doit aller chercher tous les commentaires dans la racine du dépôt. Inutile de dire que cela peut prendre plusieurs minutes même avec un dépôt de seulement quelques milliers de révisions, selon la vitesse du serveur, la bande passante du réseau, etc. Si vous essayez avec quelque chose comme le projet Apache, qui a actuellement plus de 500 000 révisions, vous pourriez attendre un moment.

La bonne nouvelle est que vous si mettez le journal en cache, vous n'aurez à attendre qu'une fois. Par la suite, ces données seront stockées localement. Ce paramètre est activé dans la configuration de TortoiseSVN.

4.26.1. Nœuds des graphiques de révision

Chaque nœud du graphe représente une révision dans le dépôt dans laquelle quelque chose a changé dans l'arborescence que vous examinez. Les différents types de nœuds peuvent être distingués par leur forme et leur couleur. Les formes sont fixes, mais les couleurs peuvent être définies à l'aide de TortoiseSVN → Paramètres

Éléments ajoutés ou copiés
: Les éléments qui ont été ajoutés ou créés en copiant un autre fichier/dossier sont affichés avec un rectangle arrondi. La couleur par défaut est le vert. Les étiquettes et le trunk sont traités comme un cas particulier et utilisent une nuance différente, selon les TortoiseSVN → Paramètres.

Éléments supprimés
: Les éléments supprimés, p. ex. une branche qui n'est plus requise, sont affichés avec un octogone (un rectangle avec les coins coupés). La couleur par défaut est le rouge.

Éléments renommés
: Les éléments renommés seront également affichés avec un octogone, mais seront de couleur bleue par défaut.

Révision de pointe de branche
: Le graphique est normalement limité aux points de branchement, mais il peut aussi être souvent utile de voir la révision HEAD respective de chaque branche. Si vous sélectionnez Afficher les révisions HEAD, chaque nœud de révision HEAD sera affiché comme une ellipse. Notez que HEAD se réfère ici à la dernière révision livrée sur ce chemin, et non à la révision HEAD du dépôt.

Révision de la copie de travail
: Si vous avez invoqué le graphique de révision à partir d'une copie de travail, vous pouvez choisir de montrer la révision BASE sur le graphique avec Montrer la révision de la copie de travail, qui marque le nœud BASE avec un contour en gras.

Copie de travail modifiée
: Si vous avez invoqué le graphique de révision à partir d'une copie de travail, vous pouvez choisir de montrer un nœud supplémentaire représentant votre copie de travail modifiée avec Afficher les modifications de la copie de travail. Il s'agit d'un nœud elliptique avec un contour en gras rouge par défaut.

Élément normal
: Tous les autres éléments sont affichés en utilisant un rectangle ordinaire.

Notez que par défaut le graphique ne montre que les points auxquels des éléments ont été ajoutés, copiés ou supprimés. Afficher chaque révision d'un projet pourrait produire un graphique très grand pour des cas non triviaux. Si vous voulez vraiment voir *toutes* les révisions pour lesquelles des modifications ont été apportées, il existe une option pour ce faire dans le menu Affichage et sur la barre d'outils.

L'affichage par défaut (sans regroupement) place les nœuds de telle sorte que leur position verticale reflète l'ordre strict de révision, afin que vous ayez un repère visuel pour l'ordre dans lequel les choses ont été faites. Lorsque deux nœuds sont dans la même colonne, l'ordre est très évident. Lorsque deux nœuds sont dans des colonnes voisines, le décalage est beaucoup plus faible car il n'est pas nécessaire d'empêcher les nœuds de se chevaucher, et en conséquence l'ordre est un peu moins évident. Ces optimisations sont nécessaires pour que les graphiques complexes conservent une taille raisonnable. Veuillez noter que cette organisation utilise le *bord* du nœud sur l'*ancien* côté comme une référence, c'est à dire le bord inférieur du nœud lorsque le graphique est représenté avec le plus ancien nœud en bas. Le bord de référence est important, car les formes des nœuds n'ont pas toutes la même hauteur.

4.26.2. Changer l'affichage

Parce qu'un graphique de révision est souvent très complexe, il existe un certain nombre de caractéristiques qui peuvent être utilisées pour adapter la façon dont vous souhaitez le voir. Celles-ci sont disponibles dans le menu Voir et à partir de la barre d'outils.

Grouper les branches
: Le comportement par défaut (sans regroupement) montre toutes les lignes triées strictement par révision. En conséquence, les branches très anciennes avec des modifications clairsemées occupent une colonne entière pour seulement quelques changements et le graphe devient très large.

  Ce mode groupe les changements par branche, de sorte qu'il n'y ait pas de tri global par révision : des révisions consécutives sur une branche seront (généralement) affichées sur des lignes consécutives. Les sous-branches, en revanche, sont arrangées de telle façon que les branches postérieures seront affichées dans la même colonne au-dessus des branches antérieures pour limiter la largeur du graphique. En conséquence, une ligne donnée peut contenir les changements de plusieurs révisions différentes.

Les plus anciens en haut
: Normallement, le graphique affiche les révisions les plus anciennes en bas, et l'arborescence se développe par le haut. Utilisez cette option pour que cette dernière s'affiche de haut en bas.

Aligner les arborescences en haut
: Lorsqu'un graphique est divisé en plusieurs petits arbres, les arbres peuvent apparaître soit dans l'ordre naturel de révision, soit alignés au bas de la fenêtre, selon que vous utilisez ou pas l'option Grouper les branches. Utilisez cette option pour développer tous les arbres du haut vers le bas.

Réduire les intersections
: Cette option est normalement active et évite d'afficher le graphique avec de nombreuses lignes confuses qui se croisent. Cependant, cela peut aussi faire apparaître les colonnes de l'affichage à des endroits moins logiques, par exemple en ligne diagonale plutôt qu'une colonne verticale, et cela peut nécessiter plus de place pour dessiner le graphique. Si cela pose problème, vous pouvez désactiver l'option dans le menu Affichage.

Chemins d'accès différentiels
: Des noms de chemin longs peuvent prendre beaucoup d'espace et rendre les cases des nœuds très large. Utilisez cette option pour afficher seulement la partie modifiée d'un chemin, en remplaçant la partie commune avec des points. Par exemple, si vous créez une branche `/branches/1.2.x/doc/html` à partir de `/trunk/doc/html`, la branche pourrait être affichée sous forme compacte `/branches/1.2.x/..` parce que les deux derniers niveaux, `doc` et `html`, n'ont pas changé.

Montrer toutes les révisions
: Cela fait exactement ce que vous attendiez et montre chaque révision où quelque chose (dans l'arbre que vous dessinez) a changé. Pour un long historique, cela peut produire un graphique véritablement énorme.

Montrer la version de tête (HEAD)
: Cela garantit que la dernière révision de chaque branche est toujours indiquée sur le graphique.

Copie exacte des sources
: Quand une branche / étiquette est créée, le comportement par défaut est de montrer la branche comme tirée du dernier nœud où une modification a été apportée. Strictement parlant cela est inapproprié puisque les branches sont souvent issues du HEAD courant plutôt que d'une révision spécifique. Il est donc possible de montrer la révision la plus juste (mais la moins utile) qui a été utilisé pour créer la copie. Notez que cette révision peut être plus récente que la révision HEAD de la branche source.

Replier les étiquettes
: Quand un projet a beaucoup d'étiquettes, afficher chaque étiquette comme un nœud distinct sur le graphique prend beaucoup de place et prend le pas sur la structure de branche de développement qui est pourtant plus intéressante. En même temps, vous pouvez avoir besoin d'accéder facilement au contenu de l'étiquette pour pouvoir comparer les révisions. Cette option cache les nœuds pour les étiquettes et à la place, les affiche sur l'infobulle pour le nœud depuis lequel elles ont été copiées. Une icône d'étiquette sur le côté droit du nœud source indique que des étiquettes ont été créées. Cela simplifie grandement l'affichage.

  Notez que si un tag est lui-même utilisé comme source pour obtenir une copie, peut-être une nouvelle branche basée sur un tag, alors ce tag sera affiché comme un nœud distinct plutôt que plié.

Cacher les répertoires supprimés
: Cache les chemins qui ne sont plus présents dans la révision HEAD du dépôt, par exemple, des branches supprimées.

  Si vous avez sélectionné l'option Replier les étiquettes, alors une branche effacée sur laquelle des étiquettes ont été prises sera toujours affichée, sinon les étiquettes disparaîtraient aussi. La dernière révision qui a été étiquetée sera affichée avec la couleur utilisée pour les nœuds effacés au lieu d'afficher une révision de suppression séparée.

  Si vous sélectionnez l'option Masquer les balises, alors ces branches vont disparaître à nouveau puisqu'elles ne sont pas nécessaires pour afficher les étiquettes.

Cacher les branches inutilisées
: Cache les branches où aucun changement n'a été livré dans le fichier ou le sous-dossier respectif. Cela ne signifie pas nécessairement que la branche n'a pas été utilisée, simplement qu'aucune modification n'a été apportée à *cette* partie de la branche.

Montrer la version de la CdT
: Marque la révision sur le graphique qui correspond à la révision de mise à jour de l'élément pour lequel vous affichez le graphique. Si vous venez de mettre à jour, ce sera HEAD, mais si d'autres personnes ont livré des changements depuis votre dernière mise à jour, votre CdT peut se retrouver à quelques révisions en-dessous. Le nœud est marqué en lui donnant un contour gras.

Montrer les modifications de la CdT
: Si votre CdT contient des modifications locales, cette option la représente comme un nœud séparé elliptique, lié au nœud pour lequel votre CdT a été mise à jour en dernier. La couleur du contour par défaut est le rouge. Vous pouvez avoir besoin d'actualiser le graphique à l'aide de **F5** pour afficher les changements récents.

Filtrer
: Parfois, le graphique de révision contient plus de révisions que vous ne voulez en voir. Cette option ouvre une boîte de dialogue qui vous permet de restreindre la plage des révisions affichées, et de cacher des chemins particuliers par leur nom.

  Si vous cachez un chemin donné et que ce nœud a des nœuds fils, les fils seront affichés comme un arbre séparé. Si vous voulez également cacher les fils, utilisez la case à cocher Supprimer le(s) sous-répertoire(s).

Arbre à rayures
: Lorsque le graphique contient plusieurs arbres, il est parfois intéressant d'utiliser des couleurs de fond alternées pour pouvoir les distinguer.

Montrer l'aperçu
: Affiche une petite image du graphique en entier, avec la fenêtre courante dans un rectangle que vous pouvez faire glisser. Cela vous permet de naviguer sur le graphique plus facilement. Notez que pour les très grands graphiques, l'aperçu peut devenir inutile en raison du facteur de zoom extrême et ne sera donc pas montré dans de tels cas.

4.26.3. Utiliser le graphique

Pour rendre plus facile la navigation dans un large graphe, utilisez la fenêtre Aperçu. Cela montre le graphique entier dans une petite fenêtre, avec la partie actuellement affichée en surbrillance. Vous pouvez faire glisser la zone en surbrillance pour changer la région affichée.

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.

Si vous sélectionnez deux révisions (faites un clic gauche avec Ctrl), vous pouvez utiliser le menu contextuel pour afficher les différences entre ces révisions. Vous pouvez vouloir afficher les différences comme aux points de création de branche, mais le plus souvent vous voudrez afficher les différences aux points de fin de branche, c'est-à-dire à la révision HEAD.

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.

Vous pouvez également fusionner les modifications dans la (les) révision(s) sélectionnée(s) dans une autre copie de travail. Une boîte de dialogue de sélection de dossier vous permet de choisir la copie de travail vers laquelle fusionner, mais après cela il n'y a pas de demande de confirmation, ni possibilité de mener un test de fusion. C'est une bonne idée de fusionner dans une copie de travail non modifiée afin que vous puissiez annuler les modifications si elles ne fonctionnent pas ! Cette fonctionnalité est utile si vous souhaitez fusionner les versions sélectionnées d'une branche à l'autre.

Apprendre à lire le graphique de révision

Les utilisateurs qui affichent un graphique pour la première fois peuvent être surpris par le fait que le graphique de révision montre quelque chose qui ne correspond pas au modèle auquel pense l'utilisateur. Si une révision change des copies multiples ou des branches d'un fichier ou d'un dossier, par exemple, alors il y aura plusieurs nœuds pour cette seule révision. Il est bon de commencer avec les options par défaut dans la barre d'outils et de personnaliser graphique étape par étape jusqu'à ce qu'il se rapproche de votre vision.

Toutes les options de filtre essaient de perdre le moins d'informations possible. Cela peut provoquer le changement de couleur de certains nœuds, par exemple. Chaque fois que le résultat est inattendu, annuler la dernière opération de filtrage et essayer de comprendre ce qui est particulier à cette révision ou cette branche. Dans la plupart des cas, le résultat initialement prévu de l'opération de filtrage est soit inexact soit trompeur.

4.26.4. Rafraîchissement de l'affichage

Si vous voulez vérifier s'il y a de nouvelles informations sur le serveur, il vous suffit de rafraîchir la vue en utilisant **F5**. Si vous utilisez le cache du journal (activé par défaut), cela va vérifier les livraisons les plus récentes dans le dépôt et récupérer seulement les nouvelles. Si le cache du journal était en mode hors connexion, cela va aussi tenter de se reconnecter.

Si vous utilisez le cache du journal et que vous pensez que le contenu du message ou l'auteur a pu changer, vous devriez utiliser la boîte de dialogue du journal pour actualiser les messages dont vous avez besoin. Puisque le graphique des révisions se base sur la racine du dépôt, nous aurions à invalider intégralement le cache du journal, et le remplir à nouveau pourrait prendre un temps *très* long.

4.26.5. Élagage des arborescences

Il peut être difficile de naviguer dans un grand arbre et parfois vous souhaiterez en cacher certaines parties, ou le décomposer en une forêt de plus petits arbres. Si vous passez la souris sur le point où un lien entre ou sort d'un nœud, vous verrez un ou plusieurs boutons popup qui vous permettent de le faire.

Cliquez sur le bouton moins pour réduire la sous-arborescence.

Cliquez sur le bouton plus pour développer une sous-arborescence réduite. Quand une arborescence a été réduite, ce bouton reste visible pour signaler 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 cercle pour ré-attacher une sous-arborescence préalablement séparée. Quand une arborescence a été séparée, ce bouton reste visible pour signaler qu'il y a une sous-arborescence séparée.

Cliquez sur l'arrière-plan du graphique pour le menu contextuel principal, qui offre des options pour Tout développer et Tout replier. Si aucune branche n'a été repliée ou séparée, le menu contextuel ne sera pas montré.

4.27. Exporter une copie de travail Subversion

Vous pouvez parfois vouloir une copie propre de votre arborescence de travail sans le répertoire `.svn`, par exemple pour créer une tarball zippée de votre source, ou pour l'exporter sur un serveur web. Au lieu de faire une copie puis d'effacer le répertoire `.svn` manuellement, TortoiseSVN offre la commande TortoiseSVN → Exporter.... Exporter depuis une URL et exporter depuis une copie de travail sont traités un peu différemment.

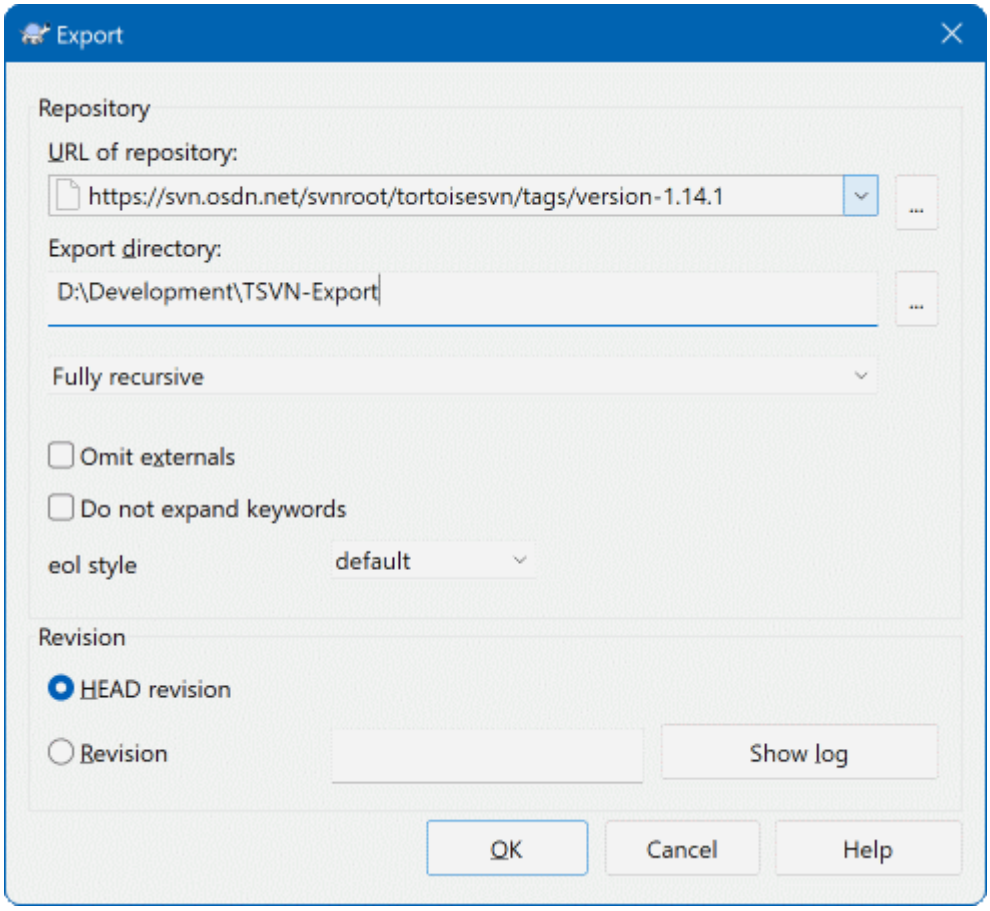


Figure 4.68. La boîte de dialogue d'exportation depuis une URL

Si vous exécutez cette commande sur un dossier non versionné, TortoiseSVN va supposer que le dossier sélectionné est la cible, et va ouvrir une boîte de dialogue pour que vous entriez l'URL et la révision vers lesquelles exporter. Cette boîte de dialogue a des options pour exporter seulement le dossier de niveau supérieur, pour omettre des références externes et pour remplacer le style de fin de ligne pour les fichiers pour lesquels `svn:eol-style` est activé.

Bien sûr, vous pouvez aussi exporter directement depuis le dépôt. Utilisez l'explorateur de dépôt pour naviguer au sous-arbre pertinent dans votre dépôt, utilisez ensuite Menu contextuel → Exporter. Vous obtiendrez la boîte de dialogue Exporter depuis l'URL décrite ci-dessus.

Si vous exécutez cette commande sur votre copie de travail, il vous sera demandé un emplacement pour sauvegarder la copie de travail *propre* sans le dossier `.svn`. Par défaut, seuls les fichiers versionnés sont exportés, mais vous pouvez utiliser la case à cocher Exporter aussi les fichiers non versionnés pour inclure tous les autres fichiers non versionnés qui existent dans votre CdT et pas dans le dépôt. Les références externes qui utilisent `svn:externals` peuvent être omises si besoin.

Une autre façon d'exporter depuis une copie de travail est de glisser avec le bouton droit le dossier de copie de travail vers un autre endroit et de choisir Menu contextuel → SVN Exporter les éléments versionnés ici ou Menu contextuel → SVN Exporter tous les éléments ici ou Menu contextuel → SVN Exporter les éléments modifiés ici. La deuxième option inclut également les fichiers non versionnés. La troisième option n'exporte que les éléments modifiés, mais conserve la structure des dossiers.

Lors d'une exportation depuis une copie de travail, si le dossier cible contient déjà un dossier qui porte le même nom que celui que vous voulez exporter, vous aurez le choix entre écraser le contenu existant ou bien créer un nouveau dossier avec un nom généré automatiquement, par exemple `Destination (1)`.

Exporter des fichiers individuellement

La boîte de dialogue d'exportation ne permet pas d'exporter des fichiers individuellement, même si Subversion le permet.

Pour exporter des fichiers individuellement avec TortoiseSVN, vous devez utiliser l'explorateur de dépôt (Section 4.25, « L'explorateur de dépôt »). Il suffit de faire glisser le(s) fichier(s) que vous souhaitez exporter depuis l'explorateur de dépôt vers l'endroit où vous le souhaitez dans l'explorateur, ou d'utiliser le menu contextuel dans l'explorateur de dépôt pour exporter les fichiers.

Exporter une arborescence des modifications

Si vous voulez exporter une copie de votre arborescence de projet mais ne contenant que les fichiers qui ont changé dans une révision particulière, ou entre deux révisions, utilisez la fonctionnalité de comparaison de révisions décrite dans Section 4.11.3, « Comparer des dossiers ».

Si vous voulez exporter la structure d'arborescence de votre copie de travail mais ne contenant que les fichiers qui ont été modifiés localement, référez-vous à SVN Exporter les éléments modifiés ici ci-dessus.

4.27.1. Retirer une copie de travail du contrôle de version

Parfois vous avez une copie de travail que vous voulez faire revenir à l'état de dossier normal sans le répertoire `.svn`. Tout ce que vous avez à faire est de supprimer le répertoire `.svn` de la racine de la copie de travail.

Une alternative possible est d'exporter le dossier vers lui-même. Dans l'explorateur Windows, faites glisser avec le bouton droit le dossier racine de la copie de travail depuis le panneau de liste des fichiers vers lui-même dans le panneau d'arborescence des dossiers. TortoiseSVN détecte ce cas spécial et vous demande si vous voulez déversionner la copie de travail. Si vous répondez *oui*, le répertoire de contrôle sera supprimé et vous aurez une arborescence non versionnée.

4.28. Relocaliser une copie de travail

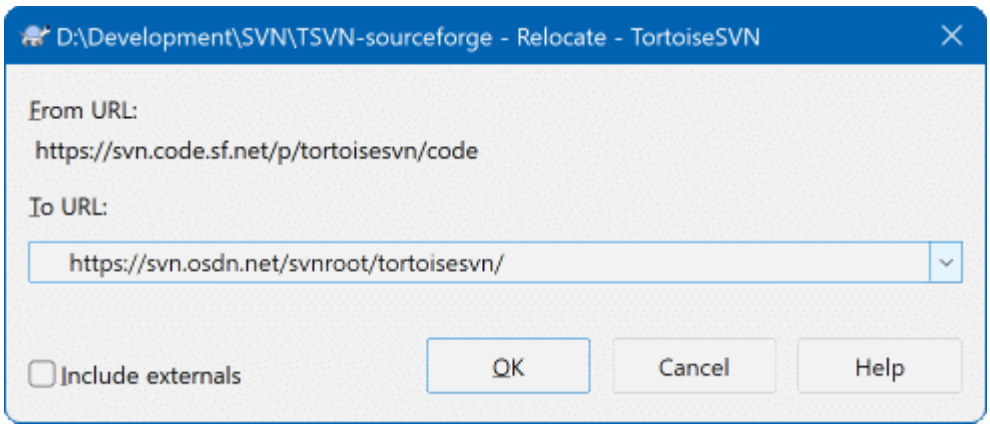


Figure 4.69. La boîte de dialogue Relocaliser

Si votre dépôt a, pour une raison ou une autre, changé d'emplacement (IP/URL), peut-être même que vous êtes coincé, que vous ne pouvez pas livrer et que vous ne voulez pas faire une nouvelle extraction de votre copie de travail depuis le nouvel emplacement et déplacer toutes vos modifications vers la nouvelle copie de travail,

TortoiseSVN → Relocaliser est la commande que vous recherchez. À la base, elle fait très peu de choses : elle réécrit toutes les URLs associées à chaque fichier et dossier avec la nouvelle URL.

> **Note**
>
> Cette opération ne fonctionne que pour les *racines* de copies de travail. De ce fait, l'entrée du menu contextuel n'est visible que pour les racines de copies de travail.

Vous pourriez être surpris de constater que TortoiseSVN contacte le dépôt dans le cadre de cette opération. Tout ce qu'il fait est l'exécution de contrôles simples pour faire en sorte que la nouvelle URL se réfère vraiment au même dépôt que la copie de travail existante.

> **Avertissement**
>
> *C'est une opération très rare*. La commande relocaliser est utilisée *seulement* si l'URL de la racine du dépôt 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 dépôt a changé dans le paramétrage du serveur.
>
> Autrement dit, vous devez relocaliser quand votre copie de travail se réfère au même emplacement dans le même dépôt, mais que le dépôt lui-même a été déplacé.
>
> Cela ne s'applique pas si :
>
> - Vous voulez migrer vers un dépôt Subversion différent. Dans ce cas, vous devriez exécuter une extraction propre à partir du nouvel emplacement du dépôt.
> - Vous voulez basculer sur une branche différente ou sur un répertoire différent dans le même dépôt. Pour ce faire, vous devriez utiliser TortoiseSVN → Aller sur.... Lisez Section 4.20.3, « 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, le seul remède est de refaire une extraction propre.

4.29. Intégration avec des systèmes de gestion de bug / gestion 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) peuvent 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 une source d'erreurs puisque l'utilisateur a la responsabilité d'écrire correctement le commentaire afin 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 gestionnaire 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.

4.29.1. Ajouter des numéros d'incidents aux messages de log

Vous pouvez intégrer le traqueur de bugs de votre choix dans TortoiseSVN. Pour ce faire, vous devez définir quelques propriétés, qui commencent par `bugtraq:`. Elles doivent être définies sur les dossiers : (Section 4.18, « Configuration des projets »)

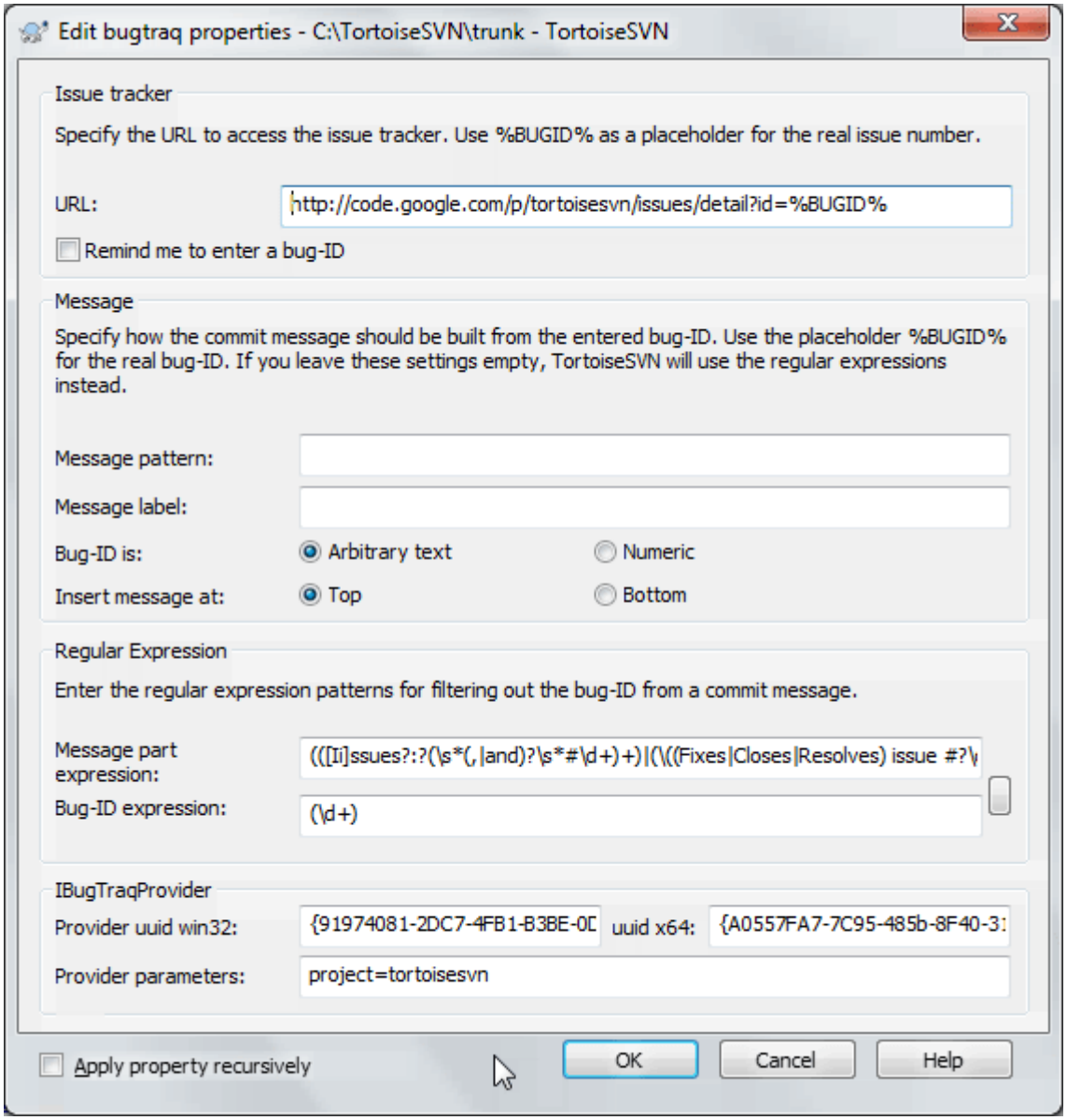


Figure 4.70. La boîte de dialogue des propriétés bugtraq

Quand vous éditez une propriété bugtraq, un éditeur de propriétés spécial est utilisé pour faciliter la saisie de valeurs appropriées.

Il y a deux façons d'intégrer TortoiseSVN avec les gestionnaires d'incidents. L'une est basée sur des chaînes simples, l'autre sur les `expressions régulières`. Les propriétés utilisées par les deux approches sont :

bugtraq:url
: Positionnez cette propriété sur l'URL de votre outil de suivi de bogue. Elle doit être correctement encodée en URI et doit contenir `%BUGID%`. Le paramètre `%BUGID%` est remplacé par le numéro d'incident que vous

avez saisi. Cela permet à TortoiseSVN d'afficher un lien dans la boîte de dialogue de journal, pour que vous puissiez basculer directement vers votre outil de suivi de bogue depuis le journal de révision. Il n'est pas obligatoire de fournir cette propriété, mais dans ce cas TortoiseSVN n'affiche que le numéro de l'incident et pas le lien vers celui-ci. Par exemple, le projet TortoiseSVN utilise `http://issues.tortoisesvn.net/?do=details&id=%BUGID%`.

Vous pouvez aussi utiliser des URLs relatives au lieu d'URLs absolues. Cela peut être utile quand votre gestionnaire d'incidents est sur le même serveur que votre dépôt. Ainsi, si vous changez de nom de domaine, vous n'aurez pas à modifier la propriété `bugtraq:url`. Il y a deux manières de spécifier une URL relative :

Si elle commence par la chaîne `^/`, elle est relative à la racine du dépôt. Par exemple, `^/../?do=details&id=%BUGID%` va se résoudre en `https://tortoisesvn.net/?do=details&id=%BUGID%` si votre dépôt est situé sur `https://tortoisesvn.net/svn/trunk/`.

Une URL qui commence par la chaîne `/` est relative au nom d'hôte du serveur. Par exemple, `/?do=details&id=%BUGID%` va se résoudre en `https://tortoisesvn.net/?do=details&id=%BUGID%` si votre dépôt est situé quelque part sur le domaine `https://tortoisesvn.net`.

bugtraq:warnifnoissue
: Positionnez cette valeur à `true` si vous voulez que TortoiseSVN vous avertisse quand le champ du numéro d'incident est vide. Les valeurs valables sont `true/false`. *Si elle n'est pas définie, `false` est prise par défaut.*

4.29.1.1. Numéro d'incident 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
: Cette propriété active le système de gestion de bugs en mode *champ de saisie*. Si cette propriété est définie, alors TortoiseSVN vous demandera d'entrer un numéro d'incident quand vous livrez vos changements. Elle est utilisée pour ajouter une ligne à la fin du commentaire. Elle doit contenir `%BUGID%`, qui est remplacé par le numéro d'incident à la livraison. Cela assure que votre journal de livraison contient une référence au numéro d'incident qui est toujours dans un format cohérent et peut être analysé syntaxiquement par votre outil de gestion de bugs pour associer le numéro d'incident à une livraison particulière. Par exemple vous pourriez utiliser `Incident : %BUGID%`, mais cela dépend de votre outil.

bugtraq:label
: Ce texte est affiché par TortoiseSVN sur la boîte de dialogue de livraison comme label pour la boîte de saisie où vous entrez le numéro d'incident. S'il n'est pas défini, `Bug-ID/ N° d'incident` sera affiché. Gardez à l'esprit que la fenêtre ne sera pas redimensionnée pour s'adapter à ce label, gardez donc la taille du label en-dessous de 20-25 caractères.

bugtraq:number
: Si elle est définie à `true`, seuls les nombres sont autorisés dans le champ du numéro d'incident. La virgule est une exception, donc vous pouvez séparer plusieurs numéros par des virgules. Les valeurs valides sont `true/false`. *Si elle n'est pas définie, `true` est la valeur par défaut.*

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.*

4.29.1.2. Numéros d'incidents utilisant des expressions régulières

Dans l'approche avec les `expressions régulières`, TortoiseSVN n'affiche pas de champ de saisie séparé, mais marque la partie du commentaire que l'utilisateur entre qui est reconnue par le gestionnaire d'incidents. Cela se fait pendant que l'utilisateur écrit le commentaire. Cela signifie aussi que l'ID de bug peut être n'importe où à l'intérieur d'un commentaire ! Cette méthode est beaucoup plus souple et c'est celle utilisée par le projet TortoiseSVN lui-même.

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.

  Si deux expressions sont définies, alors la première est utilisée comme pré-filtre pour trouver les expressions qui contiennent des IDs de bogue. La seconde expression extrait alors les IDs de bogue du résultat de la première expression régulière. Cela vous permet d'utiliser une liste d'IDs de bogue avec du langage courant si vous le souhaitez. Par exemple, vous pourriez avoir résolu plusieurs bogues et saisir une chaîne de ce type : « Cette modification résout les incidents #23, #24 et #25 ».

  Si vous souhaitez extraire les IDs de bogue d'un message de journal du type de celui de l'expression ci-dessus, vous pourriez utiliser les expressions régulières suivantes, qui sont celles utilisées par le projet TortoiseSVN : `[Ii]ncidents?:?(\s*(,|et)?\s*#\d+)+` et `(\d+)`.

  La première expression extrait « incidents #23, #24 et #25 » du message de journal. La seconde regex extrait les nombres décimaux du résultat de la première regex, on aura donc « 23 », « 24 » et « 25 » à utiliser comme ID de bogue.

  Détaillons un peu la première regex. Elle doit commencer par le mot « incident », dont la première lettre peut être une majuscule. Elle continue par un « s » optionnel (s'il y a plusieurs incidents) et deux points également optionnels. Elle continue ensuite par un ou plusieurs groupes ayant chacun zéro, un ou plusieurs espaces de tête, une virgule optionnelle ou le mot « et » et d'autres espaces optionnels. Enfin, il y a un « # » obligatoire et un nombre décimal obligatoire.

  Si seule une expression est définie, alors l'ID du bug seul doit correspondre aux groupes de la chaîne regex. Exemple : `[Ii]ssue(?:s)? #?(\d+)`. Cette méthode est requise par quelques gestionnaires d'incidents, par exemple trac, mais il est plus difficile de construire la regex. Nous vous recommandons d'utiliser cette méthode uniquement si la documentation de votre gestionnaire d'incidents vous le demande.

  Si les expressions régulières ne vous sont pas familières, vous pouvez consulter une introduction sur https://fr.wikipedia.org/wiki/Expression_r%C3%A9guli%C3%A8re, et de la documentation en ligne et un tutoriel sur http://www.regular-expressions.info/.

  Il n'est pas toujours facile de trouver la bonne regex. Pour vous y aider, il y a une boîte de dialogue de test intégrée à la boîte de dialogue de propriétés bugtraq. Cliquez sur le bouton à droite des zones de saisie pour la faire apparaître. Vous pouvez y saisir des exemples de texte, et modifier chaque regex pour voir les résultats. Si la regex est invalide, le fond de la zone de saisie passe en rouge.

Si les deux propriétés `bugtraq:message` et `bugtraq:logregex` sont définies, `logregex` a la priorité.

Astuce

Même si vous n'avez pas de gestionnaire d'incidents avec un hook de pré-livraison analysant syntaxiquement vos commentaires, vous pouvez toujours utiliser cela pour transformer les incidents mentionnés dans vos commentaires en liens !

Et même si vous n'avez pas besoin de liens, les numéros des incidents apparaissent dans une colonne distincte dans la boîte de dialogue de log, facilitant la recherche des changements qui se rapportent à un incident particulier.

Certaines propriétés `tsvn:` exigent une valeur `true/false`. TortoiseSVN comprend aussi `yes` comme synonyme de `true` et `no` comme synonyme de `false`.

Positionnez les propriétés sur les dossiers

Ces propriétés doivent être positionnées sur les dossiers pour que le système fonctionne. Quand vous livrez un fichier ou un dossier, ce sont les propriétés de ce dossier qui sont lues. S'il n'y trouve

pas les propriétés, TortoiseSVN va les chercher en remontant l'arborescence jusqu'à ce qu'il arrive à un dossier non versionné ou à la racine (par exemple `C:\`). Si vous pouvez être sûr que chaque utilisateur extrait à partir du même endroit, par exemple de `trunk/` et pas d'un sous-dossier, alors il suffit de positionner les propriétés sur `trunk/`. Si vous ne pouvez pas en être sûr, vous devriez positionner les propriétés sur chaque sous-dossier de manière récursive. Une propriété positionnée plus bas dans la hiérarchie du projet surcharge celles positionnées sur les niveaux plus élevés (c'est-à-dire plus près de `trunk/`).

Depuis la version 1.8, TortoiseSVN et Subversion utilisent ce qu'on appelle des `propriétés héritées`, ce qui signifie qu'une propriété positionnée sur un dossier est aussi implicitement positionnée sur tous les sous-dossiers automatiquement. Il n'est donc plus nécessaire de positionner les propriétés sur tous les dossiers, mais uniquement sur le dossier racine.

Pour les propriétés de projet *uniquement*, c'est-à-dire `tsvn:`, `bugtraq:` et `webviewer:` vous pouvez utiliser la case à cocher Récursif pour appliquer la propriété à tous les sous-dossiers, sans avoir à la mettre aussi sur tous les fichiers.

Lorsque vous ajoutez un nouveau sous-dossier à une copie de travail via TortoiseSVN, toutes les propriétés de projet du dossier parent seront automatiquement ajoutées à ce nouveau dossier fils.

Aucune information sur le suivi des incidents à partir du navigateur du dépôt

Comme l'intégration du suivi d'incidents repose sur l'accès aux propriétés Subversion, vous n'en verrez les résultats que quand vous travaillez sur une copie de travail extraite. Récupérer des propriétés sur le serveur est une opération lente, donc vous ne verrez pas cette fonctionnalité à l'œuvre dans le navigateur de dépôt, à moins que vous n'ayez démarré le navigateur de dépôt depuis votre copie de travail. Si vous avez démarré le navigateur de dépôt en saisissant l'URL du dépôt, vous ne verrez pas cette fonctionnalité.

Pour la même raison, les propriétés du projet ne seront pas propagées automatiquement quand un dossier enfant est ajouté en utilisant le navigateur du dépôt.

L'intégration de suivi d'incidents n'est pas limitée à TortoiseSVN ; elle peut être utilisée avec n'importe quel client Subversion. Pour plus d'informations, lisez la *spécification de l'intégration de suivi d'incidents* [https://svn.code.sf.net/p/tortoisesvn/code/trunk/doc/notes/issuetrackers.txt] dans le dépôt des sources de TortoiseSVN. (La Section 3, « Licence » détaille comment accéder au dépôt.)

4.29.2. Récupérer des informations depuis le gestionnaire d'incidents

La section précédente porte sur l'ajout d'information concernant les incidents dans des messages de log. Mais que faire si vous avez besoin d'obtenir des informations du gestionnaire d'incidents ? La boîte de dialogue de révision possède une interface COM qui permet l'intégration d'un programme externe qui peut dialoguer avec votre logiciel de gestion. Typiquement, vous voudrez peut-être interroger le gestionnaire pour obtenir la liste des incidents ouverts qui vous sont assignés, de sorte que vous pouvez choisir les incidents qui sont abordées dans cette livraison.

Toute interface de ce type est évidemment extrêmement spécifique à votre système de gestion d'incidents, ce qui fait que nous ne pouvons pas en fournir, et décrire comment créer un tel logiciel dépasse le cadre de ce manuel. La définition d'interface et les exemples de plugin en C# et C++/ATL sont disponibles dans le dossier `contrib` du *dépôt TortoiseSVN* [https://svn.code.sf.net/p/tortoisesvn/code/trunk/contrib/issue-tracker-plugins]. (La Section 3, « Licence » détaille comment accéder au dépôt.) Un résumé de l'API est également fourni au Chapitre 7, *IBugtraqProvider interface*. Un autre exemple (fonctionnel) de plugin en C# est *Gurtle* [http://code.google.com/p/gurtle/], qui implémente l'interface COM requise pour interagir avec le gestionnaire d'incidents de *Google Code* [http://code.google.com/hosting/].

À titre d'exemple, supposons que votre administrateur système vous a fourni un plugin de gestionnaire d'incidents que vous avez installé, et que vous avez paramétré certaines de vos copies de travail pour utiliser le plugin dans la

boîte de configuration de TortoiseSVN. Lorsque vous ouvrez la boîte de dialogue de livraison à partir d'une copie de travail à laquelle le plugin a été attribué, vous pourrez voir un nouveau bouton en haut de la boîte de dialogue.

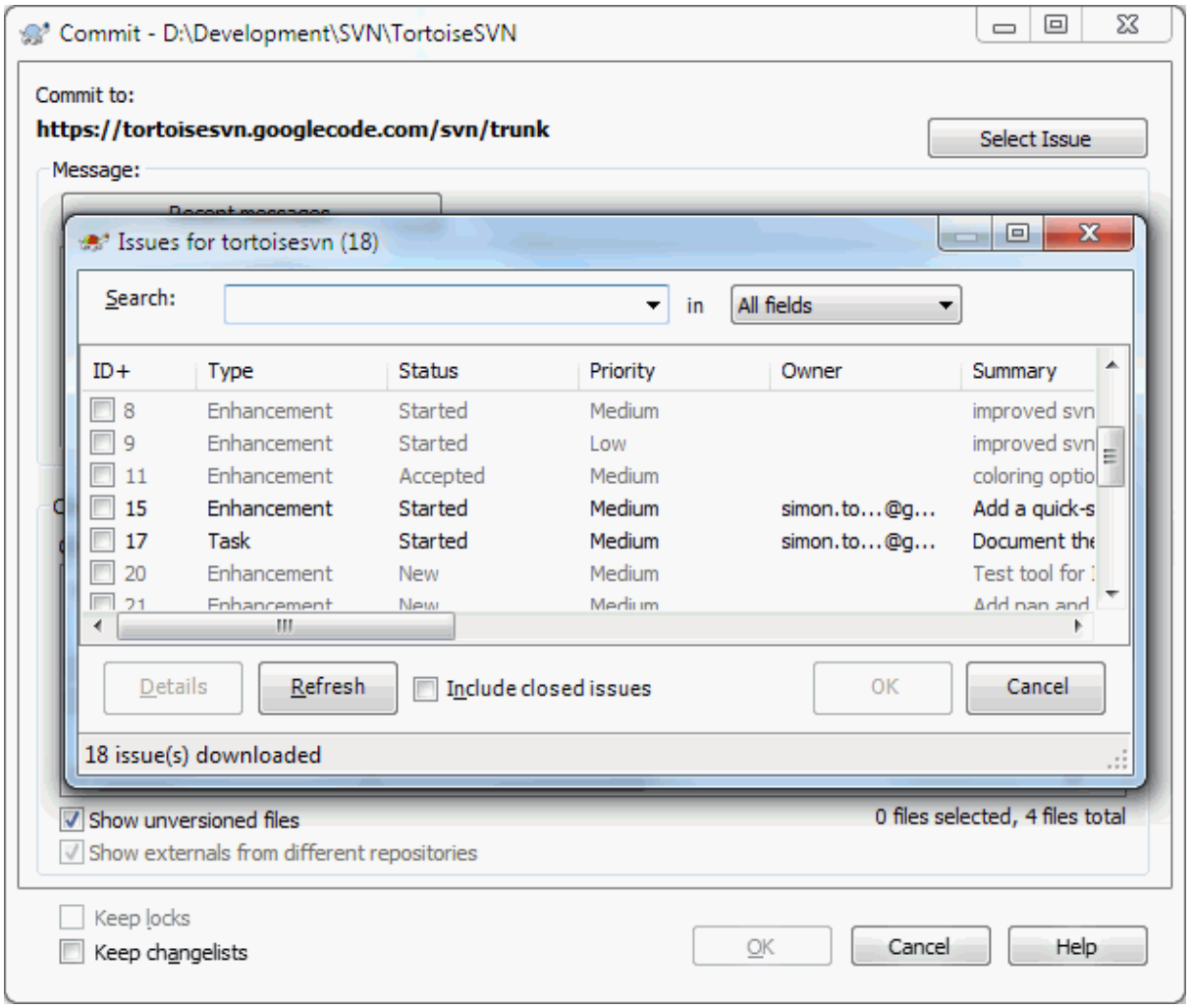


Figure 4.71. Exemple de fenêtre de gestionnaire d'incidents

Dans cet exemple, vous pouvez sélectionner un ou plusieurs incidents en suspens. Le plugin peut alors générer du texte spécialement mis en forme qu'il ajoute à votre message de log.

4.30. Intégration avec des visionneuses de dépôt web

Il existe plusieurs visionneuses de dépôt web compatibles avec Subversion, comme *ViewVC* [http://www.viewvc.org/] et *WebSVN* [https://websvnphp.github.io/]. TortoiseSVN vous fournit le moyen de vous relier à ces visionneuses.

Vous pouvez intégrer la visionneuse de dépôt de votre choix dans TortoiseSVN. Pour ce faire, vous n'avez qu'à renseigner quelques propriétés permettant la liaison. Elles sont à positionner sur les dossiers : (Section 4.18, « Configuration des projets »)

webviewer:revision
: Positionnez cette propriété à l'URL où votre visionneuse de dépôt vous affiche tous les changements d'une révision spécifique. Elle doit être correctement encodée comme une URI et doit contenir `%REVISION%`. `%REVISION%` est remplacé par le numéro de la révision en question. Cela permet à TortoiseSVN d'afficher un élément de menu contextuel dans la boîte de dialogue du journal menu contextuel → Voir la révision dans la visionneuse web.

webviewer:pathrevision
: Positionnez cette propriété à l'URL où votre visionneuse de dépôt vous affiche les changements d'un fichier spécifique dans une révision spécifique. Elle doit être correctement encodée comme une URI et doit contenir `%REVISION%` et `%PATH%`. `%PATH%` est remplacé par le chemin relatif à la racine du dépôt. Cela permet à TortoiseSVN d'afficher un élément de menu contextuel dans la boîte de dialogue du journal menu contextuel → Voir la révision pour le chemin dans la visionneuse web. Par exemple, si, dans le panneau du bas de la boîte de dialogue du journal, vous faites un clic droit sur l'entrée d'un fichier `/trunk/src/fichier`, alors le `%PATH%` dans l'URL sera remplacé par `/trunk/src/fichier`.

Vous pouvez utiliser des URLs relatives à la place des URLs absolues. Cela peut être utile quand votre visionneuse est sur le même serveur que votre dépôt. Ainsi, si vous changez de nom de domaine, vous n'aurez pas à modifier les propriétés `webviewer:revision` et `webviewer:pathrevision`. Le format est le même que pour la propriété `bugtraq:url`. Voir Section 4.29, « Intégration avec des systèmes de gestion de bug / gestion d'incidents ».

Positionnez les propriétés sur les dossiers

Ces propriétés doivent être positionnées sur les dossiers pour que le système fonctionne. Quand vous livrez un fichier ou un dossier, ce sont les propriétés de ce dossier qui sont lues. S'il n'y trouve pas les propriétés, TortoiseSVN va les chercher en remontant l'arborescence jusqu'à ce qu'il arrive à un dossier non versionné ou à la racine (par exemple `C:\`). Si vous pouvez être sûr que chaque utilisateur extrait à partir du même endroit, par exemple de `trunk/` et pas d'un sous-dossier, alors il suffit de positionner les propriétés sur `trunk/`. Si vous ne pouvez pas en être sûr, vous devriez positionner les propriétés sur chaque sous-dossier de manière récursive. Une propriété positionnée plus bas dans la hiérarchie du projet surcharge celles positionnées sur les niveaux plus élevés (c'est-à-dire plus près de `trunk/`).

Pour les propriétés de projet *uniquement*, c'est-à-dire `tsvn:`, `bugtraq:` et `webviewer:` vous pouvez utiliser la case à cocher Récursif pour appliquer la propriété à tous les sous-dossiers, sans avoir à la mettre aussi sur tous les fichiers.

Lorsque vous ajoutez un nouveau sous-dossier à une copie de travail via TortoiseSVN, toutes les propriétés de projet du dossier parent seront automatiquement ajoutées à ce nouveau dossier fils.

Limitations de l'utilisation de l'explorateur de dépôt

Comme l'intégration d'une visionneuse de dépôt repose sur l'accès aux propriétés Subversion, vous n'en verrez les résultats que quand vous travaillez sur une copie de travail extraite. Récupérer des propriétés sur le serveur est une opération lente, donc vous ne verrez pas cette fonctionnalité à l'œuvre dans le navigateur de dépôt, à moins que vous n'ayez démarré le navigateur de dépôt depuis votre copie de travail. Si vous avez démarré le navigateur de dépôt en saisissant l'URL du dépôt, vous ne verrez pas cette fonctionnalité.

Pour la même raison, les propriétés du projet ne seront pas propagées automatiquement quand un dossier enfant est ajouté en utilisant le navigateur du dépôt.

4.31. 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.

4.31.1. Configuration générale

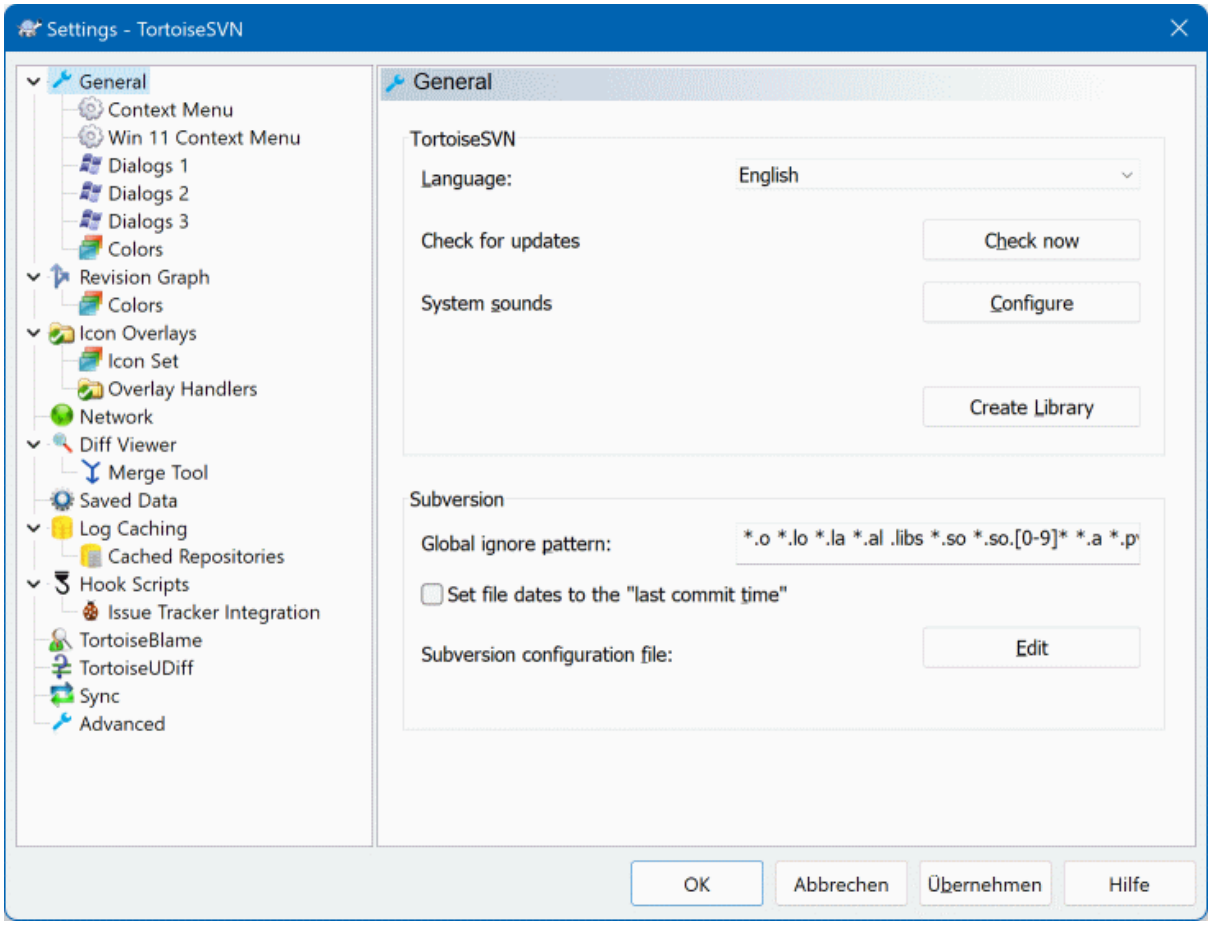


Figure 4.72. La boîte de dialogue Configuration, page Général

Cette boîte de dialogue vous permet de spécifier votre langue préférée et la configuration spécifique à Subversion.

Langue
: Sélectionne la langue de l'interface utilisateur. Naturellement, vous devez préalablement avoir installé le Language Pack correspondant pour avoir le choix d'une autre langue d'interface que l'anglais, qui est la valeur par défaut.

Vérifier les mises à jour
: TortoiseSVN entrera en contact avec son site de téléchargement périodiquement pour voir s'il y a une version plus récente du programme disponible. S'il y en a une, un lien de notification sera affiché dans la boîte de dialogue de livraison. Utilisez Vérifier maintenant si vous voulez une réponse tout de suite. La nouvelle version ne sera pas téléchargée; vous verrez simplement une boîte de dialogue d'information vous précisant 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.

Utiliser des boîtes de dialogue Aero
: Sur Windows Vista et les systèmes ultérieurs, cela détermine si les boîtes de dialogue utilisent le style Aero.

Créer une bibliothèque
: Sur Windows 7, vous pouvez créer une bibliothèque dans laquelle il est possible de regrouper des copies de travail qui sont dispersées en divers endroits sur votre système.

Modèle d'exclusion global

: Les modèles d'exclusion globaux sont utilisés pour empêcher des fichiers non versionnés d'apparaître, par exemple dans la boîte de dialogue de livraison. Les fichiers correspondant aux modèles sont aussi ignorés lors d'un import. Ignorez des fichiers ou des répertoires en saisissant les noms ou les extensions. Les modèles sont séparés par des espaces, par exemple `bin obj *.bak *.~?? *.jar *.[Tt]mp`. Ces modèles ne doivent contenir aucun séparateur de chemin. Notez également qu'il n'y a aucun moyen de différencier les fichiers des répertoires. Lisez Section 4.14.1, « Correspondance avec des modèles dans la liste des ignorés » pour plus d'information sur la syntaxe de correspondance des modèles.

: 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. Il n'est pas versionné, ce qui signifie qu'il n'affectera pas les autres utilisateurs. Inversement, vous pouvez aussi utiliser les propriétés versionnées `svn:ignore` ou `svn:global-ignores` pour exclure des fichiers ou des répertoires de la gestion de version. Lisez Section 4.14, « Ignorer des fichiers et des répertoires » pour plus d'informations.

Met les dates de fichier aux « dates de dernière livraison »
: Cette option indique à TortoiseSVN de mettre la date des fichiers à la date de dernière livraison lors d'une extraction ou d'une mise à jour. Autrement TortoiseSVN utilisera la date actuelle. Si vous développez des logiciels, il est généralement mieux d'utiliser la date actuelle parce que les systèmes de compilation regardent normalement les dates pour décider quels fichiers ont besoin d'être compilés. Si vous utilisez « la date de dernière livraison » et revenez à une révision de fichier plus vieille, votre projet peut ne pas se compiler comme vous vous y attendez.

Fichier de configuration de Subversion
: Utilisez Editer pour éditer directement le fichier de configuration de Subversion. Certains paramètres ne peuvent pas être modifiés directement par TortoiseSVN, et doivent être définis ici à la place. Pour plus d'informations à propos du fichier de `config` de Subversion, consultez le chapitre *Zone de configuration des exécutables* [http://svnbook.red-bean.com/fr/1.8/svn.advanced.confarea.html]. La section *Configuration automatique des propriétés* [http://svnbook.red-bean.com/fr/1.8/svn.advanced.props.html#svn.advanced.props.auto] est d'un intérêt particulier, et c'est ce qui est configuré ici. Remarquez que Subversion peut lire les informations de configuration depuis plusieurs emplacements, et il faut savoir lequel a la priorité. Référez-vous à *Configuration via la base de registre Windows* [http://svnbook.red-bean.com/fr/1.8/svn.advanced.confarea.html#svn.advanced.confarea.windows-registry] pour en savoir plus.

Appliquer les modifications locales à svn:externals lors d'une mise à jour
: Cette option indique à TortoiseSVN de toujours appliquer les modifications locales à la propriété `svn:externals` lors de la mise à jour de la copie de travail.

4.31.1.1. Paramètres du menu contextuel

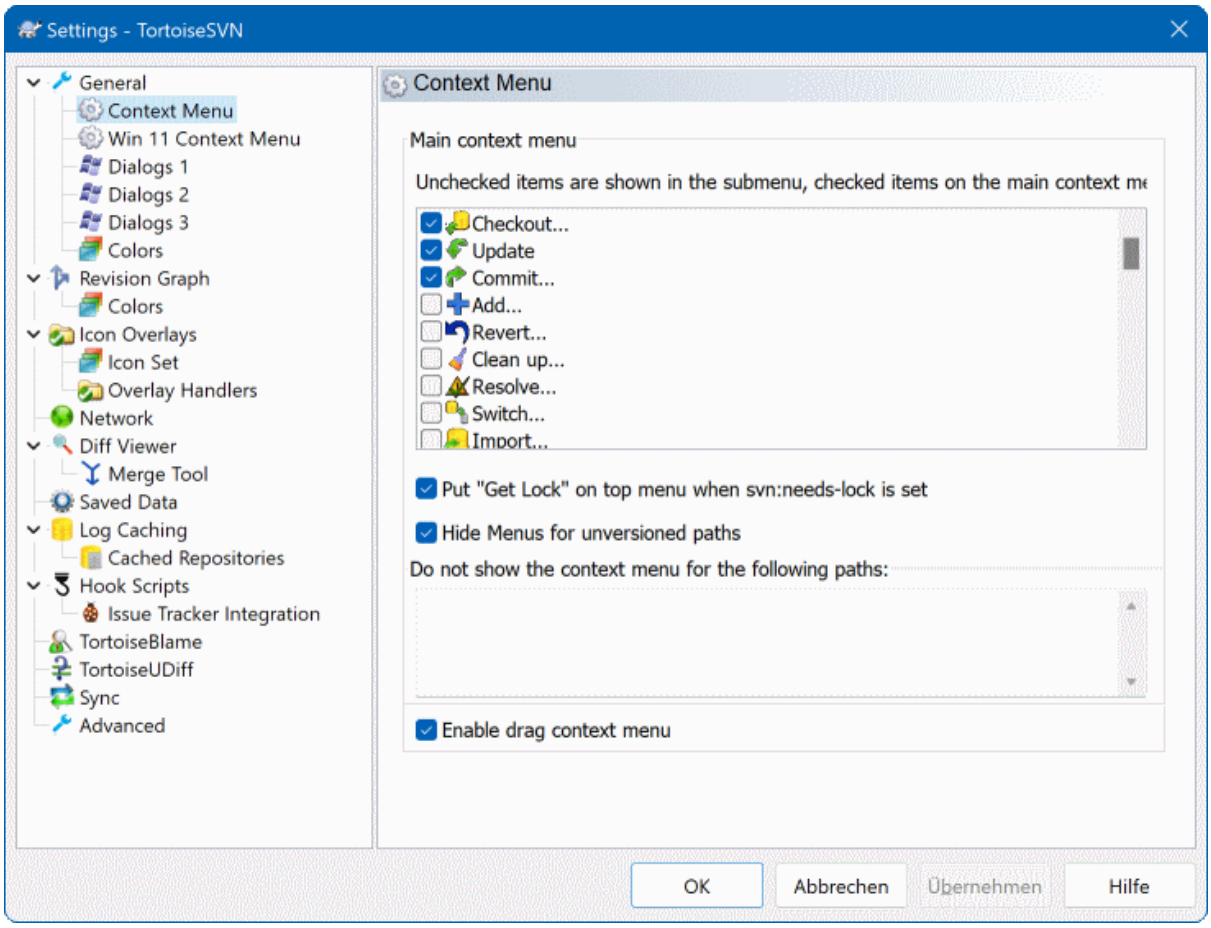


Figure 4.73. La boîte de dialogue Configuration, page Menu contextuel

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.

La plupart du temps, vous n'aurez pas besoin du menu contextuel TortoiseSVN, à part pour les dossiers qui sont versionnés dans Subversion. Pour les dossiers non versionnés, vous n'avez vraiment besoin du menu contextuel que quand vous voulez faire une extraction. Si vous cochez l'option `Cacher les menus pour les dossiers non versionnés`, TortoiseSVN n'ajoutera pas ses commandes au menu contextuel pour les dossiers non versionnés. Mais les commandes sont ajoutées pour tous les éléments et chemins dans un dossier versionné. Et vous pouvez récupérer les commandes pour les dossiers non versionnés en gardant la touche **Maj** appuyée quand vous faites apparaître le menu contextuel.

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.

4.31.1.2. Réglages des boîtes de dialogue TortoiseSVN 1

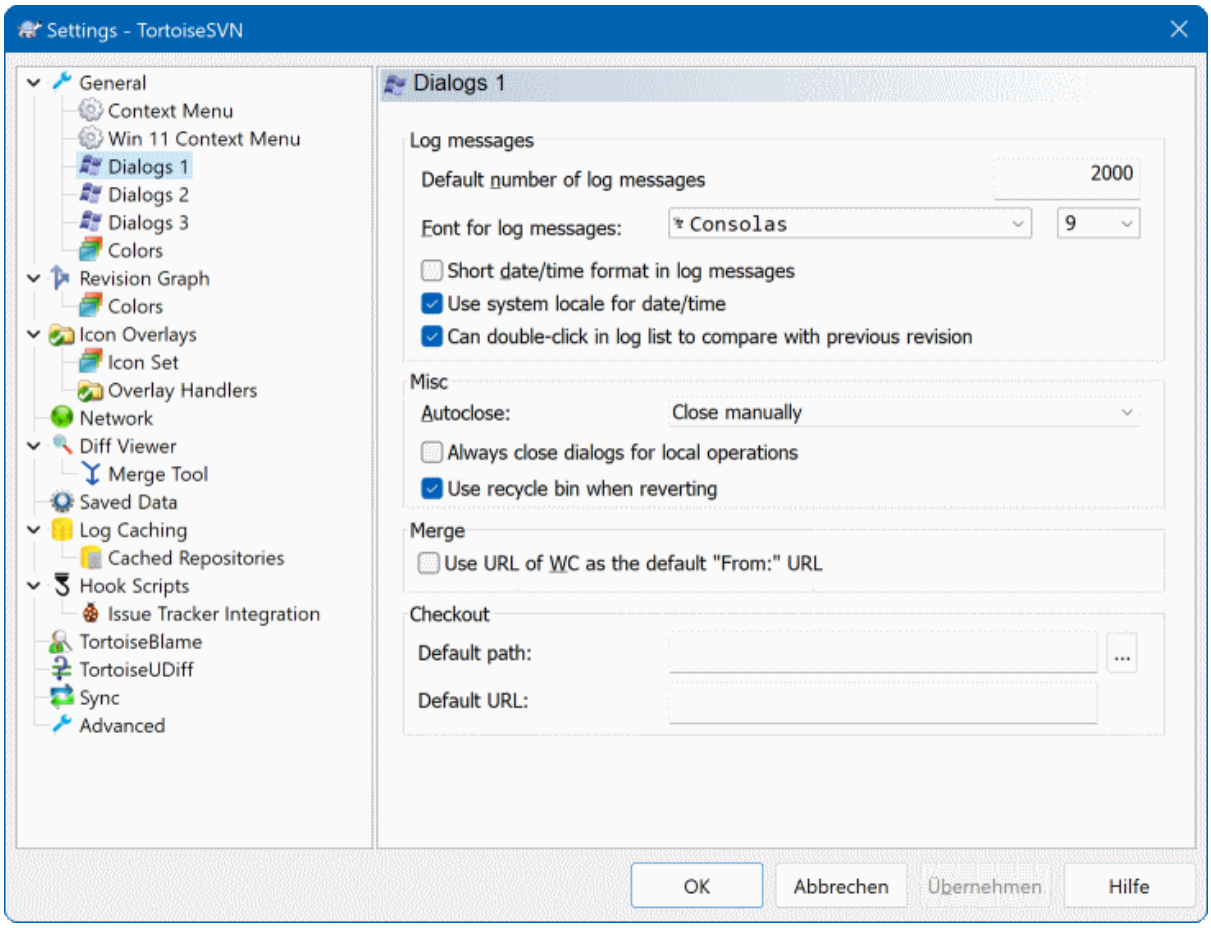


Figure 4.74. La boîte de dialogue Configuration, page Boîtes de dialogue 1

Cette boîte de dialogue vous permet de configurer certaines des boîtes de dialogue de TortoiseSVN de la façon que vous préférez.

Nombre par défaut de commentaires
: Limite le nombre de commentaires que TortoiseSVN va chercher quand vous choisissez TortoiseSVN → Voir le journal pour la première fois. Utile avec des connexions serveur lentes. Vous pouvez toujours utiliser Afficher tout ou 100 suivants pour obtenir plus de 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.

Double-cliquer dans la liste du journal pour comparer avec la révision précédente
: Si vous devez fréquemment comparer des révisions dans le panneau du haut de la boîte de dialogue du journal, vous pouvez utiliser cette option pour autoriser cette action sur un double clic. Elle n'est pas activée par défaut parce que récupérer le diff est souvent un processus long, et les gens préfèrent souvent éviter de devoir attendre après un double clic accidentel.

Fermeture automatique
: 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 dépôt 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 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.

Toujours fermer les boîtes de dialogue pour les opérations locales.
: Les opérations locales, comme l'ajout de fichiers ou l'annulation des modifications n'ont pas besoin de communiquer avec le dépôt et se déroulent rapidement, de sorte que la boîte de dialogue de progression n'est souvent que de peu d'intérêt. Sélectionnez cette option si vous voulez que la boîte de dialogue de progression se ferme automatiquement après ces opérations, sauf s'il y a des erreurs.

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.

4.31.1.3. Réglages des boîtes de dialogues TortoiseSVN 2

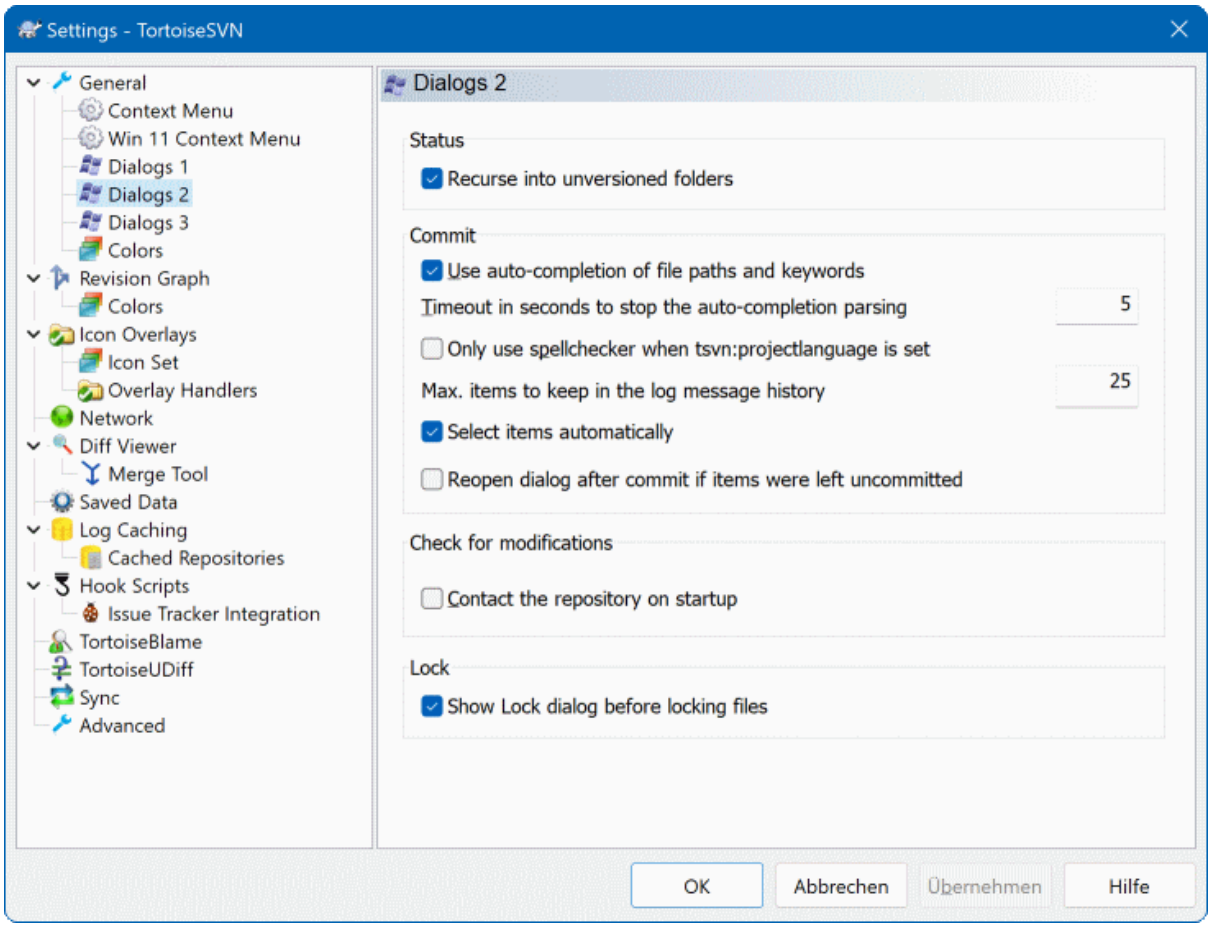


Figure 4.75. La boîte de dialogue Configuration, page Boîtes de dialogue 2

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.

  Dans la fenêtre Vérifier les modifications vous pouvez choisir de voir les éléments ignorés. Si cette case est cochée alors dès qu'un dossier ignoré est trouvé, tous les éléments enfants seront également affichés.

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
: L'analyseur syntaxique d'autocomplétion peut être assez lent s'il y a beaucoup de gros fichiers à vérifier. Ce délai empêche la boîte de dialogue de livraison de se bloquer trop longtemps. Si vous manquez d'informations d'autocomplétion importantes, vous pouvez étendre le délai.

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
: Lorsque vous tapez un commentaire dans la boîte de dialogue de livraison, TortoiseSVN le stocke pour une éventuelle réutilisation plus tard. Par défaut, il conserve les 25 derniers commentaires pour chaque dépôt, mais vous pouvez personnaliser ce nombre ici. Si vous avez beaucoup de dépôts différents, vous devriez le réduire pour éviter de remplir votre registre.

  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.

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.

Rouvre la boîte de dialogue de dépôt après un dépôt réussi si il y avait des éléments non déposés.
: This reopens the commit dialog automatically at the same directory after a successful commit. The dialog is reopened only if there still are items left to commit.

Contacter le dépôt au démarrage
: La boîte de dialogue Vérifier les modifications vérifie la copie de travail par défaut et entre seulement en contact avec le dépôt quand vous cliquez sur Vérifier le dépôt. Si vous voulez toujours vérifier le dépôt, 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.

  Si votre projet a la propriété `tsvn:lockmsgminsize`, vous verrez tout de même la fenêtre de verrouillage parce que le projet *a besoin* de messages de verrouillage.

4.31.1.4. Boîte de Dialogue de Paramètres de TortoiseSVN 3

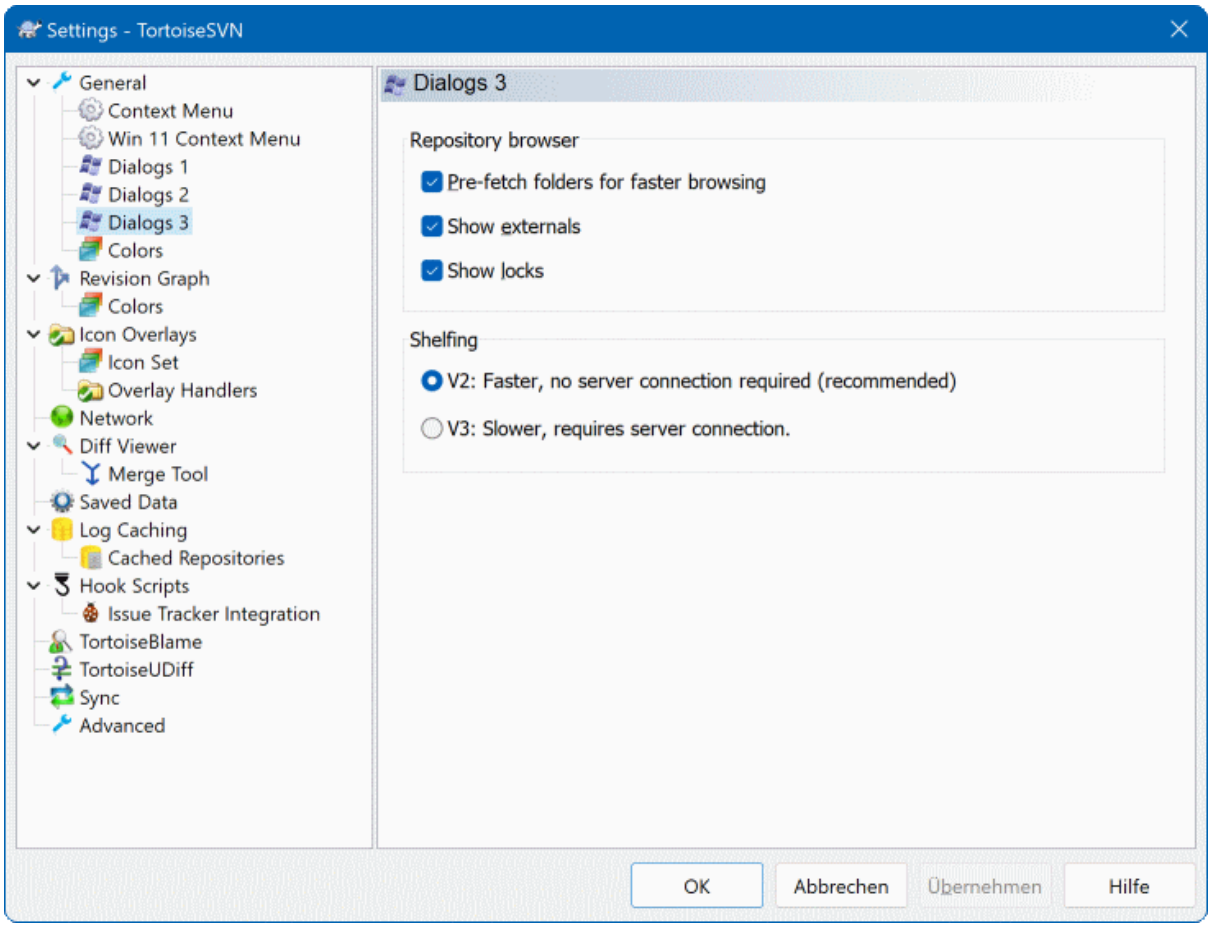


Figure 4.76. La boîte de dialogue Configuration, page Boîtes de dialogue 3

Configuration pour le navigateur du dépôt :

Pré-charger les dossiers pour une navigation plus rapide
: If this box is checked (default state), then the repository browser fetches information about shown folders in the background. That way as soon as you browse into one of those folders, the information is already available.

  Some servers however can't handle the multiple requests this causes or when not configured correctly treat so many requests as something bad and start blocking them. In this case you can disable the pre-fetching here.

Afficher les références externes
: If this box is checked (default state), then the repository browser shows files and folders that are included with the `svn:externals` property as normal files and folders, but with an overlay icon to mark them as from an external source.

  As with the pre-fetch feature explained above, this too can put too much stress on weak servers. In this case you can disable this feature here.

There are two versions of shelfing implemented in SVN. Here you can select which version you want to use. Note that changing this setting might require an OS restart to take effect.

V2
: Cette version est beaucoup plus rapide que V3, et son usage est recommandé.

  However, the speed comes at a prize: V2 does not handle directory changes, and can't handle copies and moves of files.

V3
: this is the latest version of the shelfing feature. It can handle changes to directories as well as file moves/copies.

  However, V3 is much slower than V2 and can be unusably slow for big repositories or if you have a slow connection to the repository.

4.31.1.5. Configuration des couleurs de TortoiseSVN

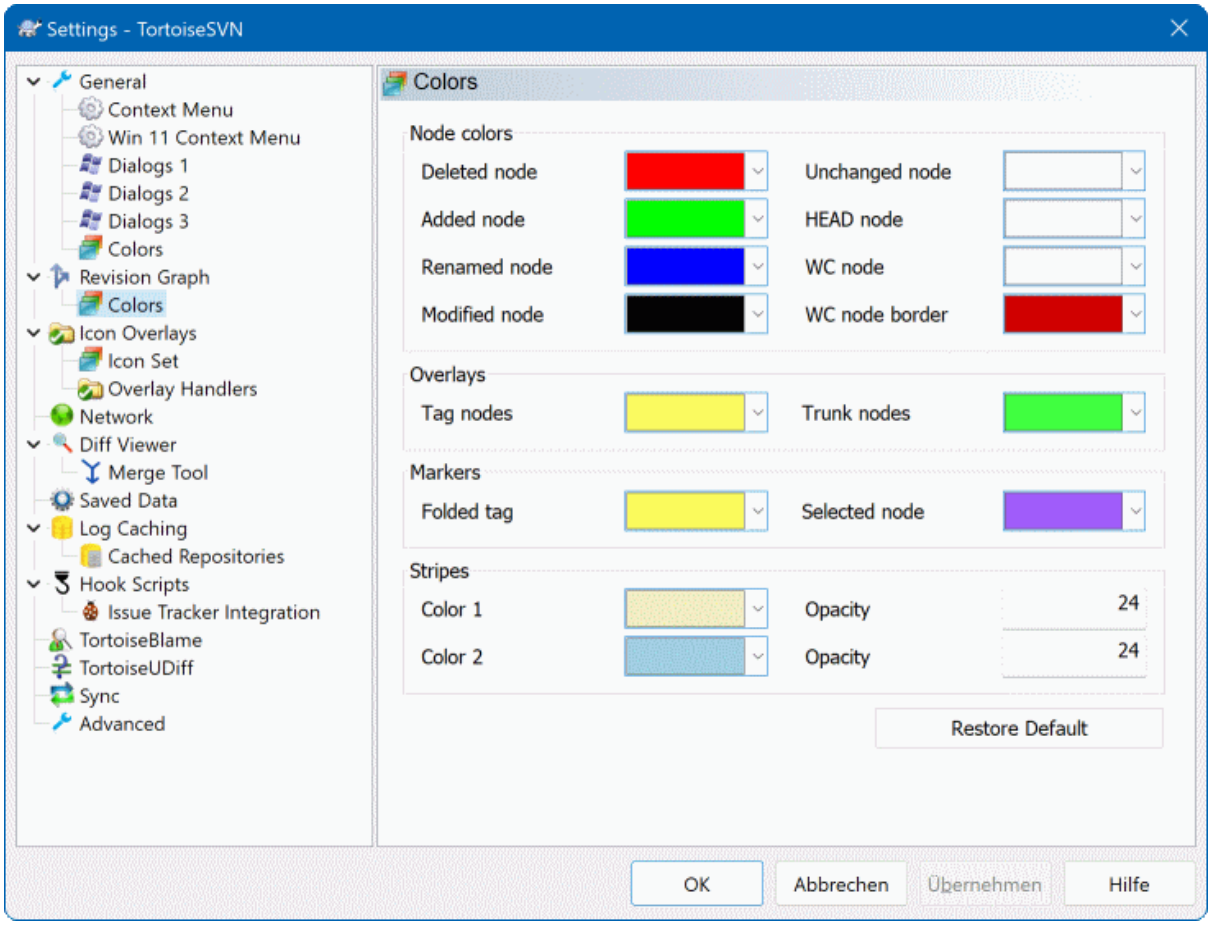


Figure 4.77. La boîte de dialogue Configuration, page Couleurs

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 dépôt.

Manquant / supprimé / remplacé
: Éléments supprimés du dépôt, 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 dépôt fusionnés avec succès dans la CdT sans créer de conflits.

Modifié / copié
: Ajoutés avec l'historique, ou chemins copiés dans le dépôt. 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 dépôt.

Noeud ajouté
: Un élément ajouté au dépôt, par un ajout, une copie ou un déplacement.

Noeud renommé
: Un élément qui a été renommé dans le dépôt.

Noeud remplacé
: L'élément original a été supprimé et un nouvel élément avec le même nom le remplace.

Correspondance du filtre
: Lors de l'utilisation de filtrage dans la boîte de dialogue du journal, les termes de recherche sont mis en évidence dans les résultats en utilisant cette couleur.

autres paramètres :

4.31.2. Options du Graphe des Révisions

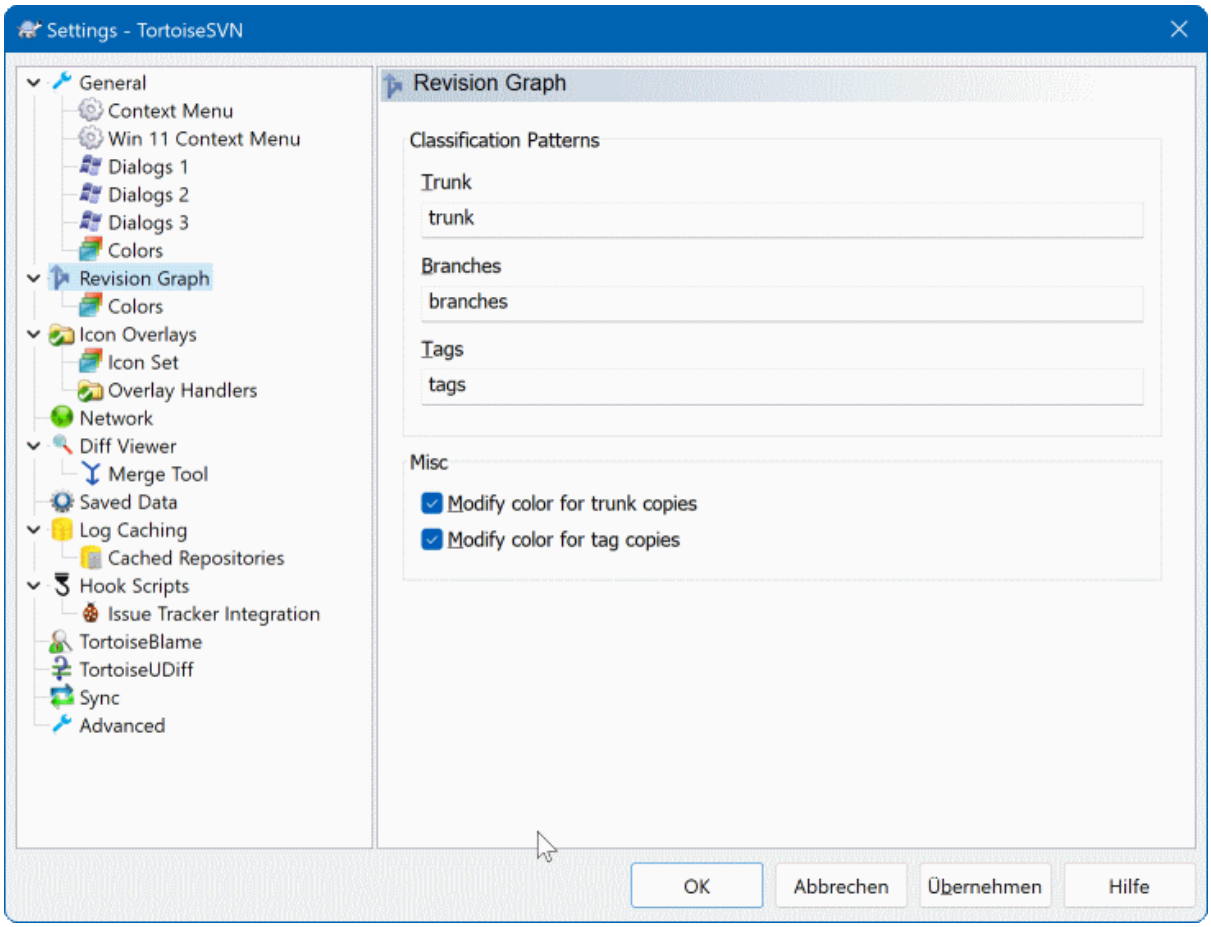


Figure 4.78. La boîte de dialogue Configuration, page graphique de révision

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.

Commit tag detection

Please note that these patterns are also used to detect commits to a tag, not just for the revision graph.

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.

4.31.2.1. Couleurs du Graphes de Révision

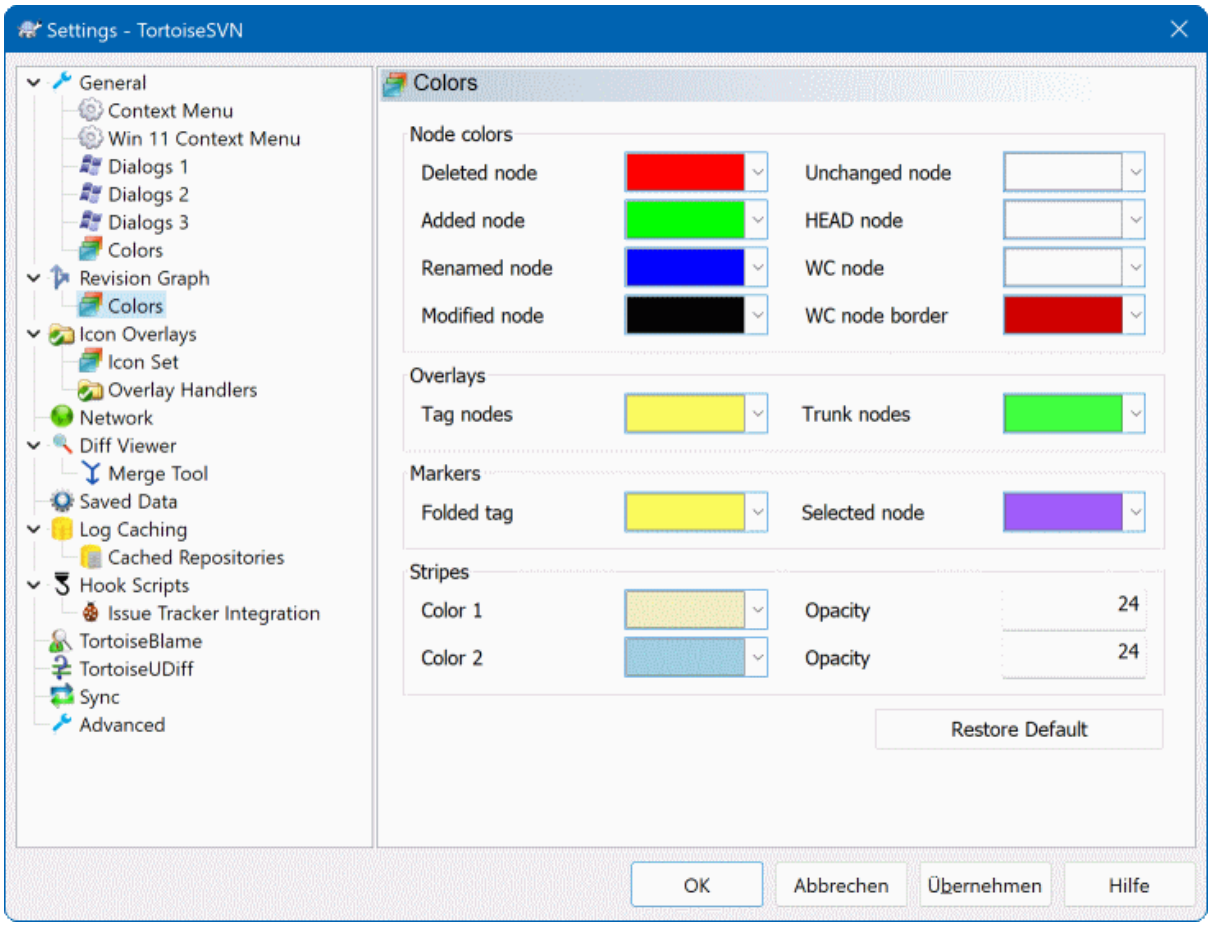


Figure 4.79. La boîte de dialogue Configuration, page des couleur du graphe de révision

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 dépôt

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.

4.31.3. Configuration du recouvrement d'icônes

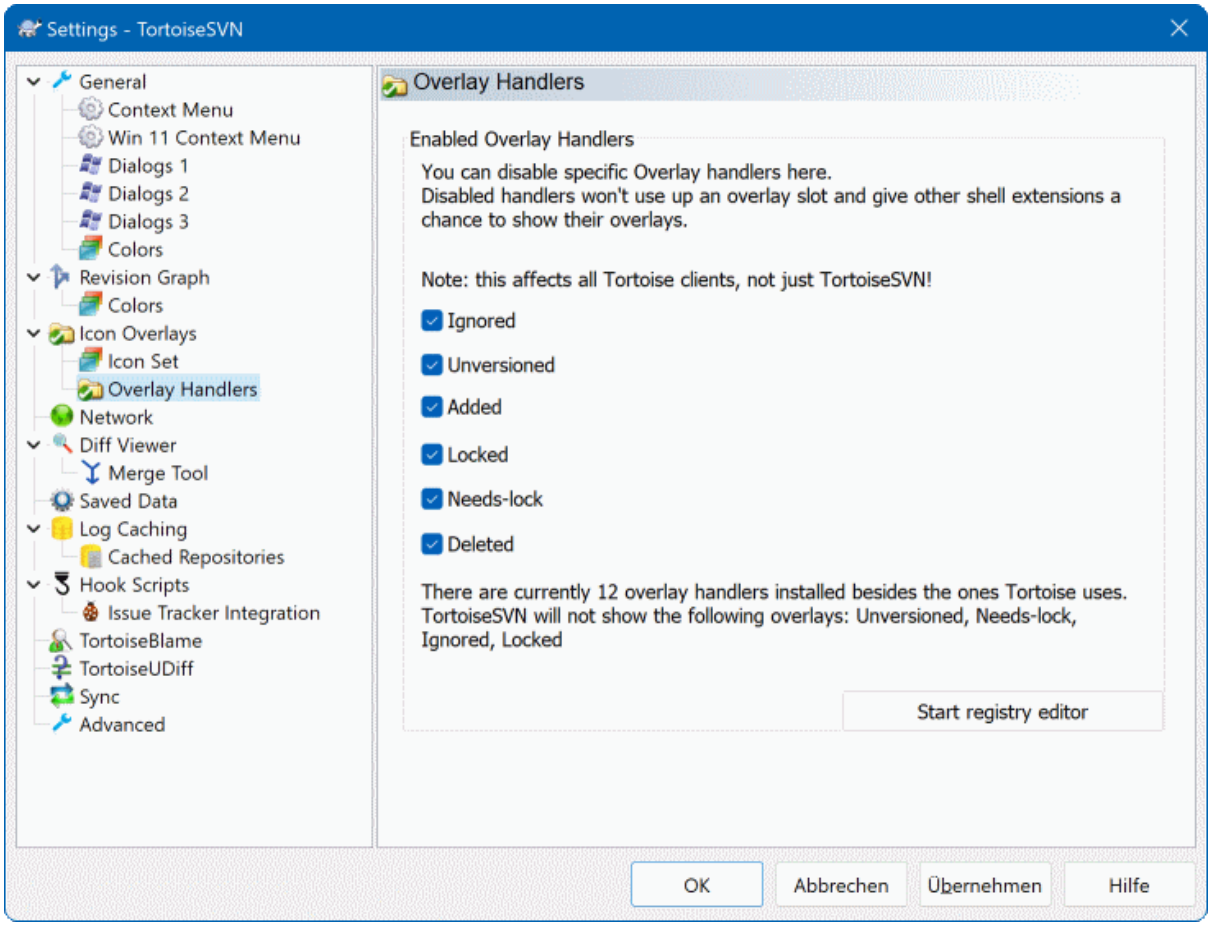


Figure 4.80. La Boîte de Dialogue Configuration, Page des Icônes de Recouvrement

Cette page vous permet de choisir à quels éléments TortoiseSVN doit associer des icônes de recouvrement

Puisque cela prend du temps pour récupérer le statut d'une copie de travail, TortoiseSVN utilise un cache pour stocker le statut pour que l'explorateur ne soit pas trop bloqué en affichant les recouvrements. Vous pouvez choisir quel type de cache TortoiseSVN devrait utiliser selon votre système et la taille de votre copie de travail ici :

Défaut
: Met en cache toute l'information de statut dans un processus séparé (`TSVNCache.exe`). Ce processus observe tous les disques pour les changements et va chercher à nouveau le statut si les fichiers à l'intérieur d'une copie de travail sont modifiés. Le processus s'exécute avec la priorité la plus faible possible pour que les autres programmes ne soient pas ralentis à cause de lui. Cela signifie aussi que l'information de statut n'est pas *en temps réel* mais cela peut prendre quelques secondes pour que les recouvrements changent.

  Avantage : les recouvrements montrent le statut récursivement, c'est-à-dire si un fichier est modifié dans les profondeurs d'une copie de travail, tous les dossiers jusqu'à la racine de la copie de travail montreront aussi le recouvrement modifié. Et puisque le processus peut envoyer des notifications au shell, les recouvrements sur l'arborescence gauche changent aussi.

  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.

Inconvénient : L'information de statut des fichiers et des dossiers n'est pas affichée dans l'Explorateur. Pour voir si vos copies de travail sont modifiées, vous devez utiliser la boîte de dialogue « Vérifier les modifications ».

Par défaut, les icônes de recouvrement et les menus contextuels apparaîtront dans toutes les boîtes de dialogue Ouvrir/Enregistrer comme dans l'explorateur Windows. Si vous voulez qu'elles n'apparaissent *que* dans l'explorateur Windows, cochez la case Voir les recouvrements et le menu contextuel seulement dans l'explorateur.

You can force the status cache to *None* for elevated processes by checking the Disable status cache for elevated processes box. This is useful if you want to prevent another `TSVNCache.exe` process getting created with elevated privileges.

Vous pouvez aussi choisir de marquer les répertoires comme modifiés s'ils contiennent des éléments non versionnés. Cela peut être utile pour vous rappeler que vous avez créé de nouveaux fichiers qui ne sont pas encore versionnés. Cette option n'est disponible que lorsque vous utilisez l'option de statut de cache *par défaut* (voir ci-dessous)

If you have files in the `ignore-on-commit` changelist, you can chose to make those files not propagate their status to the parent folder. That way if only files in that changelist are modified, the parent folder still shows the unmodified overlay icon.

Le groupe suivant vous permet de choisir les quels éléments de stockage montreront les recouvrements. Par défaut, seuls les disques durs sont sélectionnés. Vous pouvez même désactiver tous les recouvrements d'icône, mais qu'y-a-t-il d'amusant à cela ?

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, ou si vous voulez seulement que TortoiseSVN regarde dans des dossiers spécifiques.

Tout chemin que vous spécifiez ici est supposé s'appliquer de manière récursive, de sorte qu'aucun des dossiers enfant ne montrera le recouvrement quoi qu'il arrive. Si vous voulez exclure *seulement* le dossier nommé, ajoutez `?` après le 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.

Les utilisateurs se demandent parfois comment ces trois paramètres interagissent. Pour tout chemin donné, il y a vérification des listes d'inclusion et d'exclusion, en parcourant vers le haut la structure du répertoire jusqu'à ce qu'une correspondance soit trouvée. Lorsque la première correspondance est trouvée, il y a vérification selon la règle d'inclusion ou d'exclusion. S'il y a un conflit, une spécification unique de répertoire a priorité sur une spécification récursive, puis l'intégration l'emporte sur l'exclusion.

An example will help here:

```
Exclude:
C:
C:\develop\?
C:\develop\tsvn\obj
C:\develop\tsvn\bin

Include:
C:\develop
```

These settings disable icon overlays for the C: drive, except for `c:\develop`. All projects below that directory will show overlays, except the `c:\develop` folder itself, which is specifically ignored. The high-churn binary folders are also 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.

Un moyen simple de contourner cela est d'empêcher le chemin d'origine de montrer les recouvrements, qui se présentent alors sur le chemin `subst` à la place.

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 a folder contains a working copy. The Show excluded root folders as 'normal' checkbox allows you to do this. With this option, working copy root 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.

Une exception particulière à cela est que les lecteurs `A:` et `B:` ne sont jamais pris en compte dans l'option Montrer les dossiers exclus comme 'normaux'. C'est parce que Windows est obligé de chercher sur le disque, ce qui peut entraîner un retard de plusieurs secondes lors du démarrage d'Explorer, même si votre PC possède un lecteur de disquette.

4.31.3.1. Sélection du jeu d'icônes

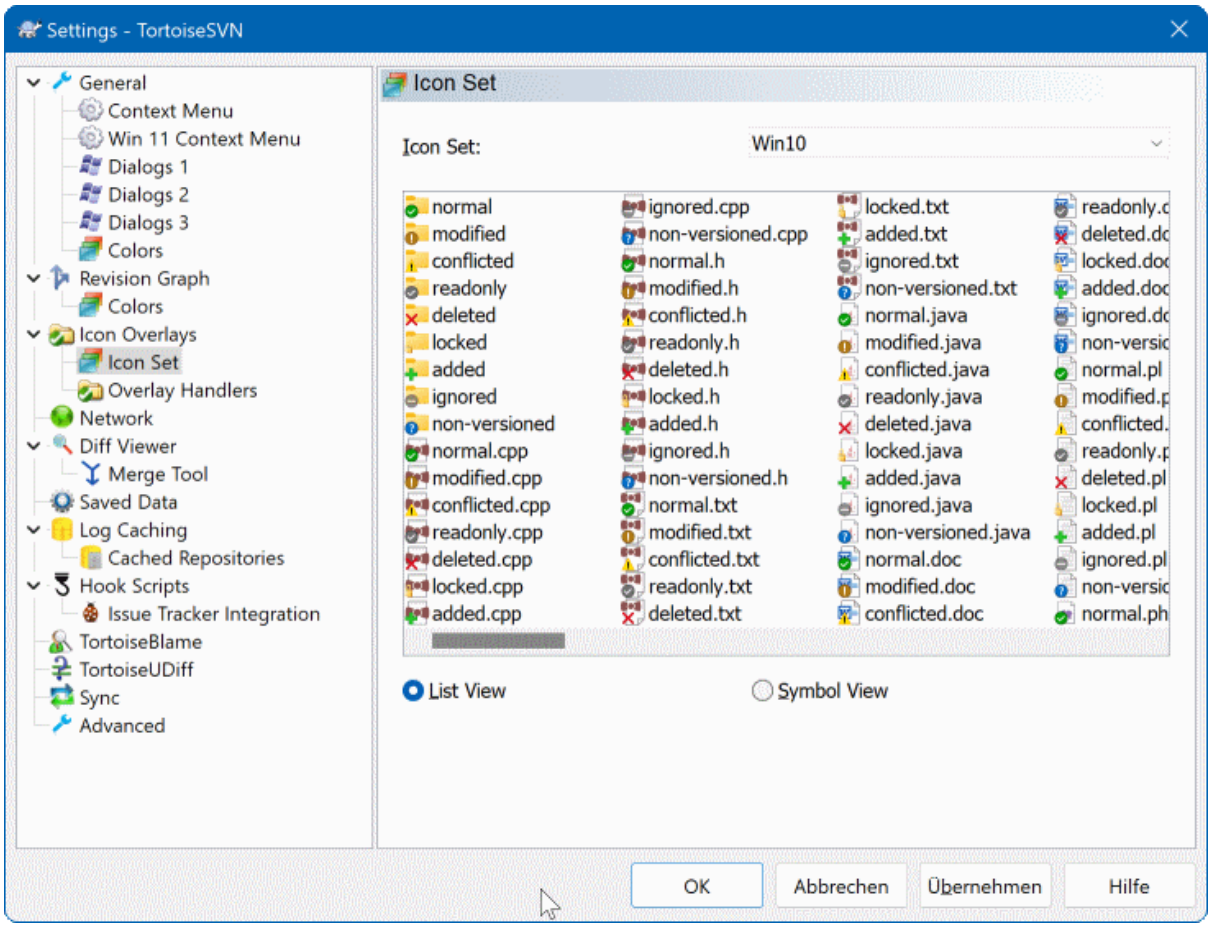


Figure 4.81. La boîte de dialogue Configuration, page Ensemble d'icônes

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.

4.31.3.2. Gestionnaires de recouvrement autorisés

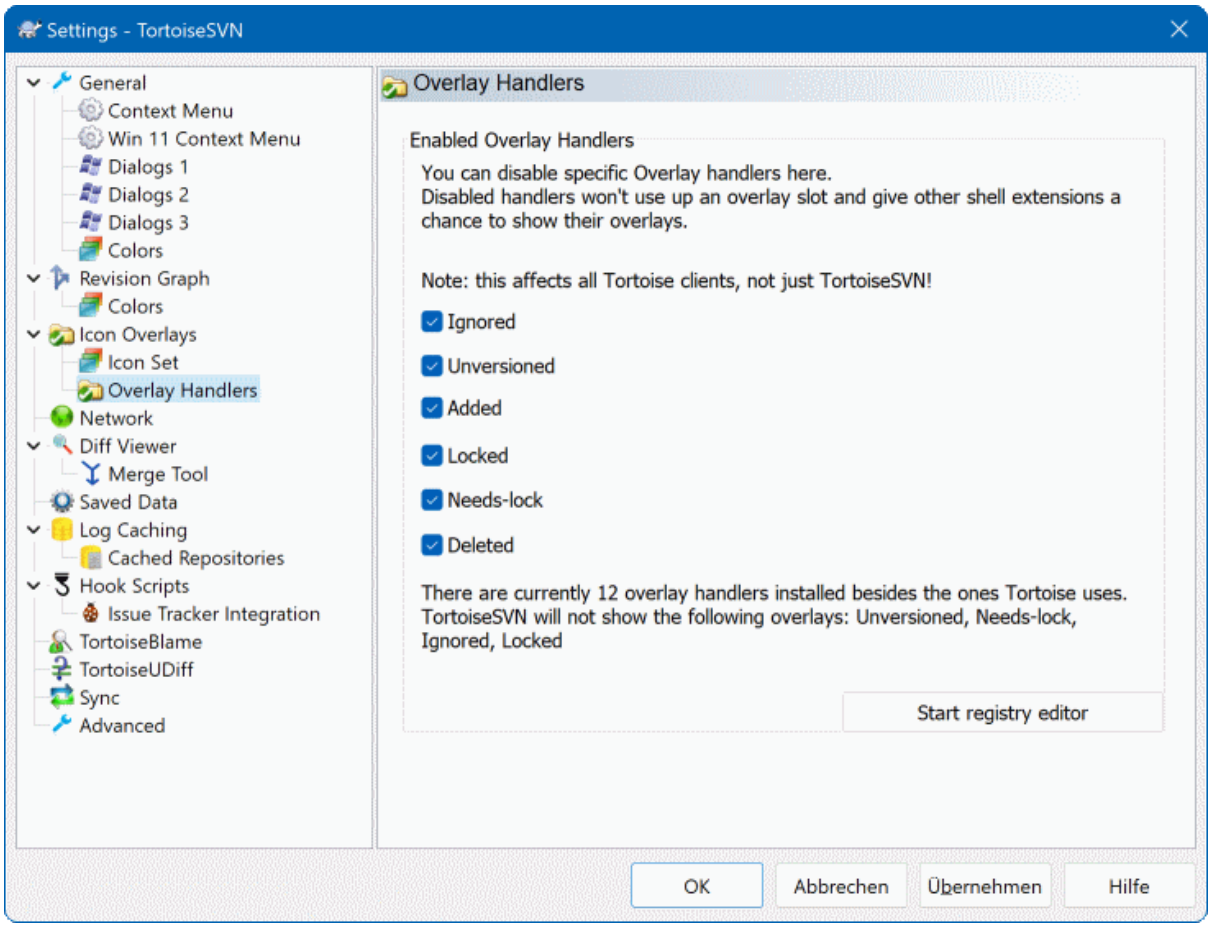


Figure 4.82. La boîte de dialogue Configuration, page Jeu d'icônes

Because the number of overlays available is severely restricted, you can choose to disable some handlers to ensure that the ones you want will be loaded. Because TortoiseSVN uses the common TortoiseOverlays component which is shared with other Tortoise clients (e.g. TortoiseCVS, TortoiseHg) this setting will affect those clients too.

4.31.4. Configuration du réseau

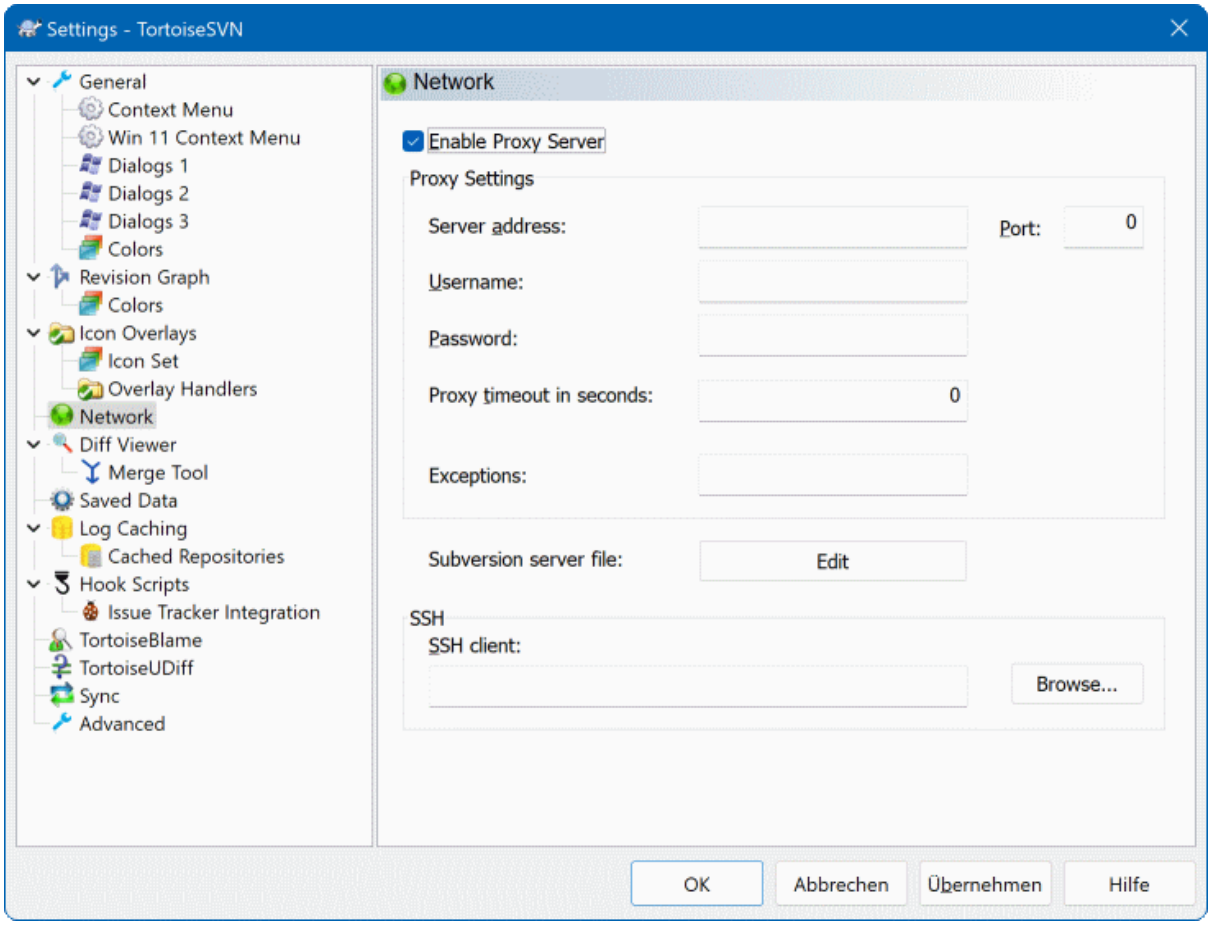


Figure 4.83. La boîte de dialogue Configuration, page Réseau

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/1.8/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 dépôt 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* [https://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* [https://tortoisesvn.net/ssh_howto.html].

4.31.5. Réglages des programmes externes

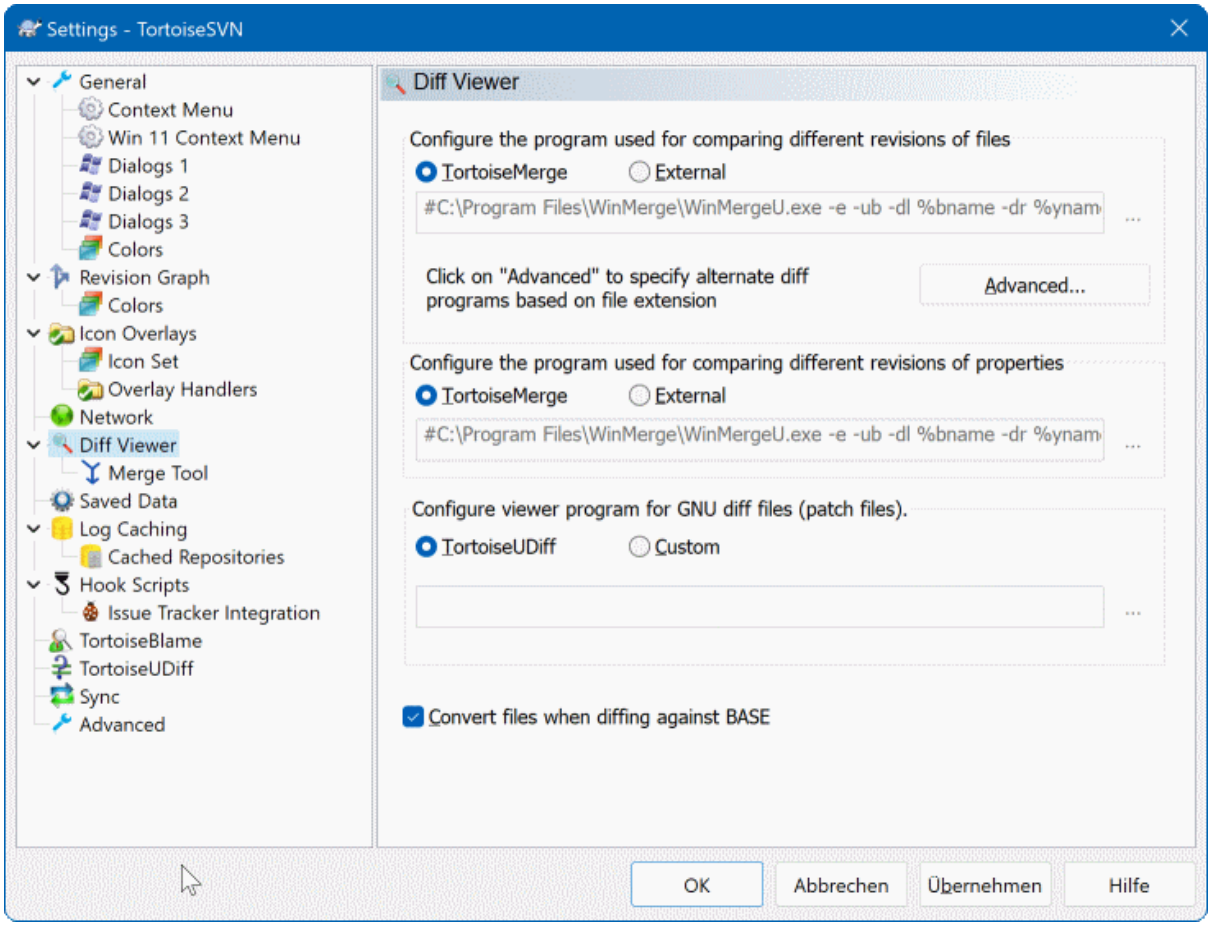


Figure 4.84. La boîte de dialogue Configuration, page Visualisateur de différence

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 4.11.6, « Outils de différenciation/fusion externes » pour une liste de quelques programmes externes de différenciation/fusion que les gens utilisent avec TortoiseSVN.

4.31.5.1. 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

%nqbname
Le titre de la fenêtre pour le fichier de base, sans les guillemets

%mine
Votre propre fichier, avec vos changements

%yname
Le titre de la fenêtre pour votre fichier

%nqyname
Le titre de la fenêtre pour votre fichier, sans les guillemets

%burl
L'URL du fichier original, si disponible

%nqburl
L'URL du fichier d'origine, s'il est disponible, sans les guillemets

%yurl
L'URL du second fichier, si disponible

%nqyurl
L'URL du second fichier, s'il est disponible, sans les guillemets

%brev
La révision du fichier original, si disponible

%nqbrev
La révision du fichier d'origine, s'il est disponible, sans les guillemets

%yrev
La révision du second fichier, si disponible

%nqyrev
La révision du second fichier, s'il est disponible, sans les guillemets

%peg
La révision peg, si disponible

%nqpeg
The peg revision, if available, without quotes

%fname
Le nom du fichier. C'est une chaine vide si 2 fichiers différents sont comparés au lieu de deux états du même fichier.

%nqfname
Le nom du fichier, sans les guillemets

The window titles are not pure filenames. TortoiseSVN treats that as a name to display and creates the names accordingly. So e.g. if you're doing a diff from a file in revision 123 with a file in your working copy, the names will be `filename : revision 123` and `filename : working copy`.

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
```

or with UltraCompare:

```
C:\Path-To\uc.exe %base %mine -title1 %bname -title2 %yname
```

or with DiffMerge:

```
C:\Path-To\DiffMerge.exe -nosplash -t1=%bname -t2=%yname %base %mine
```

Si vous utilisez la propriété `svn:keywords` pour développer des mots-clés et en particulier la `révision` d'un fichier, alors il peut y avoir des différences entre les fichiers qui est purement due à la valeur actuelle du mot-clé. De même si vous utilisez `svn:eol-style = native` le fichier BASE aura des fins de ligne `LF` tandis que votre fichier aura des fins de ligne `CR-LF`. TortoiseSVN cachera normalement ces différences automatiquement en analysant syntaxiquement d'abord le fichier BASE pour étendre les mots-clés et les fins de ligne avant de faire l'opération de comparaison. Cependant, cela peut prendre du temps avec de gros fichiers. Si Convertir les fichiers lors d'une comparaison avec la BASE est décoché alors TortoiseSVN sautera le prétraitement des fichiers.

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.

Si vous avez configuré un outil de comparaison alternatif, vous pouvez accéder à TortoiseMerge *et* à l'outil tiers à partir des menus contextuels. Menu contextuel → Voir les différences utilise l'outil de comparaison primaire, et **Shift** + Menu contextuel → Voir les différences utilise l'outil de comparaison secondaire.

At the bottom of the dialog you can configure a viewer program for unified-diff files (patch files). No parameters are required. The Default setting is to use TortoiseUDiff which is installed alongside TortoiseSVN, and colour-codes the added and removed lines.

Etant que le Diff Unifié est juste un format texte, vous pouvez utiliser votr éditeur texte favori si vous préférez.

4.31.5.2. 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

%nqbname
: Le titre de la fenêtre pour le fichier de base, sans les guillemets

%mine
: votre propre fichier, avec vos changements

%yname
: Le titre de la fenêtre pour votre fichier

%nqyname
: Le titre de la fenêtre pour votre fichier, sans les guillemets

%theirs
: le fichier tel qu'il est dans le dépôt

%tname
: Le titre de la fenêtre pour le fichier dans le dépôt

%nqtname
: Le titre de la fenêtre pour le fichier dans le dépôt, sans les guillemets

%merged
: le fichier en conflit, le résultat de l'opération de fusion

%mname
: Le titre de la fenêtre pour le fichier fusionné

%nqmname
: Le titre de la fenêtre pour le fichier fusionné, sans les guillemets

%fname
: Le nom du fichier en conflit

%nqfname
: Le nom du fichier en conflit, sans les guillemets

For example, with Perforce Merge:

```
C:\Path-To\P4Merge.exe %base %theirs %mine %merged
```

or with KDiff3:

```
C:\Path-To\kdiff3.exe %base %mine %theirs -o %merged
                      --L1 %bname --L2 %yname --L3 %tname
```

or with Araxis:

```
C:\Path-To\compare.exe /max /wait /3 /title1:%tname /title2:%bname
                       /title3:%yname %theirs %base %mine %merged /a2
```

or with WinMerge (2.8 or later):

```
C:\Path-To\WinMerge.exe %merged
```

or with DiffMerge:

```
C:\Path-To\DiffMerge.exe -caption=%mname -result=%merged -merge
      -nosplash -t1=%yname -t2=%bname -t3=%tname %mine %base %theirs
```

4.31.5.3. Réglages avancés de comparaison/fusion

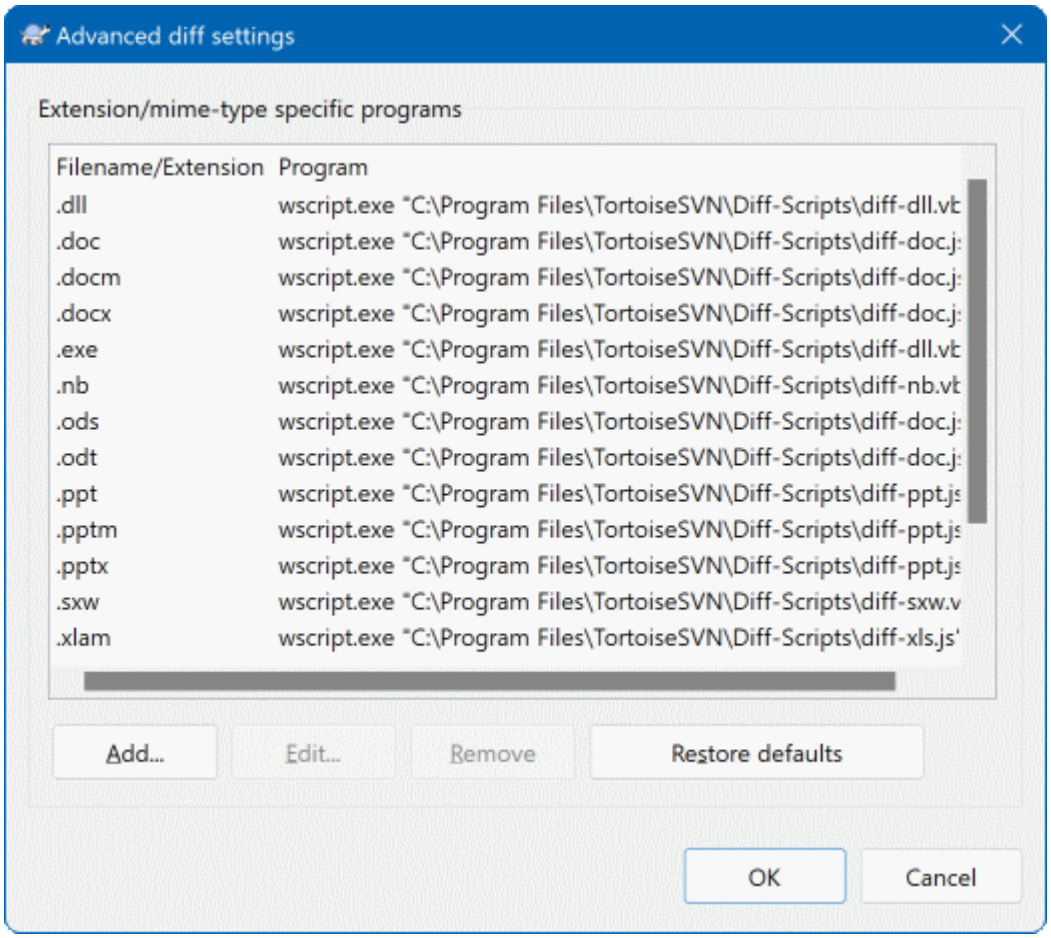


Figure 4.85. La boîte de dialogue Configuration, Boîte de dialogue Comparaison/fusion avancée

Dans les réglages avancés, vous pouvez définir un programme de comparaison et de fusion différent pour chaque extension de fichier. Par exemple, vous pourriez associer Photoshop comme programme de « comparaison » pour les fichiers `.jpg` :-) Vous pouvez aussi associer la propriété `svn:mime-type` à un programme de comparaison ou de fusion.

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`.

4.31.6. Configuration des données sauvegardées

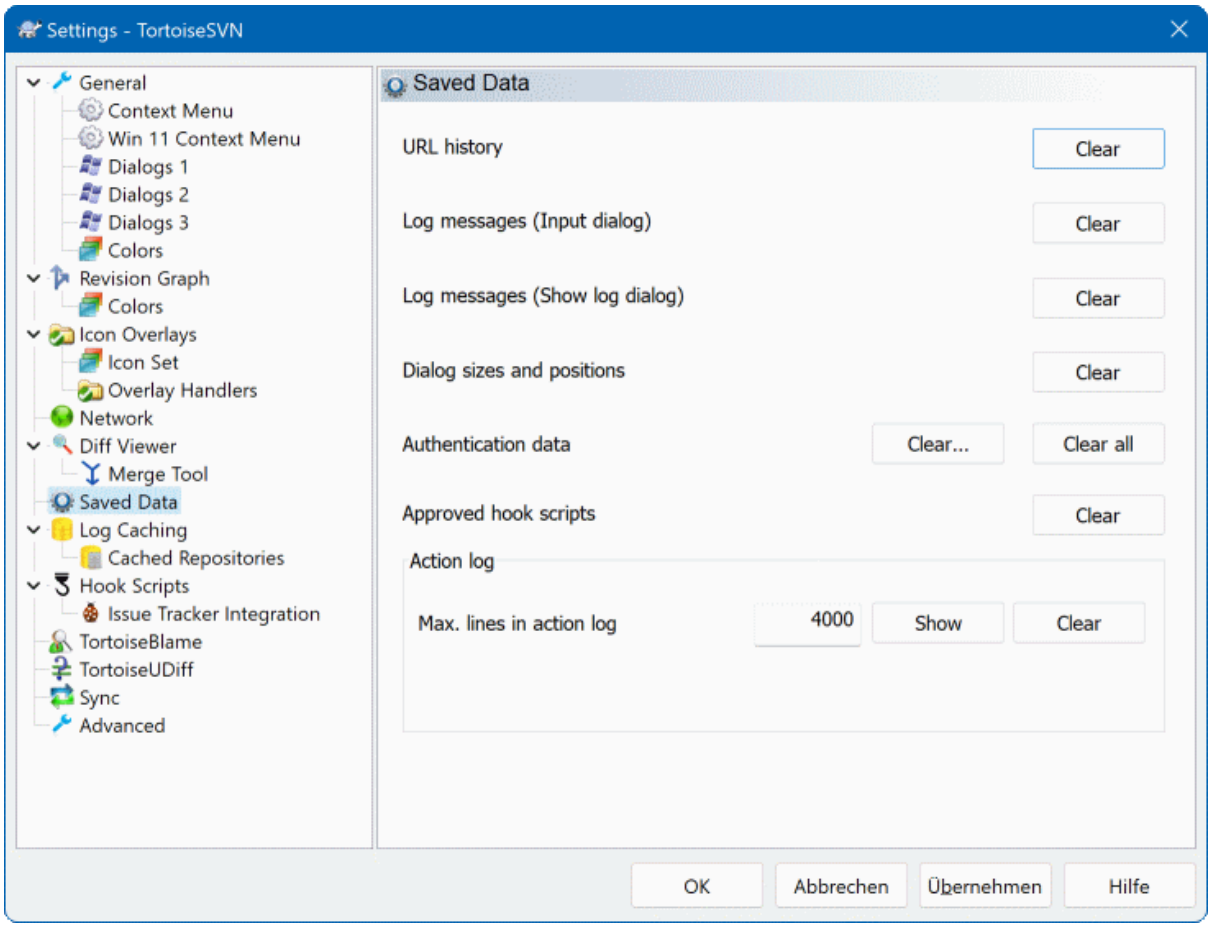


Figure 4.86. La boîte de dialogue Configuration, Page Données sauvegardées

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 dépôt, 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.

  Si vous souhaitez supprimer un seul élément de l'une des zones de liste déroulante vous pouvez le faire sur place. Il suffit de cliquer sur la flèche pour afficher le contenu de la liste déroulante, déplacer la souris sur l'élément que vous souhaitez supprimer et appuyer sur **Maj+Suppr**.

Messages de log (fenêtre d'édition)
: TortoiseSVN stocke les commentaires récents de livraison que vous saisissez. Ceux-ci sont stockés par dépôt, donc si vous avez accès à beaucoup de dépôts, cette liste peut devenir assez longue.

Messages de log (Montrer la fenêtre de log)
: TortoiseSVN met en cache les messages de log, récupérés par la boîte de dialogue Montrer les logs, pour faire gagner du temps pour la prochaine fois où vous afficherez le log. Si quelqu'un d'autre édite un message de log et que ce message est déjà mis en cache, vous ne verrez pas les changements jusqu'à ce que vous vidiez le cache. La mise en cache du log est activable via l'onglet Log Cache.

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 dépôt 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, use the Clear... instead of the Clear all button.

Log des Actions
: TortoiseSVN tient un journal de tout ce qui est écrit dans ses dialogues d'avancement. Cela peut être utile lorsque, par exemple, vous souhaitez vérifier ce qui s'est passé dans une commande de mise à jour récente.

  Le fichier journal est limité en longueur et quand il devient trop gros le contenu le plus ancien est supprimé. Par défaut 4000 lignes sont conservées, mais vous pouvez personnaliser ce nombre.

  Depuis cet endroit vous pouvez voir le contenu du fichier de commentaires, ainsi que le vider.

4.31.7. Mise en Cache des messages de log

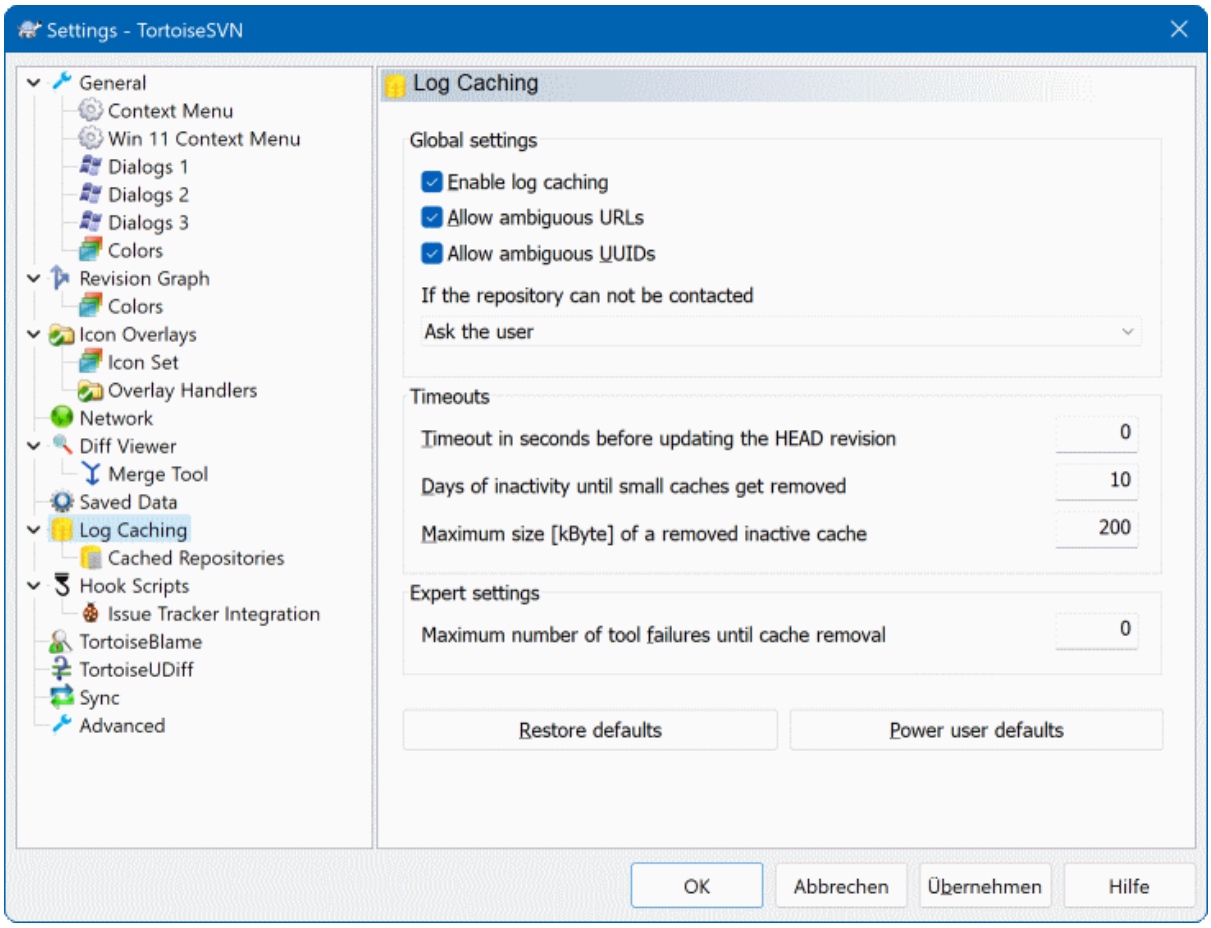


Figure 4.87. La boîte de dialogue de Configuration, Page de Mise en Cache des Logs

Cette boîte de dialogue vous permet de configurer la fonctionnalité de mise en cache du journal TortoiseSVN, qui conserve une copie locale des messages de log et des chemins modifiés afin d'éviter des téléchargements consommateurs de ressources à partir du serveur. Utiliser le cache journal peut accélérer considérablement l'ouverture de la boîte de dialogue journal et le graphique de révision. Un autre avantage est que les messages du journal sont toujours accessibles en mode hors connexion.

Activer la mise en cache des messages de log
: Permet la mise en cache chaque fois que des données log sont demandés. Si coché, les données seront récupérées à partir du cache lorsqu'elles seront disponibles, et les messages qui ne sont pas dans le cache seront récupérés sur le serveur et ajoutés au cache.

  Si le cache est désactivé, les données seront toujours récupérées directement à partir du serveur et non stockées localement.

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, unchecked it 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, unchecked it to improve performance.

Si le dépôt n'est pas disponible
: Si vous travaillez en mode hors connexion, ou si le serveur de dépôt est hors service, le cache du journal peut toujours être utilisé. Bien sûr, le cache peut ne pas être à jour, il existe alors des options pour vous permettre de choisir si cette fonction doit être utilisée.

  Lorsque les données du journal sont issues du cache sans avoir contacté le serveur, la boîte de dialogue utilisant ces messages indiquera l'état hors connexion dans sa barre de titre.

Temps d'attente maximum écoulé avant la mise à jour de la révision de tête (HEAD).
: Lorsque vous ouvrez la boîte de dialogue du journal, vous voudrez normalement contacter le serveur pour vérifier tous les nouveaux messages. Si le délai fixé ici est non-nul, le serveur ne sera contacté que lorsque le délai d'attente se sera écoulé depuis le dernier contact. Cela peut réduire les allers-retours avec le serveur si vous ouvrez fréquemment la boîte de dialogue et si le serveur est lent, mais les données présentées peuvent ne pas être complètement à jour. Si vous souhaitez utiliser cette fonctionnalité, nous vous suggérons d'utiliser une valeur de 300 (5 minutes) comme un compromis.

Nombre de jours d'inactivité avant que les petits caches soient supprimés
: Si vous naviguez parmi un grand nombre de dépôts, vous accumulerez beaucoup de caches journal. Si vous ne les utilisez pas beaucoup, le cache ne grossira pas trop et TortoiseSVN les purgera après un délai défini par défaut. Utilisez cet élément pour contrôler la purge du cache.

Taille maximum des caches inactifs à supprimer
: Les plus gros caches sont plus coûteux à rétablir, TortoiseSVN ne purge donc que les petits caches. Ajustez le seuil avec cette valeur.

Nombre maximum d'échecs de l'outil avant suppression du cache
: De temps en temps quelque chose se passe mal avec la mise en cache et cause un crash. Dans ce cas le cache est normalement supprimée automatiquement pour éviter une réapparition du problème. Si vous utilisez le paramètrage de nuit le moins stable vous pouvez choisir de garder le cache de toute façon.

4.31.7.1. Dépôts mis en mémoire cache

Sur cette page vous pouvez voir une liste des dépôts qui sont mis en cache localement, et l'espace utilisé pour le cache. Si vous sélectionnez l'un des dépôts, vous pourrez ensuite utiliser les boutons ci-dessous.

Cliquez sur le bouton Mise à jour pour rafraichir complètement la mémoire cache et remplir tous les trous. Pour les gros dépôts cette tâche peut être très coûteuse, mais utile si vous souhaitez vous déconnecter et avoir le meilleur cache possible.

Cliquez sur le bouton Exportater pour exporter l'intégralité du cache dans un ensemble de fichiers CSV. Cela peut être utile si vous voulez traiter les données du journal à l'aide d'un programme externe, surtout pour les développeurs.

Cliquez sur Supprimer pour supprimer toutes les données mises en cache pour les dépôts sélectionnés. Cela ne va pas désactiver le cache pour le dépôt, ainsi la prochaine fois que vous demanderez les données du journal, un nouveau cache sera créé.

4.31.7.2. Statistiques d'Utilisation du Cache

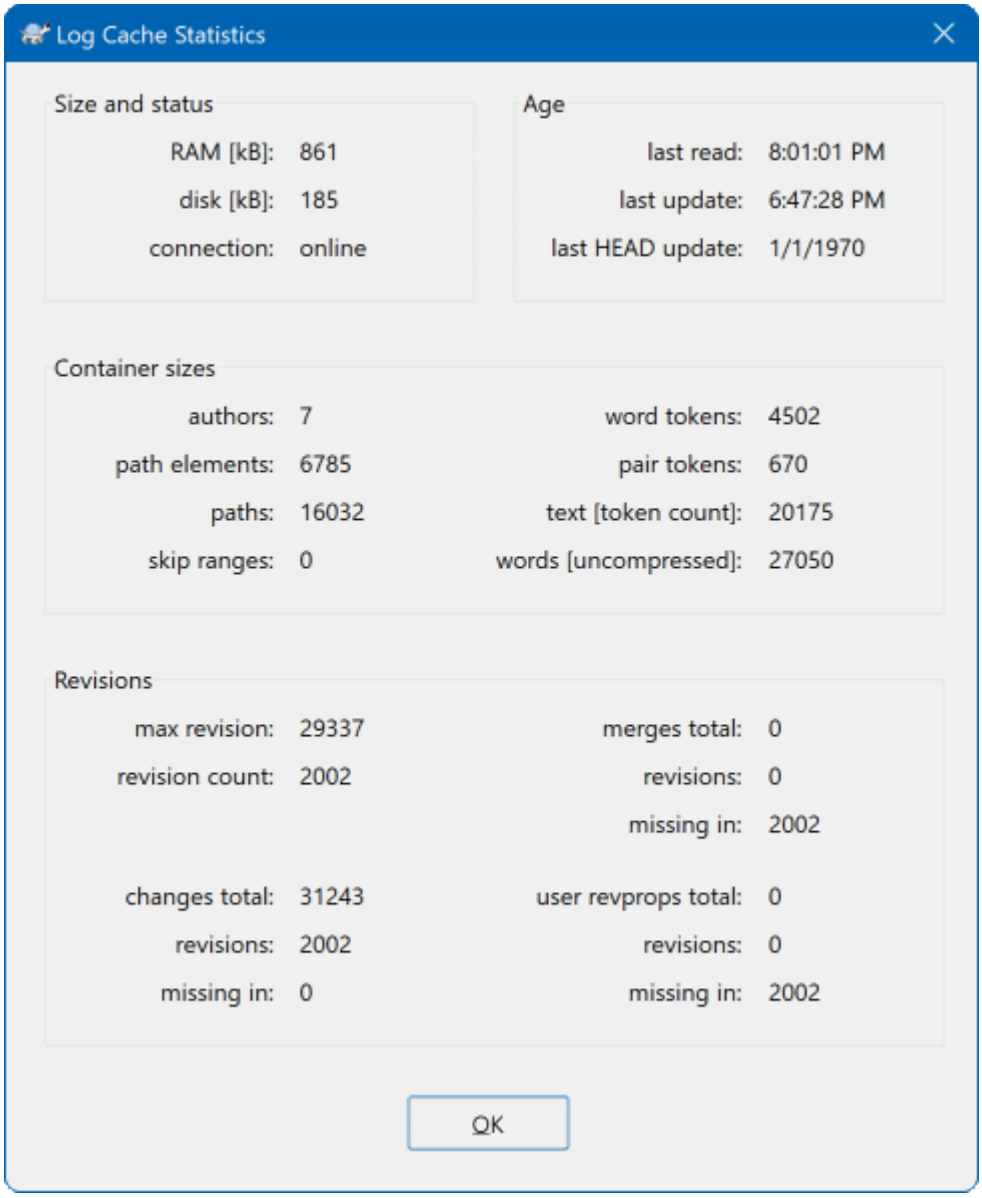


Figure 4.88. La Fenêtre de propriétés, Statistiques d'Utilisation de la Mémoire Cache

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
: La quantité d'espace disque utilisé pour le cache. Les données sont compressées, l'utilisation du disque est alors généralement assez modeste.

Connexion
: Montre si le dépôt 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
: Les plages de révisions que nous n'avons pas récupérées, tout simplement parce qu'elles n'ont pas été demandées. Ceci est une mesure du nombre de trous dans le 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.

4.31.8. Scripts hook côté client

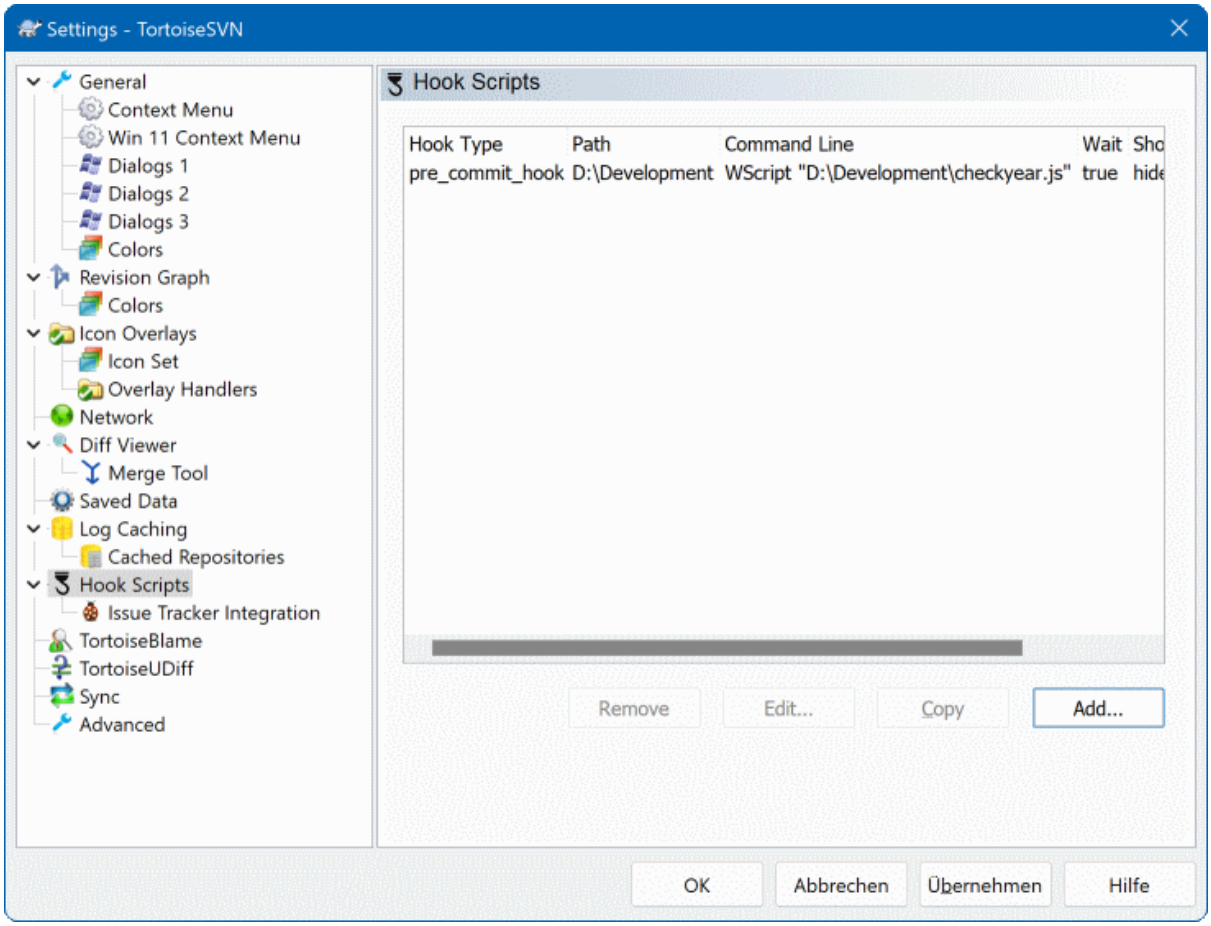


Figure 4.89. La boîte de dialogue Configuration, page Scripts hook

Cette boîte de dialogue vous permet de définir les scripts hook qui seront exécutés automatiquement lors de certaines actions de Subversion. Contrairement aux scripts de hook expliqué dans Section 3.3, « Scripts hook côté serveur », ces scripts sont exécutés localement sur le client.

Une application pour de tels (hooks) pourrait appeler un programme comme `SubWCRev.exe` pour mettre à jour les numéros de version après une révision, et peut-être pour déclencher une reconstruction.

Note that you can also specify such hook scripts using special properties on your working copy. See the section Section 4.18.2, « Propriétés de projet TortoiseSVN » for details.

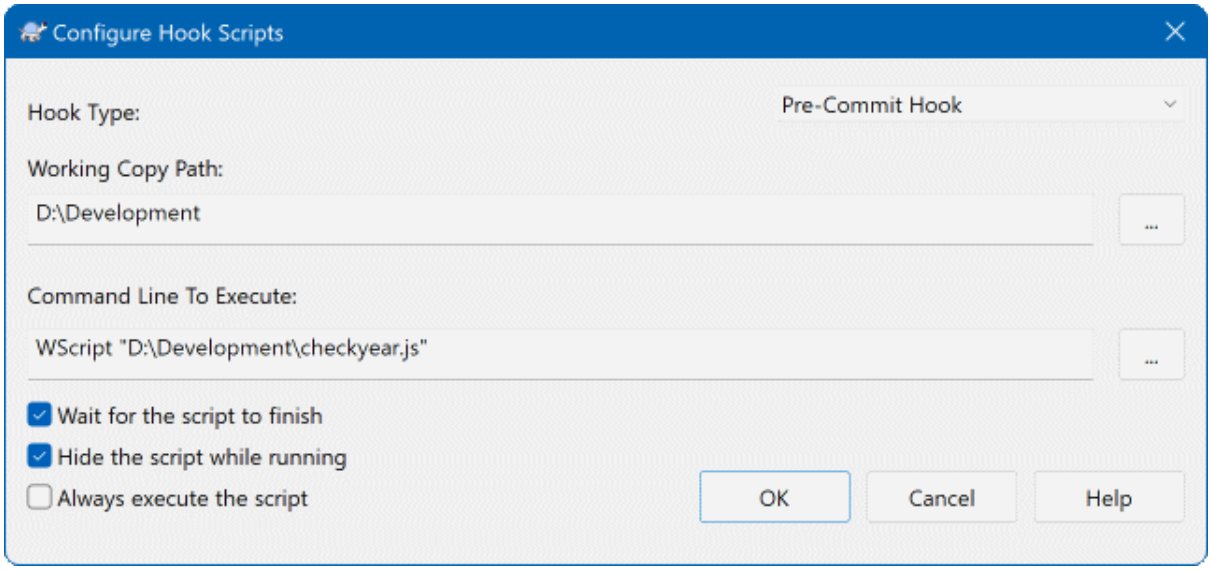


Figure 4.90. La fenêtre de paramétrage, configuration des scripts de hook

Pour ajouter un nouveau script hook, cliquez simplement sur Ajouter et saisissez les détails.

Il y a actuellement ces types de scripts 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 manuel
: If this is specified, the commit dialog shows a button Run Hook which when clicked runs the specified hook script. The hook script receives a list of all checked files and folders and the commit message if there was one entered.

Vérification de la livraison
: Called after the user clicks OK in the commit dialog, and before the commit dialog closes. This hook gets a list of all the checked files. If the hook returns an error, the commit dialog stays open.

  If the returned error message contains paths on newline separated lines, those paths will get selected in the commit dialog after the error message is shown.

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
: Called after the commit finishes successfully.

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 ou l'inversion Subversion ne commence.

Post-update
: Appelé après une mise à jour, une inversion ou une extraction (quelle soit réussie ou non).

Pré-connection
: Appelé avant une tentative de communication avec le dépôt. Appelé au plus une fois en cinq minutes.

Pré-verrouillage
: Appelé avant une tentative de verrouillage d'un fichier.

Post-verrouillage
: Appelé après qu'un fichier ait été verrouillé.

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 spécifiez la ligne de commande à exécuter, en commençant par le chemin du script hook ou de l'exécutable. Cela peut être un fichier batch, un fichier exécutable ou an autre fichier qui a une association de fichier windows valide, ex. un script perl. Noter que le script ne doit pas utiliser un chemin UNC comme le Shell execute de Windows n'autorise pas l'exécution de tels scripts à cause de restrictions de sécurité.

La ligne de commande comporte plusieurs paramètres qui sont renseignés par TortoiseSVN. Les paramètres disponibles dépendent du script de hook appelé. Chaque script de hook a ses propres paramètres qui sont passés dans l'ordre suivant:

Start-commit
: `PATHMESSAGEFILECWD`

Pre-commit manuel
: `PATHMESSAGEFILECWD`

Vérification de la livraison
: `PATHMESSAGEFILECWD`

Pre-commit
: `PATHDEPTHMESSAGEFILECWD`

Post-commit
: `PATHDEPTHMESSAGEFILEREVISIONERRORCWD`

Start-update
: `PATHCWD`

Pre-update
: `PATHDEPTHREVISIONCWD`

Post-update
: `PATHDEPTHREVISIONERRORCWDRESULTPATH`

Pré-connection
: no parameters are passed to this script. You can pass a custom parameter by appending it to the script path.

Pré-verrouillage
: `PATHLOCKFORCEMESSAGEFILECWD`

Post-verrouillage
: `PATHLOCKFORCEMESSAGEFILEERRORCWD`

La signification de chacun des paramètres est décrite ici :

PATH
: A path to a temporary file which contains all the paths for which the operation was started in UTF-8 encoding. Each path is on a separate line in the temp file.

Notez que pour ces opérations faites à distance, ex. dans le navigateur de dépôt, ces chemins ne sont pas des chemins locaux mais les urls des éléments affectés.

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
La révision du dépôt pour laquelle la mise à jour devrait être faite ou après qu'une livraison est terminée.

LOCK
Soit `vrai` lors du verrouillage, ou `faux` lors du déverrouillage.

FORCE
Soit `vrai` ou `faux`, selon que l'opération soit forcée ou non.

ERROR
Chemin vers un fichier contenant le message d'erreur. S'il n'y a pas d'erreur, le fichier sera vide.

CWD
Le répertoire de travail courant dans lequel le script est exécuté. Il est fixé dans le répertoire racine commun à tous les chemins affectés.

RESULTPATH
A path to a temporary file which contains all the paths in UTF-8 encoding which were somehow touched by the operation. Each path is on a separate line in the temp file.

Notez que, bien que nous ayons donné des noms à ces paramètres pour plus de commodité, vous n'avez pas à vous référer à ces noms dans la configuration de hook. Tous les paramètres définis pour un hook particulier sont toujours transmis, que vous le souhaitiez ou non ;-)

Si vous voulez que l'opération Subversion attende que le hook soit terminé, vérifiez Attendez que le script se termine.

Normally you will want to hide ugly DOS boxes when the script runs, so Hide the script while running is checked by default. Also you need to check this if your hook script might return an error that should stop the operation. The `forcer` flag can be set if the user must not proceed with the operation without running the script, i.e. the script must always run. If the `forcer` flag is not checked, then the user is shown a button Retry without hooks to retry the operation without running the hook script.

Sample client hook scripts can be found in the `contrib` folder in the *TortoiseSVN repository* [https://svn.code.sf.net/p/tortoisesvn/code/trunk/contrib/hook-scripts]. (Section 3, « Licence » explains how to access the repository.)

When debugging hook scripts you may want to echo progress lines to the DOS console, or insert a pause to stop the console window disappearing when the script completes. Because I/O is redirected this will not normally work. However you can redirect input and output explicitly to CON to overcome this. e.g.

```
echo Checking Status > con
pause < con > con
```

Un petit outil est inclus dans le dossier d'installation de TortoiseSVN nommé `ConnectVPN.exe`. Vous pouvez utiliser cet outil configuré comme un hook pré-connecté pour vous connecter automatiquement à votre VPN, avant que TortoiseSVN n'essaie de se connecter à un dépôt. Indiquez simplement le nom de la connexion VPN comme premier paramètre à l'outil.

4.31.8.1. Intégration d'un gestionnaire d'incidents

TortoiseSVN peut utiliser un plugin COM pour interroger le gestionnaire d'incidents dans la boîte de dialogue de livraison. L'utilisation de tels plugins est décrite à Section 4.29.2, « Récupérer des informations depuis le gestionnaire d'incidents ». Si votre administrateur système vous a fourni un plugin, que vous avez déjà installé et enregistré, c'est le lieu où préciser comment l'intègrer à votre copie de travail.

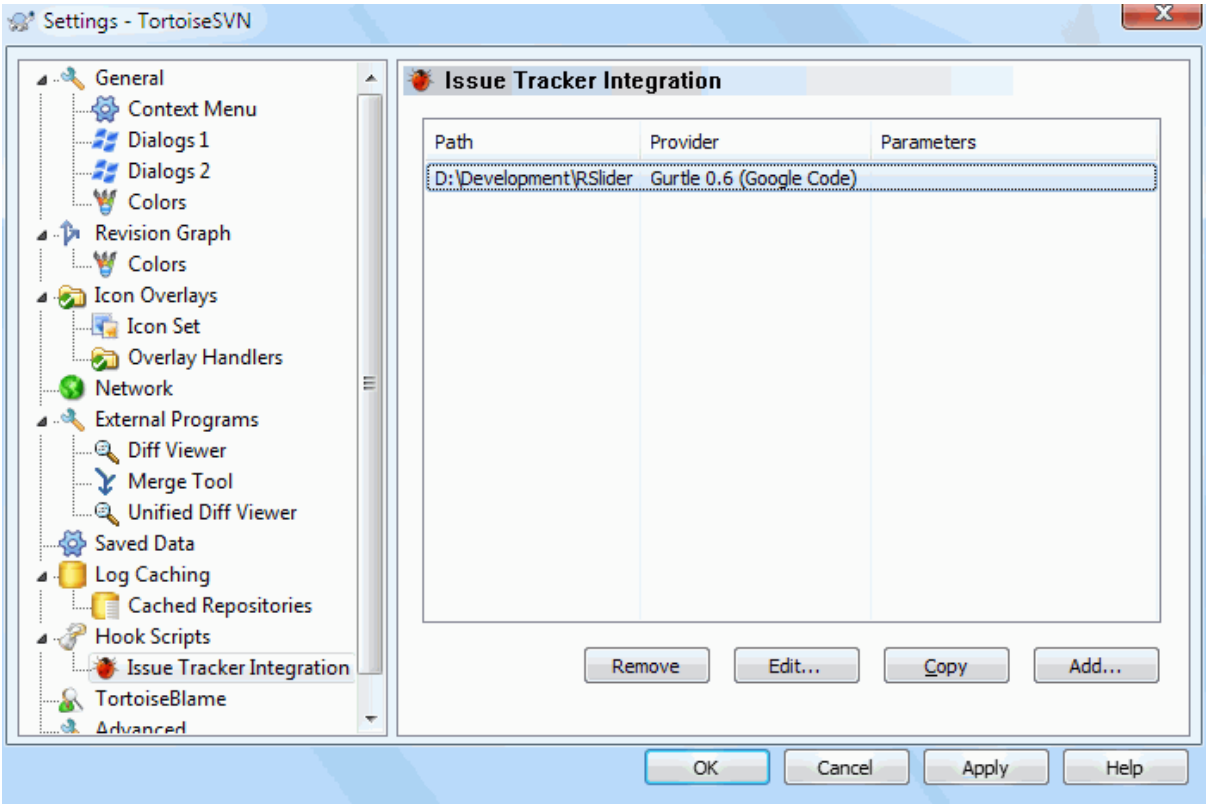


Figure 4.91. La Fenêtre de Propriétés, Page d'Intégration d'un Gestionnaire d'Incidents

Cliquez sur Ajouter... pour utiliser le plugin avec une copie de travail particulière. Ici vous pouvez spécifier le chemin de la copie de travail, choisir le plugin à utiliser à partir d'une liste déroulante de tous les plugins enregistrés de gestion d'incidents, et tous les paramètres à transmettre. Les paramètres sont spécifiques au plugin, mais peuvent inclure votre nom d'utilisateur du gestionnaire d'incidents, afin que le plugin puisse récupérer les incidents qui vous sont assignés.

Si vous souhaitez que les utilisateurs utilisent tous le même plug-in COM pour votre projet, vous pouvez aussi spécifier le plugin avec les propriétés `bugtraq:provideruuid`, `bugtraq:provideruuid64` et `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:provideruuid64
: C'est le même que `bugtraq:provideruuid`, mais pour la version 64-bit de IBugtraqProvider.

bugtraq:providerparams
: Cette propriété spécifie les paramètres envoyés à IBugTraqProvider.

Veuillez consultez la documentation de votre plugin IBugtraqProvider pour savoir ce qu'il faut indiquer dans ces deux propriétés.

4.31.9. Configuration de TortoiseBlame

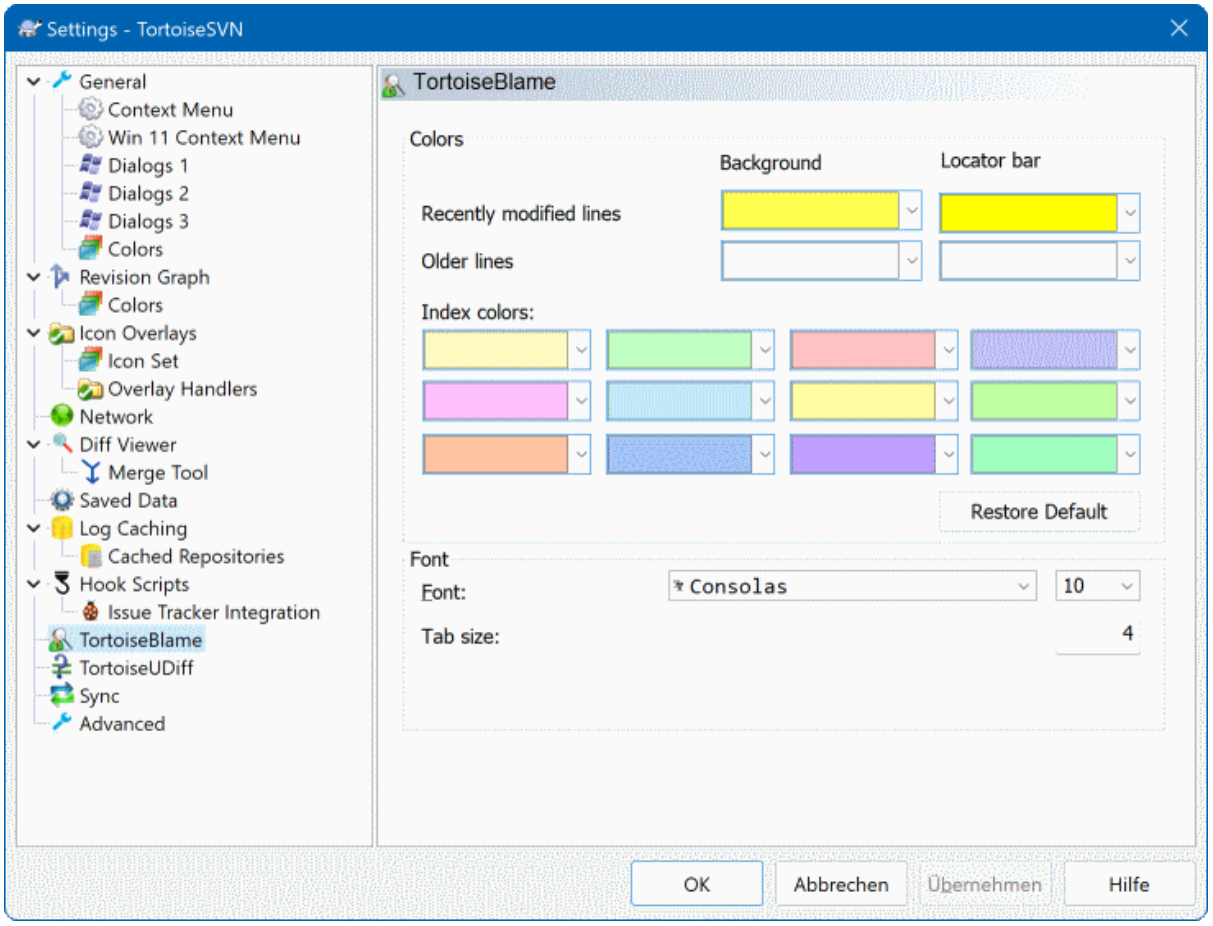


Figure 4.92. La boîte de dialogue ce configuration, page de bannissement.

Les paramètres utilisés par TortoiseBlame sont contrôlés à partir du menu contextuel principal, et non directement par TortoiseBlame.

Couleurs
: TortoiseBlame utilise la couleur de fond pour indiquer l'âge des lignes. Vous spécifiez les couleurs extrêmes en choisissant une couleur pour la révision la plus récente et une autre pour la révision la plus ancienne.

TortoiseBlame applique une interpolation linéaire entre ces deux couleurs afin de colorer chaque ligne en fonction de son numéro de révision.

Vous pouvez spécifier des couleurs différentes à utiliser pour la barre de repère. La valeur par défaut est d'utiliser fort contraste sur la barre de repère tout en gardant la fenêtre principale de lumière de fond de sorte que vous pouvez toujours lire le texte.t

Police
: Vous pouvez choisir la police utilisée pour afficher le texte et la taille à utiliser. Cela vaut tant pour le contenu du fichier, que pour l'auteur et les informations de révision figurant dans le volet de gauche.

Tabulations
: Définit combien d'espaces utiliser à la place d'une tabulation dans le fichier.

4.31.10. Réglages TortoiseUDiff

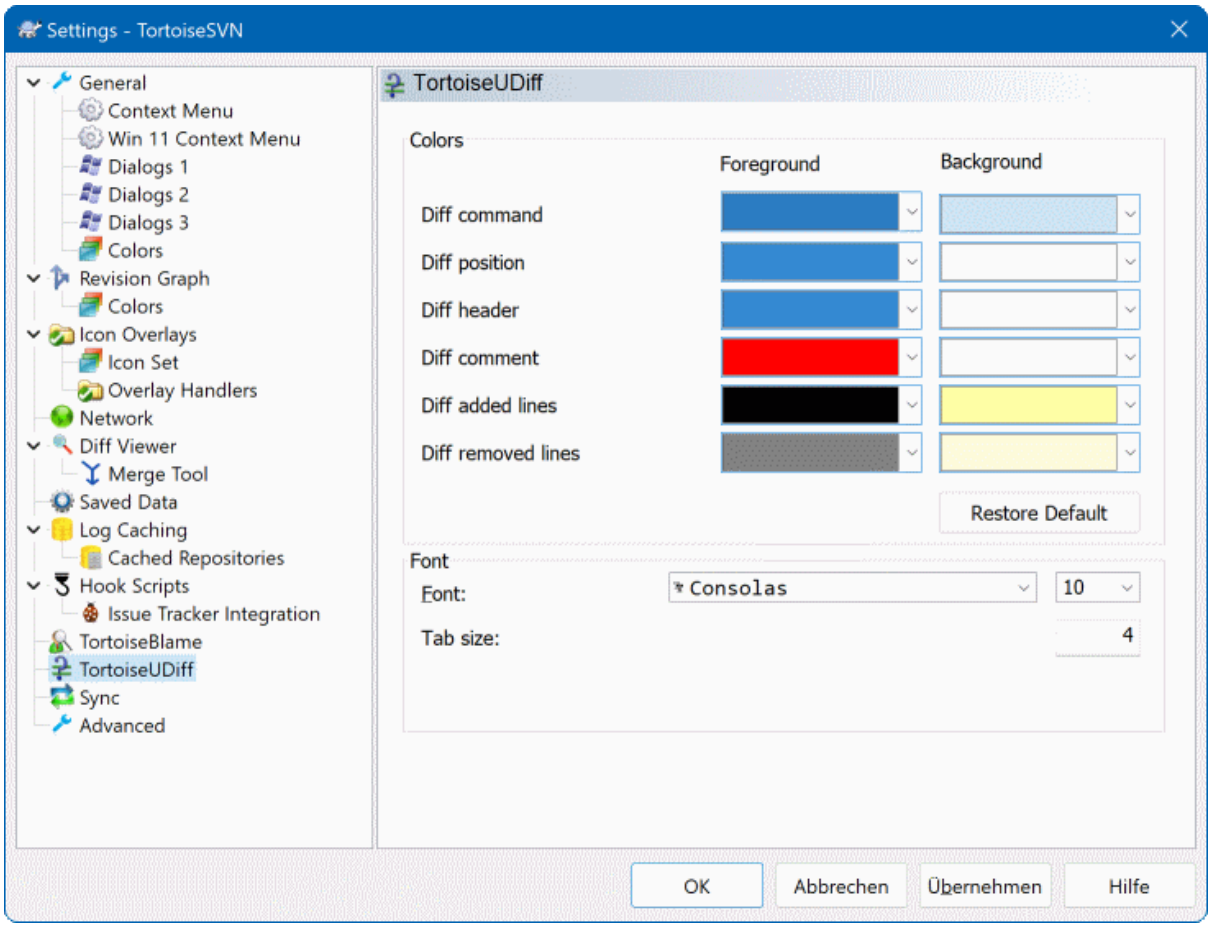


Figure 4.93. La boite de dialogue de configuration, page TortoiseUDiff

Les paramètres utilisés par TortoiseUDiff sont contrôlés à partir du menu contextuel principal, et non directement par TortoiseUDiff lui-même.

Couleurs
: Les couleurs par défaut utilisés par TortoiseUDiff sont habituellement correctes, mais vous pouvez les configurer ici.

Police
: Vous pouvez sélectionner la police utilisée pour afficher le texte et la taille de point à utiliser.

Tabulations
: Définit combien d'espaces utiliser à la place d'une tabulation dans le fichier diff.

4.31.11. Export des Paramètres de TSVN

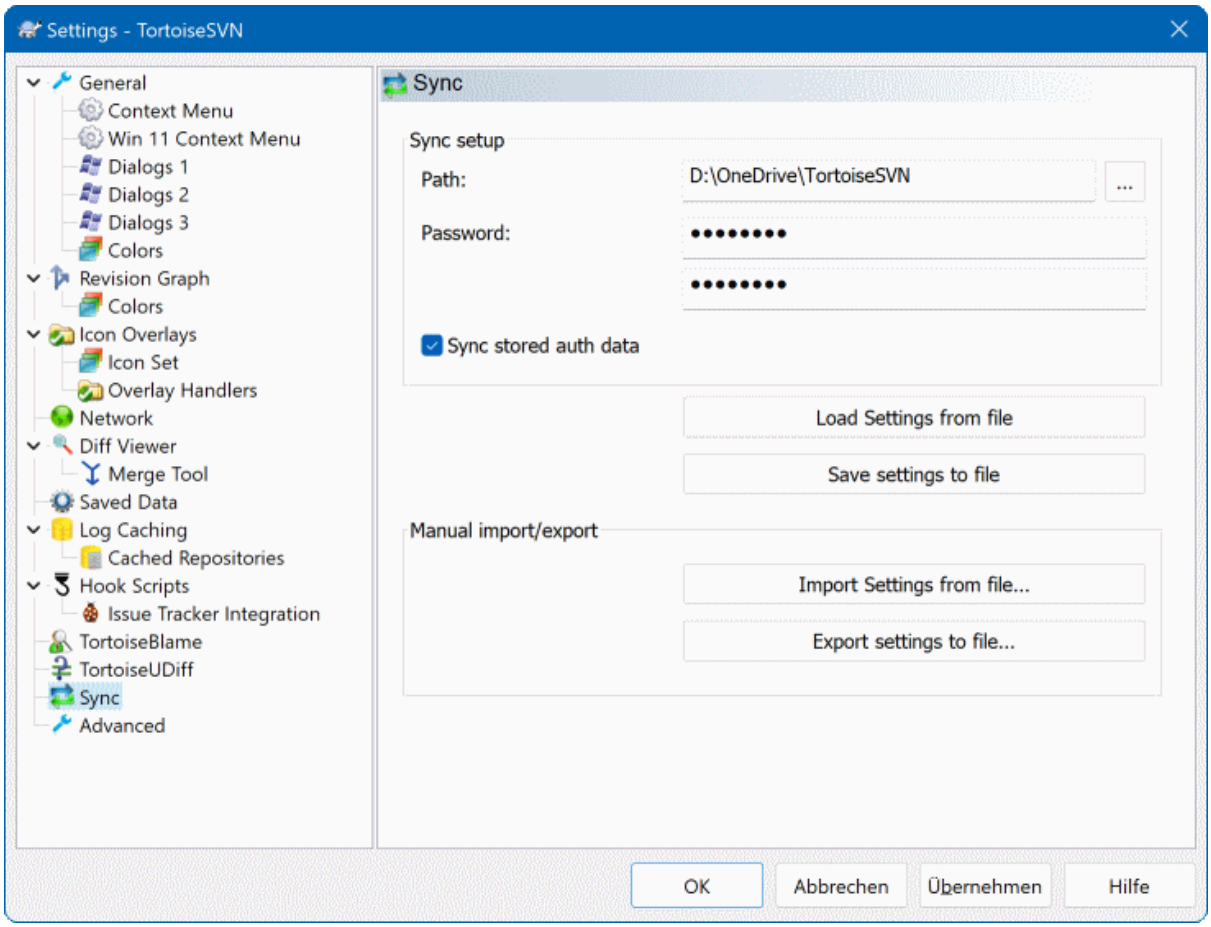


Figure 4.94. La boîte de dialogue Configuration, page Synchro

You can sync all TortoiseSVN settings to and from an encrypted file. The file is encrypted with the password you enter so you don't have to worry if you store that file on a cloud folder like OneDrive, GDrive, DropBox, ...

When a path and password is specified, TortoiseSVN will sync all settings automatically and keep them in sync.

You can also export/import an encrypted files with all the settings manually. When you do that, you're asked for the path of the file and the password to encrypt/decrypt the settings file.

When exporting the settings manually, you can also optionally include all local settings which are not included in a normal export or in a sync. Local settings are settings which include local paths which usually vary between computers. These local settings include the configured diff and merge tools and hook scripts.

4.31.12. Réglages Avancés

Quelques réglages rarement utilisés ne sont disponibles que dans la page Avancé de la boîte de dialogue des paramètres. Ces paramètres modifient le registre directement et vous devez savoir pour quoi chacun de ces paramètres est utilisé et ce qu'il fait. Ne modifiez pas ces paramètres, sauf si vous êtes sûr d'avoir besoin de le faire.

AllowAuthSave
: Parfois, plusieurs utilisateurs utilisent le même compte sur le même ordinateur. Dans de telles situations, il n'est pas vraiment souhaitable de sauvegarder les données d'authentification. Définir cette valeur à `faux` désactive le bouton `sauvegarder l'authentification` dans la boîte de dialogue d'authentification.

AllowUnversionedObstruction
: Si une mise à jour ajoute un nouveau fichier à partir du dépôt qui existe déjà dans la copie de travail locale comme fichier non versionné, l'action par défaut est de conserver le fichier local, en le montrant comme une

version (éventuellement) modifiée du nouveau fichier issu du dépôt. Si vous préférez que TortoiseSVN crée un conflit dans de telles situations, définissez cette valeur à `faux`.

AlwaysExtendedMenu
Comme avec l'explorateur, TortoiseSVN montre des commandes supplémentaires si la touche **Majkeycap> est enfoncvrai.**

AutoCompleteMinChars
The minimum amount of chars from which the editor shows an auto-completion popup. The default value is `3`.

AutocompleteRemovesExtensions
La liste d'auto-complétion indiquée dans l'éditeur de message de livraison affiche les noms des fichiers listés pour la livraison. Pour inclure également ces noms sans les extensions, mettez cette valeur à `vrai`.

BlockPeggedExternals
File externals that are pegged to a specific revision are blocked by default from being selected for a commit. This is because a subsequent update would revert those changes again unless the pegged revision of the external is adjusted.

Set this value to `false` in case you still want to commit changes to such external files.

BlockStatus
Si vous ne voulez pas que l'explorateur mette à jour l'état des (overlays) alors qu'une autre commande TortoiseSVN est en cours d'exécution (par exemple Mettre à jour, Livrer, ...), définissez cette valeur à `vrai`.

CacheTrayIcon
Pour ajouter une icône de notification de cache pour le programme TSVNCache, mettez cette valeur à `true`. C'est vraiment utile seulement pour les développeurs puisque ça vous permet de fermer le programme proprement.

ColumnsEveryWhere
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, set this value to `true`. Note that the extra columns are only available in XP. Vista and later doesn't support that feature any more. However some third-party explorer replacements do support those even on Windows versions later than XP.

ConfigDir
Vous pouvez spécifier un emplacement différent pour le fichier de configuration Subversion ici. Ceci affectera toutes les opérations de TortoiseSVN.

CtrlEnter
Dans la plupart des dialogues de TortoiseSVN, vous pouvez utiliser **Ctrl + Entrée** pour fermer la boîte de dialogue comme si vous aviez cliqué sur le bouton OK. Si vous ne le souhaitez pas, mettez cette valeur à `fauxliteral>.`

Debug
Définissez ceci à `true` si vous voulez qu'une boîte de dialogue apparaisse pour chaque commande, affichant la ligne de commande utilisée pour lancer TortoiseProc.exe.

DebugOutputString
Définissez ceci à `true` si vous voulez que TortoiseSVN écrive des messages de débogage pendant l'exécution. Ces messages peuvent uniquement être capturés qu'avez des outils de débogages spéciaux.

DialogTitles
The format par défaut (valeur 0) des titres de boîtes de dialogue est `url/chemin - nom de la boîte de dialogue - TortoiseSVN`. Si vous définissez cette valeur à 1, le format change pour `nom de la boîte de dialogue - url/chemin - TortoiseSVN`.

DiffBlamesWithTortoiseMerge
: TortoiseSVN vous permet d'utiliser un comparateur de fichier tiers. Cependant, la plupart de ces logiciels ne sont pas adaptés à la gestion d'annotation (Section 4.24.2, « Annoter les différences »), auquel cas, vous pourriez vouloir revenir à TortoiseMerge. Pour ce faire, mettez cette valeur à `true`.

DlgStickySize
: This value specifies the number of pixels a dialog has to be near a border before the dialog sticks to it. The default value is 3. To disable this value set the value to zero.

FixCaseRenames
: Some apps change the case of filenames without notice but those changes aren't really necessary nor wanted. For example a change from `file.txt` to `FILE.TXT` wouldn't bother normal Windows applications, but Subversion is case sensitive in these situations. So TortoiseSVN automatically fixes such case changes.

  If you don't want TortoiseSVN to automatically fix such case changes for you, you can set this value to `false`.

FullRowSelect
: Le contrôle de la liste des états qui est utilisé dans différentes boîtes de dialogue (par exemple livrer, vérifier les modifications, ajouter, revenir en arrière, ...) sélectionne les lignes en entier (par exemple, si vous sélectionnez une entrée, la ligne complète est sélectionnée, pas seulement la première colonne). C'est très bien, mais la ligne sélectionnée recouvre alors l'image de fond en bas à droite, ce qui peut sembler laid. Pour désactiver la sélection complète de ligne, définir cette valeur à `fauxliteral>.`

GroupTaskbarIconsPerRepo
: This option determines how the Win7 taskbar icons of the various TortoiseSVN dialogs and windows are grouped together. This option has no effect on Vista!

  1. La valeur par défaut est 0. Avec ce paramètre, les icônes sont regroupées par type d'application. Toutes les boîtes de dialogue de TortoiseSVN sont regroupées, toutes les fenêtres de TortoiseMerge sont regroupées, ...

Figure 4.95. Barre des taches avec groupement par défaut

  2. Si mis à 1, alors au lieu d'avoir toutes les boîtes de dialoguue dans un groupe par application, elles sont regroupées par dépôt. Par exemple, si vous avez une boîte de dialogie de log et une de commit ouvertes pour le dépôt A, et une boîte de dialogue de vérification de modifications et de log pour le dépôty B, il y a alors deux groupes d'icônes d'appplication affichées dans la barre des tâches de Win7, un groupe par dépôt. Mais les fenêtres TortoiseMerge ne sont pas regroupées avec les boîtes de dialogue TortoiseSVN.

Figure 4.96. Barre des taches avec groupement par dépôt

  3. If set to 2, then the grouping works as with the setting set to 1, except that TortoiseSVN, TortoiseMerge, TortoiseBlame, TortoiseIDiff and TortoiseUDiff windows are all grouped together. For example, if you have the commit dialog open and then double click on a modified file, the opened TortoiseMerge diff window will be put in the same icon group on the taskbar as the commit dialog icon.

Figure 4.97. Barre des taches avec groupement par dépôt

4. If set to 3, then the grouping works as with the setting set to 1, but the grouping isn't done according to the repository but according to the working copy. This is useful if you have all your projects in the same repository but different working copies for each project.

5. If set to 4, then the grouping works as with the setting set to 2, but the grouping isn't done according to the repository but according to the working copy.

GroupTaskbarIconsPerRepoOverlay
: Ceci n'a pas d'effet si l'option `GroupTaskbarIconsPerRepo` est définie à 0 (voir ci-dessus).

  Si cette option est mise à `true`, alors chaque icône de la barre des tâches de Win7 affiche un petit rectangle coloré en surimpression, indiquant pour quel dépôt les boîtes de dialogue/fenêtres sont utilisées.

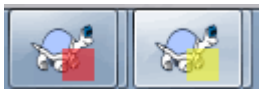


Figure 4.98. Taskbar grouping with repository color overlays

HideExternalInfo
: If this is set to `false`, then every `svn:externals` is shown during an update separately.

  If it is set to `true` (the default), then update information for externals is only shown if the externals are affected by the update, i.e. changed in some way. Otherwise nothing is shown as with normal files and folders.

HookCancelError
: If this is set to `true`, then cancelling the dialog to approve a hook script to run will show an error dialog indicating the user cancelled.

IncludeExternals
: Par défaut, TortoiseSVN fonctionne toujours avec une mise à jour des parties extérieures incluses. Cela évite les problèmes avec les copies de travail incompatibles. Si vous avez quand même beaucoup de parties extérieures, une mise à jour peut prendre un certain temps. Définissez cette valeur à `faux` pour exécuter la mise à jour par défaut avec les parties extérieures exclues. Pour mettre à jour avec les parties extérieures comprises, exécutez soit la boîte de dialogue `Mettre à jour à la révision...` ou définissez cette valeur à nouveau à `vrailiteral>.`

LogFindCopyFrom
: Lorsque la fenêtre de log est lancée depuis le gestionnaire de fusion, les révisions déjà fusionnées apparaissent en gris. Celles créées au-delà du point de création de la branche sont également affichées mais en noir puisqu'elles ne peuvent être fusionnées.

  Si cette option est positionnée à `true` alors TortoiseSVN essaye de trouver la révision où la branche a été créée et cache toutes les révisions au-delà. Par défaut, cette option est désactivé car coûteuse en temps. De

plus, cette option ne fonctionne pas avec certains server SVN (ex: Google Code Hosting, voir *issue #5471* [http://code.google.com/p/support/issues/detail?id=5471]).

LogMultiRevFormat
A format string for the log messages when multiple revisions are selected in the log dialog.

You can use the following placeholders in your format string:

%1!ld!
gets replaced with the revision number text

%2!s!
gets replaced with the short log message of the revision

LogStatusCheck
La boîte de dialogue journal affiche en gras la copie de travail dans laquelle se trouve la révision. Mais cela demande que la boîte de dialogue journal récupère le statut du chemin. Alors que pour des copies de travail très volumineuses cela peut prendre un certain temps, vous pouvez définir cette valeur à `faux` pour désactiver cette fonctionnalité.

MaxHistoryComboItems
Comboboxes for URLs and paths show a history of previously used URLs/paths if possible. This settings controls how many previous items are saved and shown. The default is 25 items.

MergeLogSeparator
Lorsque vous fusionnez les révisions d'une autre branche, et que les informations de suivi de fusion sont disponibles, les messages de log des révisions que vous fusionnez seront collectés pour constituer un message de livraison. Une chaîne pré-définie est utilisée pour séparer les messages individuel de log des révisions fusionnées. Si vous préférez, vous pouvez définir ceci à une valeur contenant un caractère de séparation de votre choix.

NumDiffWarning
If you want to show the diff at once for more items than specified with this settings, a warning dialog is shown first. The default is 10.

OldVersionCheck
TortoiseSVN vérifie s'il y a une nouvelle version disponible environ une fois par semaine. Si une version mise à jour est trouvée, la boîte de dialogue de livraison montre un lien de contrôle avec cette info. Si vous préférez l'ancien comportement où une boîte de dialogue apparaît pour vous informer sur la mise à jour, mettez cette valeur à `vrai`.

RepoBrowserTrySVNParentPath
The repository browser tries to fetch the web page that's generated by an SVN server configured with the SVNParentPath directive to get a list of all repositories. To disable that behavior, set this value to `false`.

ScintillaBidirectional
This option enables the bidirectional mode for the commit message edit box. If enabled, right-to-left language text editing is done properly. Since this feature is expensive, it is disabled by default. You can enable this by setting this value to `true`.

ScintillaDirect2D
This option enables the use of Direct2D accelerated drawing in the Scintilla control which is used as the edit box in e.g. the commit dialog, and also for the unified diff viewer. With some graphic cards however this sometimes doesn't work properly so that the cursor to enter text isn't always visible. If that happens, you can turn this feature off by setting this value to `false`.

OutOfDateRetry
This parameter specifies how TortoiseSVN behaves if a commit fails due to an out-of-date error:

0
The user is asked whether to update the working copy or not, and the commit dialog is not reopened after the update.

1
: This is the default. The user is asked whether to update the working copy or not, and the commit dialog is reopened after the update so the user can proceed with the commit right away.

2
: Similar to `1`, but instead of updating only the paths selected for a commit, the update is done on the working copy root. This helps to avoid inconsistent working copies.

3
: The user is not asked to update the working copy. The commit simply fails with the out-of-date error message.

PlaySound
: If set to `true`, TortoiseSVN will play a system sound when an error or warning occurs, or another situation which is important and requires your attention. Set this to `false` if you want to keep TortoiseSVN quiet. Note that the project monitor has its own setting for playing sounds, which you can configure in its settings dialog.

ShellMenuAccelerators
: TortoiseSVN utilise des raccourcis pour ses entrées de menu contextuel. Cela peut conduire à des raccourcis en double (par exemple, `SVN Livrer` a le raccourci **Alt-C**, de même que `Copier` dans l'explorateur). Si vous n'avez pas envie ou besoin des raccourcis des entrées TortoiseSVN, définissez cette valeur à `faux`.

ShowContextMenuIcons
: Cela peut être utile si vous utilisez autre chose que l'explorateur Windows ou si vous avez des problèmes avec le menu contextuel qui ne s'affiche pas correctement. Mettez cette valeur à `faux` si vous ne souhaitez pas que TortoiseSVN affiche les icônes des éléments du menu contextuel. Mettez cette valeur à `vrai` pour afficher à nouveau les icônes.

ShowAppContextMenuIcons
: Si vous ne voulez pas que TortoiseSVN affiche les icônes des menus contextuels dans ses propres boîtes de dialogue, mettez cette valeur à `fauxliteral>.`

ShowNotifications
: Set this value to `false` if you don't want the project monitor to show notification popups when new commits are detected.

StyleCommitMessages
: La boîte de dialogue de livraison et de log utilise des styles (par exemple gras, italique) dans les messages de livraison (voir Section 4.4.5, « Commentaires de livraison » pour plus de détails). Si vous ne le voulez pas, définissez la valeur à `faux`.

UpdateCheckURL
: Cette valeur contient l'URL à partir de laquelle TortoiseSVN essaie de télécharger un fichier texte pour savoir s'il y a des mises à jour disponibles. Cela pourrait être utile pour les administrateurs d'entreprise qui ne veulent pas que leurs utilisateurs mette à jour TortoiseSVN sans leur approbation.

UseCustomWordBreak
: The standard edit controls do not stop on forward slashes like they're found in paths and urls. TortoiseSVN uses a custom word break procedure for the edit controls. If you don't want that and use the default instead, set this value to 0. If you only want the default for edit controls in combo boxes, set this value to 1.

VersionCheck
: TortoiseSVN vérifie s'il existe une nouvelle version disponible environ une fois par semaine. Si vous ne voulez pas que TortoiseSVN fasse ce contrôle, définissez cette valeur à `faux`.

4.32. Étape Finale

Faites une donation!

Even though TortoiseSVN and TortoiseMerge are free, you can support the developers by sending in patches and playing an active role in the development. You can also help to cheer us up during the endless hours we spend in front of our computers.

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: https://tortoisesvn.net/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. Moniteur de projet

Le moniteur de projet est un outil utile qui surveille les dépôts et vous notifie dans le cas où il y ai de nouvelles livraisons.

Les projets peuvent être surveillés via un chemin de copie de travail ou directement via leurs URLs de dépôt.

The project monitor scans each project in a configurable interval, and every time new commits are detected a notification popup is shown. Also the icon that is added to the system tray changes to indicate that there are new commits.

Snarl

If *Snarl* [https://snarl.fullphat.net/] is installed and active, then the project monitor automatically uses Snarl to show the notifications about newly detected commits.

5.1. Adding projects to monitor

If you first start the project monitor, the tree view on the left side is empty. To add projects, click on the button at the top of the dialog named Add Project.

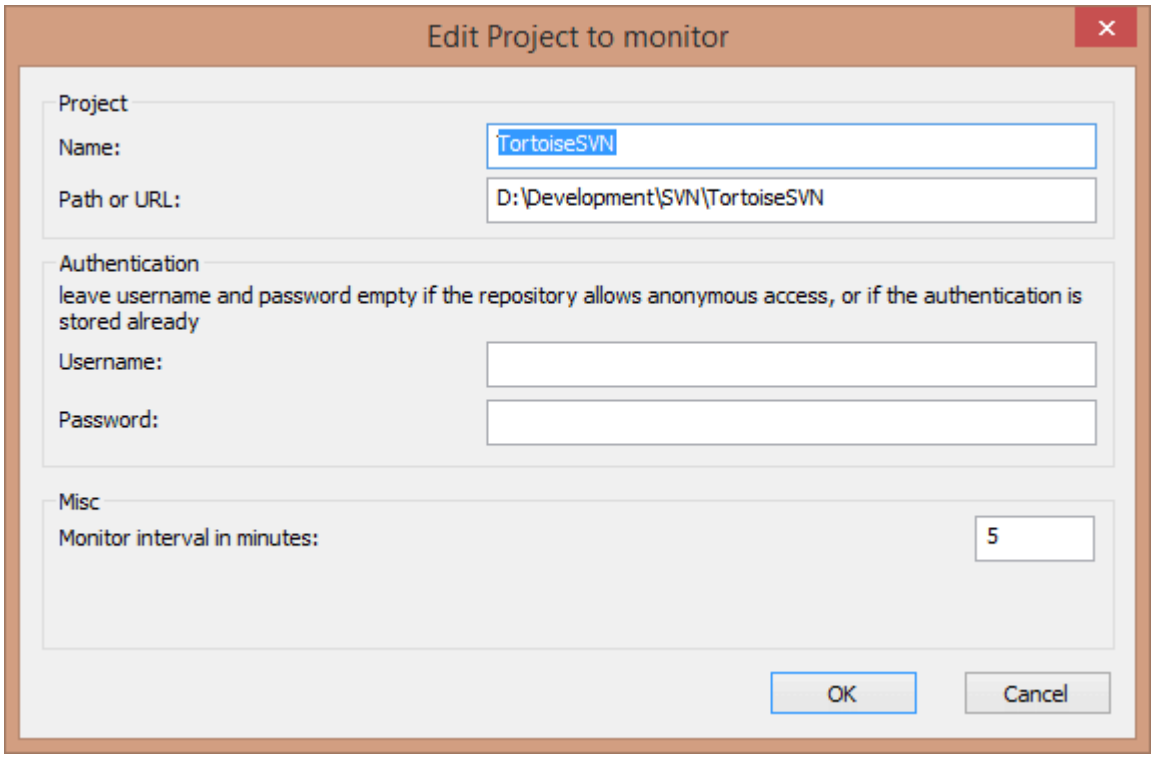


Figure 5.1. The edit project dialog of the project monitor

To add a project for monitoring, fill in the required information. The name of the project is not optional and must be filled in, all other information is optional.

If the box for `Path or Url` is left empty, then a folder is added. This is useful to group monitored projects.

If you want to monitor all repositories served via the *SVNParentPath* [http://svnbook.red-bean.com/en/1.8/svn.serverconfig.httpd.html#svn.serverconfig.httpd.basic] directive, enter the root Url for your repositories and check the box `Url points to SVNParentPath list`.

The fields `Username` and `Password` should only be filled in if the repository does not provide anonymous read access, and only if the authentication is not stored by Subversion itself. If you're accessing the monitored repository with TortoiseSVN or other svn clients and you've stored the authentication already, you should leave this empty: you won't have to edit those projects manually if the password changes.

The `Monitor interval in minutes` specifies the minutes to wait in between checks. The smallest interval is one minute.

check interval

If there are a lot of users monitoring the same repository and the bandwidth on the server is limited, a repository admin can set the minimum for check intervals using an `svnrobots.txt` file. A detailed explanation on how this works can be found on the project monitor website:

https://tools.stefankueng.com/svnrobots.html [https://tools.stefankueng.com/svnrobots.html]

5.2. Monitor dialog

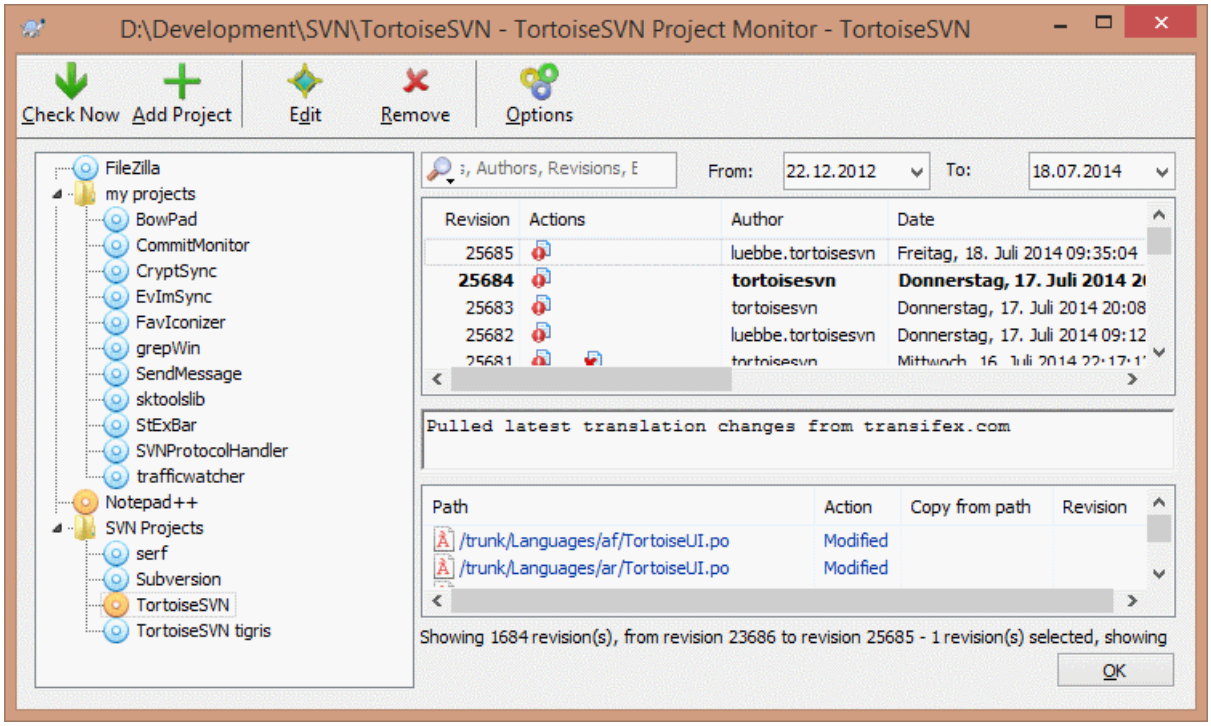


Figure 5.2. The main dialog of the project monitor

The project monitor shows all monitored projects on the left in a tree view. The projects can be moved around, for example one project can be moved below another project, making it a child/subproject.

A click on a project shows all the log messages of that project on the right.

Projects that have updates are shown in bold, with the number of new commits in brackets at the right. A click on a project marks it automatically as read.

5.2.1. Main operations

The toolbar at the top of the dialog allows to configure and operate the project monitor.

Vérifier maintenant
: While each monitored project is checked according to the interval that's set up, clicking this button will force a check of all projects immediately. Note that if there are updates, the notification won't show up until all projects have been checked.

Ajouter un projet
: Opens a new dialog to set up a new project for monitoring.

Editer
: Opens the configuration dialog for the selected project.

Supprimer
: Removes the selected project after a confirmation dialog is shown.

Marquer tout comme lu
: Marks all revisions in all projects as read. Note that if you select a project with unread revisions, those revisions are automatically marked as read when you select another project.

  If you hold down the **Shift** key when clicking the button, all error states are also cleared if there are any.

Tout mettre à jour
: Runs an **Update** on all monitored working copies. Projects that are monitored via an url are not updated, only those that are set up with a working copy path.

Options
: Shows a dialog to configure the behavior of the project monitor.

Chapitre 6. 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 ».

6.1. La ligne de commande SubWCRev

SubWCRev reads the Subversion status of all files in a working copy, excluding externals by default. It records the highest commit revision number found, and the commit timestamp of that revision, it also records whether there are local modifications in the working copy, or mixed update revisions. The revision number, update revision range and modification status are displayed on stdout.

SubWCRev.exe is called from the command line or a script, and is controlled using the command line parameters.

```
SubWCRev WorkingCopyPath [SrcVersionFile DstVersionFile] [-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 dépôt. 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 dépôt 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.

You can specify ignore patterns for SubWCRev to prevent specific files and paths from being considered. The patterns are read from a file named `.subwcrevignore`. The file is read from the specified path, and also from the working copy root. If the file does not exist, no files or paths are ignored. The `.subwcrevignore` file can contain multiple patterns, separated by newlines. The patterns are matched against the paths relative to the repository root and paths relative to the path of the `.subwcrevignore` file. For example, to ignore all files in the `doc` folder of the TortoiseSVN working copy, the `.subwcrevignore` would contain the following lines:

```
/trunk/doc
/trunk/doc/*
```

Or, assuming the `.subwcrevignore` file is in the working copy root which is checked out from trunk, using the patterns

```
doc
doc/*
```

is the same as the example above.

To ignore all images, the ignore patterns could be set like this:

```
*.png
*.jpg
*.ico
*.bmp
```

Important

The ignore patterns are case-sensitive, just like Subversion is.

Astuce

To create a file with a starting dot in the Windows explorer, enter `.subwcrevignore.`. Note the trailing dot.

Un grand nombre d'options peuvent changer la manière dont SubZCRev fonctionne. Si vous en utilisez plus d'une, elles doivent etre spécifiées dans un même groupe, par exemple `nm` et non pas `-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.
-N	If this switch is given, SubWCRev will exit with `ERRORLEVEL 11` if the working copy contains unversioned items that are not ignored.
-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 dépôt. Le comportement par défaut est d'ignorer les références externes.
-E	If this switch is given, same as `-e`, but it ignores the externals with explicit revisions, when the revision range inside of them is only the given explicit revision in the properties. So it doesn't lead to mixed revisions.
-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'.
-F	If this switch is given, SubWCRev will ignore any `.subwcrevignore` files and include all files.
-q	If this switch is given, SubWCRev will perform the keyword substitution without showing working copy status on stdout.

Tableau 6.1. Liste des commutateurs de ligne de commande disponibles

S'il n'y a pas d'erreur, SubWCRev renvoie zéro. Mais dans le cas où une erreur survient, le message d'erreur est écrit sur la sortie d'erreur standard et affiché dans le console. Et les codes d'erreur renvoyés sont :

Code d'Erreur	Description
1	Erreur de syntaxe. Une ou plusieurs commandes sont invalides.
2	Le fichier ou le dossier spécifié dans le ligne de commande est introuvable.
3	Le fichier source n'a pas pu être ouvert, ou le fichier cible n'a pas pu être créé.

Code d'Erreur	Description
4	Impossible d'allouer la mémoire. Cela peut arrivé si, par exemple, le fichier source est trop gros.
5	Le fichier source ne peut pas être scanné correctement.
6	Erreur SVN : Subversion a renvoyé une erreur quand SubWCRev a essayé d'obtenir des informations de la copie de travail.
7	The working copy has local modifications. This requires the `-n` switch.
8	The working copy has mixed revisions. This requires the `-m` switch.
9	The output file already exists. This requires the `-d` switch.
10	Le chemin spécifié n'est pas une copie de travail ou n'appartient à aucune.
11	The working copy has unversioned files or folders in it. This requires the `-N` switch.

Tableau 6.2. Liste des codes d'erreur de SubWCRev

6.2. 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.
$WCREV&$	Replaced with the highest commit revision in the working copy, ANDed with the value after the & char. For example: `$WCREV&0xFFFF$`
$WCREV-$, $WCREV+$	Replaced with the highest commit revision in the working copy, with the value after the + or - char added or subtracted. For example: `$WCREV-1000$`
$WCDATE$, $WCDATEUTC$	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://msdn.microsoft.com/en-us/library/fe06s4ak.aspx].
$WCNOW$, $WCNOWUTC$	Remplacé par la date/heure actuelle du système. Cela peut être utile pour indiquer l'heure de compilation. Le format de l'heure est décrit comme ceci `$WCDATE$`.
$WCRANGE$	Replaced with the update revision range in the working copy. If the working copy is in a consistent state, this will be a single revision. If the working copy contains mixed revisions, either due to being out of date, or due to a deliberate update-to-revision, then the range will be shown in the form 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.
$WCUNVER$	`$WCUNVER?TText:FText$` is replaced with `TText` if there are unversioned items in the working copy, or `FText` if not.
$WCEXTALLFIXED$	`$WCEXTALLFIXED?TText:FText$` is replaced with `TText` if all externals are fixed to an explicit revision, or `FText` if not.
$WCISTAGGED$	`$WCISTAGGED?TText:FText$` is replaced with `TText` if the repository URL contains the tags classification pattern, or `FText` if not.

Mot-clé	Description
$WCURL$	Remplacé par l'URL du dépôt 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.
$WCLOCKDATE$, $WCLOCKDATEUTC$	Remplacé par la date de verrouillage. La mise en forme de l'heure peut être utilisée comme décrit pour `$WCDATE$`.
$WCLOCKOWNER$	Remplacer par le nom du propriétaire du verrou
$WCLOCKCOMMENT$	Remplacé par le commentaire du verrou
$WCUNVER$	`$WCUNVER?TText:FText$` is replaced with `TText` if there are unversioned files or folders in the working copy, or `FText` if not.

Tableau 6.3. Les des mots-clés disponibles

SubWCRev does not directly support nesting of expressions, so for example you cannot use an expression like:

```
#define SVN_REVISION    "$WCMIXED?$WCRANGE$:$WCREV$$"
```

But you can usually work around it by other means, for example:

```
#define SVN_RANGE       $WCRANGE$
#define SVN_REV         $WCREV$
#define SVN_REVISION    "$WCMIXED?SVN_RANGE:SVN_REV$"
```

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 `$WCLOCKCOMMENT$`.

6.3. Exemple de mot-clé

L'exemple ci-dessous montre comment les mots-clés dans un fichier modèle sont substitués dans le fichier de résultat.

```
// Test file for SubWCRev

char *Revision      = "$WCREV$";
char *Revision16    = "$WCREV&0xFF$";
char *Revisionp100  = "$WCREV+100$";
char *Revisionm100  = "$WCREV-100$";
char *Modified      = "$WCMODS?Modified:Not modified$";
char *Unversioned   = "$WCUNVER?Unversioned items found:no unversioned items$";
char *Date          = "$WCDATE$";
```

```
char *CustDate      = "$WCDATE=%a, %d %B %Y$";
char *DateUTC       = "$WCDATEUTC$";
char *CustDateUTC   = "$WCDATEUTC=%a, %d %B %Y$";
char *TimeNow       = "$WCNOW$";
char *TimeNowUTC    = "$WCNOWUTC$";
char *RevRange      = "$WCRANGE$";
char *Mixed         = "$WCMIXED?Mixed revision WC:Not mixed$";
char *ExtAllFixed   = "$WCEXTALLFIXED?All externals fixed:Not all externals fixed$";
char *IsTagged      = "$WCISTAGGED?Tagged:Not tagged$";
char *URL           = "$WCURL$";
char *isInSVN       = "$WCINSVN?versioned:not versioned$";
char *needslck      = "$WCNEEDSLOCK?TRUE:FALSE$";
char *islocked      = "$WCISLOCKED?locked:not locked$";
char *lockdateutc   = "$WCLOCKDATEUTC$";
char *lockdate      = "$WCLOCKDATE$";
char *lockcustutc   = "$WCLOCKDATEUTC=%a, %d %B %Y$";
char *lockcust      = "$WCLOCKDATE=%a, %d %B %Y$";
char *lockown       = "$WCLOCKOWNER$";
char *lockcmt       = "$WCLOCKCOMMENT$";

#if $WCMODS?1:0$
#error Source is modified
#endif

// End of file
```

Après l'exécution de `SubWCRev.exe path\to\workingcopy testfile.tmpl testfile.txt`, le fichier de sortie `testfile.txt` doit être de la forme :

```
// Test file for SubWCRev

char *Revision      = "22837";
char *Revision16    = "53";
char *Revisionp100  = "22937";
char *Revisionm100  = "22737";
char *Modified      = "Modified";
char *Unversioned   = "no unversioned items";
char *Date          = "2012/04/26 18:47:57";
char *CustDate      = "Thu, 26 April 2012";
char *DateUTC       = "2012/04/26 16:47:57";
char *CustDateUTC   = "Thu, 26 April 2012";
char *TimeNow       = "2012/04/26 20:51:17";
char *TimeNowUTC    = "2012/04/26 18:51:17";
char *RevRange      = "22836:22837";
char *Mixed         = "Mixed revision WC";
char *ExtAllFixed   = "All externals fixed";
char *IsTagged      = "Not tagged";
char *URL           = "https://svn.code.sf.net/p/tortoisesvn/code/trunk";
char *isInSVN       = "versioned";
char *needslck      = "FALSE";
char *islocked      = "not locked";
char *lockdateutc   = "1970/01/01 00:00:00";
char *lockdate      = "1970/01/01 01:00:00";
char *lockcustutc   = "Thu, 01 January 1970";
char *lockcust      = "Thu, 01 January 1970";
char *lockown       = "";
char *lockcmt       = "";
```

```
#if 1
#error Source is modified
#endif

// End of file
```

Astuce

Un fichier comme ceci sera inclus dans la compilation, ainsi il faut s'attendre à ce qu'il versionné. Assurez-vous de versionner le fichier de modèle, et pas le fichier généré, sinon chaque fois que vous régénérer le fichier de version, vous aurez besoin de livrer la modification, ce qui au final signifie que le fichier de version doit être mise à jour.

6.4. 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	Cette méthode parcourt la copie de travail en regroupant les informations de révision. Naturellement, vous devez l'appeler avant d'utiliser les autres méthodes pour accéder à l'information. Le premier paramètre est le chemin. Le deuxième paramètre doit être vrai si vous souhaitez inclure les révisions de dossier. Équivalent à la commande `-f`. Le troisième paramètre doit être vrai si vous voulez inclure svn:externals. Équivalent à la commande `-e`.
.GetWCInfo2	The same as `GetWCInfo()` but with a fourth parameter that sets the equivalent to the `-E` command line switch.
.Revision	La plus haute révision de livraison dans la copie de travail. Equivalent à `$WCREV$`.
.Date	La date/heure de livraison de la plus haute révision de livraison. 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.
.HasUnversioned	Vrai s'il existe des éléments non versionnés
.Url	Remplacé par l'URL du référentiel de la copie de travail utilisée dans `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	Chaîne de caractère représentant la date à laquelle le verrou a été créé, ou une chaîne vide si l'élément n'est pas verrouillé.
.LockOwner	Chaîne de caractère représentant le propriétaire du verrou ou une chaîne vide si l'élément n'est pas verrouillé.
.LockComment	Le message renseigné au moment du verrouillage.

Tableau 6.4. Les méthodes COM/automation sont supportées

Les exemples suivants montrent comment l'interface devrait être utilisée.

```
// testCOM.js - fichier javascript
// script de test pour le l'objet SubWCRev COM/Automatisation

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.GetWCInfo2(
    filesystem.GetAbsolutePathName("..\\.."), 1, 1, 1);

wcInfoString1 = "Révision = " + revObject1.Revision +
                "\nRévision Minimum = " + revObject1.MinRev +
                "\nRévision Maximum = " + revObject1.MaxRev +
                "\nDate = " + revObject1.Date +
                "\nURL = " + revObject1.Url + "\nAuteur = " +
                revObject1.Author + "\nAModifications = " +
                revObject1.HasModifications + "\nEstElémentSVN = " +
                revObject1.IsSvnItem + "\nRequiertVerrouillage = " +
                revObject1.NeedsLocking + "\nEstVerrouillé = " +
                revObject1.IsLocked + "\nDateCréationVerrou = " +
                revObject1.LockCreationDate + "\nPropriétaireVerrou = " +
                revObject1.LockOwner + "\nCommentaireVerrou = " +
                revObject1.LockComment;
wcInfoString2 = "Révision = " + revObject2.Revision +
                "\nRévision Minimum = " + revObject2.MinRev +
                "\nRévision Maximum = " + revObject2.MaxRev +
                "\nDate = " + revObject2.Date +
                "\nURL = " + revObject2.Url + "\nAuteur = " +
                revObject2.Author + "\nAModifications = " +
                revObject2.HasModifications + "\nEstElémentSVN = " +
                revObject2.IsSvnItem + "\nRequiertVerrouillage = " +
                revObject2.NeedsLocking + "\nEstVerrouillé = " +
                revObject2.IsLocked + "\nDateCréationVerrou = " +
                revObject2.LockCreationDate + "\nPropriétaireVerrou = " +
                revObject2.LockOwner + "\nCommentaireVerrou = " +
                revObject2.LockComment;
wcInfoString3 = "Révision = " + revObject3.Revision +
                "\nRévision Minimum = " + revObject3.MinRev +
                "\nRévision Maximum = " + revObject3.MaxRev +
                "\nDate = " + revObject3.Date +
                "\nURL = " + revObject3.Url + "\nAuteur = " +
                revObject3.Author + "\nAModifications = " +
                revObject3.HasModifications + "\nEstElémentSVN = " +
                revObject3.IsSvnItem + "\nRequiertVerrouillage = " +
                revObject3.NeedsLocking + "\nEstVerrouillé = " +
                revObject3.IsLocked + "\nDateCréationVerrou = " +
```

```
                revObject3.LockCreationDate + "\nPropriétaireVerrou = " +
                revObject3.LockOwner + "\nCommentaireVerrou = " +
                revObject3.LockComment;
wcInfoString4 = "Révision = " + revObject4.Revision +
                "\nRévision Minimum = " + revObject4.MinRev +
                "\nRévision Maximum = " + revObject4.MaxRev +
                "\nDate = " + revObject4.Date +
                "\nURL = " + revObject4.Url + "\nAuteur = " +
                revObject4.Author + "\nAModifications = " +
                revObject4.HasModifications + "\nEstElémentSVN = " +
                revObject4.IsSvnItem + "\nRequiertVerrouillage = " +
                revObject4.NeedsLocking + "\nEstVerrouillé = " +
                revObject4.IsLocked + "\nDateCréationVerrou = " +
                revObject4.LockCreationDate + "\nPropriétaireVerrou = " +
                revObject4.LockOwner + "\nCommentaireVerrou = " +
                revObject4.LockComment;

WScript.Echo(wcInfoString1);
WScript.Echo(wcInfoString2);
WScript.Echo(wcInfoString3);
WScript.Echo(wcInfoString4);
```

La liste suivante montre comment utiliser l'objet SubWCRev COM de C#:

```
en utilisant LibSubWCRev;
SubWCRev sub = new SubWCRev();
sub.GetWCInfo("C:\\PathToMyFile\\MyFile.cc", true, true);
if (sub.IsSvnItem == true)
{
    MessageBox.Show("versionné");
}
else
{
    MessageBox.Show("non versionné");
}
```

Chapitre 7. IBugtraqProvider interface

Pour obtenir une intégration plus fine avec les gestionnaires d'incident que par la simple utilisation des propriétés `bugtraq:`, TortoiseSVN peut utiliser des plugins COM. Avec ces plugins, il est possible de récupérer des informations directement à partir du gestionnaire d'incidents, d'interagir avec l'utilisateur et de fournir des informations en retour à TortoiseSVN à propos des incidents en cours, de vérifier les messages log entrés par l'utilisateur et même d'exécuter des actions après une livraison réussie comme, par exemple, clore un incident.

We can't provide information and tutorials on how you have to implement a COM object in your preferred programming language, but we have example plugins in C++/ATL and C# in our repository in the `contrib/issue-tracker-plugins` folder. In that folder you can also find the required include files you need to build your plugin. (Section 3, « Licence » explains how to access the repository.)

Important

Vous devez fournir à la fois une version 32-bit et 64-bit de votre plugin. Parce que la version x64 de TortoiseSVN ne peut pas utiliser un plugin 32 bits et vice-versa.

7.1. Conventions de nommage

If you release an issue tracker plugin for TortoiseSVN, please do *not* name it *Tortoise<Something>*. We'd like to reserve the *Tortoise* prefix for a version control client integrated into the windows shell. For example: TortoiseCVS, TortoiseSVN, TortoiseHg, TortoiseGit and TortoiseBzr are all version control clients.

Please name your plugin for a Tortoise client *Turtle<Something>*, where *<Something>* refers to the issue tracker that you are connecting to. Alternatively choose a name that sounds like *Turtle* but has a different first letter. Nice examples are:

- Gurtle - Un plugin de gestion d'anomalie pour Google code
- TurtleMine - Un plugin de gestion d'anomalie pour Redmine
- VurtleOne - Un plugin de gestion d'anomalie pour VersionOne

7.2. L'interface de IBugtraqProvider

TortoiseSVN 1.5 et supérieur 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 gestionnaire d'incidents.

```
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 appelée depuis la fenêtre des paramètres là où l'utilisateur peut ajouter et configurer le greffon. La chaîne `parameters` peut être utilisée par un greffon pour récupérer davantage d'informations nécessaires, par exemple, l'URL du gestionnaire d'incidents, les informations de connexion, etc. Le greffon doit vérifier la chaîne `parametres` et montrer une fenêtre d'erreur si la chaîne n'est pas valide. Le paramètre `hParentWnd` permet au plugin d'accéder à la fenêtre mère. Le greffon doit renvoyer TRUE si la chaîne `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 chaîne ici, laquelle sera utilisée dans la fenêtre de livraison de TortoiseSVN par le bouton qui appelle le greffon, par exemple, "Choisir l'incident" ou "Sélectionner le ticket". Assurez vous que la chaîne 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 (
  // Fenêtre principale de l'interface utilisateur de votre FAI.
  [in] HWND hParentWnd,

  // Paramètres pour votre FAI.
  [in] BSTR parameters,
  [in] BSTR commonRoot,
  [in] SAFEARRAY(BSTR) pathList,

  // The texte déjà présent dans le message de livraison.
  // Votre FAI doit inclure ce texte dans le nouveau message,
  // où c'est appropré.
  [in] BSTR originalMessage,

  // Le nouveau texte pour le message de livraison.
  // Ceci remplace le message original.
  [out, retval] BSTR *newMessage
);
```

C'est la méthode principale du plugin. Cette méthode est appelée depuis la boîte de dialogue de livraison de TortoiseSVN lorsque l'utilisateur clique sur le bouton de plugin.

La chaîne `paramètres` est la chaîne que l'utilisateur doit entrer dans la boîte de dialogue de configuration quand il configure le plugin. Habituellement, un plugin utilise ceci pour trouver l'URL du gestionnaire d'incident et / ou des informations de connexion ou autre.

La chaîne `commonRoot` contient le chemin parent de tous les éléments sélectionnés pour faire apparaître la boîte de dialogue de livraison. Notez que ce *n'est pas* le chemin racine de tous les éléments que l'utilisateur a sélectionné dans la boîte de dialogue de livraison. Pour la boîte de dialogue branche/tag, c'est le chemin qui doit être copié.

Le paramètre `pathListliteral> contient un tableau de chemins (sous forme de cha`

Le paramètre `originalMessage` contient le texte entré dans la boîte de message de log, dans la boîte de dialogue de livraison. Si l'utilisateur n'a encore entré aucun texte, cette chaîne sera vide.

La chaîne de retour `newMessageliteral> est copioriginalMessage, il doit la laisser, sinon tout texte que l'utilisateur a saisi sera perdu.`

7.3. 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 (
  // Fenêtre principale de l'interface utilisateur de votre FAI.
  [in] HWND hParentWnd,

  // Paramètres pour votre FAI.
  [in] BSTR parameters,
  // L'URL commune de livraison
  [in] BSTR commonURL,
  [in] BSTR commonRoot,
  [in] SAFEARRAY(BSTR) pathList,

  // Le texte déjà présent dans le message de livraison.
  // Votre FAI doit inclure ce texte dans le nouveau message,
  // où c'est approprié.
  [in] BSTR originalMessage,

  // Vous pouvez assigner des propriétés de révision personnalisées à la livraison
  // en configurant les deux prochains paramètres.
  // note: les deux tableaux doivent avoir la même longueur.
  //       Chaque propriété doit avoir une valeur!

  // Le contenu du champ bugID (si affiché)
  [in] BSTR bugID,

  // Le contenu modifié du champ bugID
  [out] BSTR * bugIDOut,

  // La liste des noms de propriété de révision.
  [out] SAFEARRAY(BSTR) * revPropNames,

  // La liste des valeurs de propriété de révision.
  [out] SAFEARRAY(BSTR) * revPropValues,

  // Le nouveau texte du message de livraison.
  // Ceci remplace le message original
  [out, retval] BSTR * newMessage
);
```

Cette méthode est appelée depuis la boîte de dialogue de livraison de TortoiseSVN lorsque l'utilisateur clique sur le bouton de plugin. Cette méthode est appelée à la place de `GetCommitMessage()`. Veuillez vous référer à la documentation de `GetCommitMessage` pour les paramètres utilisés.

Le paramètre `commonURL` est l'URL mère de tous les éléments sélectionnés pour faire apparaître la boîte de dialogue de livraison. Il s'agit essentiellement de l'URL du chemin `commonRoot`.

Le paramètre `bugID` correspond au contenu du champ bug-ID (s'il est montré, configuré avec la propriété `bugtraq:message`).

Le paramètre de retour `bugIDOutliteral> est utilis`

Les paramètres de retour `revPropNames` et `revPropValues` peuvent contenir des paires nom/valeur pour les propriétés de révision que la livraison devrait avoir. Un plugin doit faire en sorte que les deux tableaux ont la même taille en retour! Chaque nom de propriété dans `revPropNames` doit aussi avoir une valeur correspondante

dans `revPropValues`. Si aucune des propriétés de révision ne sont à paramétrer, le plugin doit retourner des tableaux vides.

```
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
);
```

Cette méthode est appelée juste avant que la boîte de dialogue de livraison ne soit fermée et que la livraison ne commence. Un plugin peut utiliser cette méthode pour valider les fichiers/dossiers sélectionnés pour la livraison et/ou le message de livraison entré par l'utilisateur. Les paramètres sont les mêmes que pour `GetCommitMessage2()`, à la différence que `commonURL` est l'URL commune de tous les éléments *cochés*, et `commonRoot` le chemin racine de tous les éléments cochés.

Pour la boîte de dialogue de branche/tag, `commonURL` est l'URL source de la copie, et `commonRoot` est attribué à l'URL cible de la copie.

Le paramètre de retour `errorMessage` doit contenir soit un message d'erreur que TortoiseSVN montre à l'utilisateur soit être vide pour que la livraison débute. Si un message d'erreur est retourné, TortoiseSVN montre la chaîne d'erreur dans une boîte de dialogue et maintient la boîte de dialogue de livraison ouverte afin que l'utilisateur puisse corriger tout ce qui est mauvais. Un plugin doit donc retourner une chaîne d'erreur qui informe l'utilisateur de *ce quiemphasis> est faux et comment le corriger*.

```
HRESULT  OnCommitFinished (
  // Fen^tre principale pour n'importe quelle interface utilisateur (d'erreur) qui doit
  [in] HWND hParentWnd,

  // La racine commune de tous les chemins qui ont été livrés.
  [in] BSTR commonRoot,

  // Tous les chemins qui ont été livrés.
  [in] SAFEARRAY(BSTR) pathList,


  // Le texte déjà présent dans le message de livraison.
  [in] BSTR logMessage,

  // La révision de la livraison.
  [in] ULONG revision,


  // Une erreur à montrer à l'utilisateur si cette fonction
  // renvoie autre chose que S_OK
  [out, retval] BSTR * error
);
```

Cette méthode est appelée après une livraison réussie. Un plugin peut utiliser cette méthode pour, par exemple, clore l'incident sélectionné ou ajouter des informations à l'incident au sujet de la livraison . Les paramètres sont les mêmes que pour `GetCommitMessage2`.

```
HRESULT HasOptions(
```

```
  // Si le fournisseur fournit des options
  [out, retval] VARIANT_BOOL *ret
);
```

Cette méthode est appelée à partir de la boîte de dialogue de configuration où l'utilisateur peut configurer les plugins. Si un plugin fournit sa propre boîte de dialogue de configuration avec `ShowOptionsDialog`, il doit retourner TRUE, sinon il doit retourner FALSE.

```
HRESULT ShowOptionsDialog(
  // Fenêtre principale pour la boîte de dialogue des options
  [in] HWND hParentWnd,

  // Paramètres pour votre FAI.
  [in] BSTR parameters,

  // La chaîne des paramètres
  [out, retval] BSTR * newparameters
);
```

Cette méthode est appelée à partir de la boîte de dialogue des paramètres lorsque l'utilisateur clique sur le bouton "Options " qui est affiché si `HasOptions` retourne TRUE. Un plugin peut afficher une boîte de dialogue d'options pour le rendre plus facile à configurer pour l'utilisateur.

La chaîne `paramètres` contient la chaîne de paramètres du plugin qui est déjà configuré/entré.

Le paramètre de retour `newparameters` doit contenir la chaîne des paramètres que le greffon a construit à partir des informations recueillies dans sa boîte de dialogue Options. Cette chaîne `paramameters` est passée à toutes les autres méthodes 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* [https://tortoisesvn.net/faq.html] which contains a selection of the questions we are asked the most on the TortoiseSVN mailing lists https://groups.google.com/forum/#!forum/tortoisesvn and https://groups.google.com/forum/#!forum/tortoisesvn-dev

Si vous avez une question à laquelle vous n'avez pas trouvé de réponse, le meilleur endroit pour la poser est la mailing list:

- https://groups.google.com/forum/#!forum/tortoisesvn is the one to use if you have questions about using TortoiseSVN.
- If you want to help out with the development of TortoiseSVN, then you should take part in discussions on https://groups.google.com/forum/#!forum/tortoisesvn-dev

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.

Figure B.1. The TortoiseSVN right drag context menu for moving files

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 dépôt, vous pouvez installer un script hook pre-commit qui rejette toutes les livraisons avec des commentaires vides ou trop courts.

Dans le dossier du dépôt sur le serveur, il y a un sous-dossier `hooks` qui contient quelques exemples de scripts hook que vous pouvez utiliser. Le fichier `pre-commit.tmpl` contient un script type qui rejettera les livraisons si aucun commentaire n'est fourni, ou si le commentaire est trop court. Le fichier contient aussi des commentaires sur la façon d'installer/utiliser ce script. Suivez juste les instructions de ce fichier.

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 4.18, « Configuration des projets ».

B.3. Mettre à jour les fichiers sélectionnés à partir du dépôt

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 dépôt pour voir ce qui a changé dans le dépôt. 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 dépôt

B.4.1. Utiliser la boîte de dialogue du journal de révision

De loin le moyen le plus simple de faire revenir des modifications à une ou plusieurs révisions est d'utiliser la fenêtre du journal des révisions.

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. If you want to pick out individual revisions and ranges, use the **Ctrl** key while selecting revisions. Right click on the selected revision(s), then select Context Menu → Revert changes from this revision.
4. Ou si vous voulez faire d'une révision précédente la nouvelle révision HEAD, faites un clic droit sur les révisions sélectionnées, puis sélectionnez Menu contextuel → Revenir à cette révision. Cela annulera *tous* les changements après la révision choisie.

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

If you want to enter revision numbers as a list, you can use the Merge dialog. The previous method uses merging behind the scenes; this method uses it explicitly.

1. Dans votre copie de travail, sélectionnez TortoiseSVN → Fusionner.
2. In the Merge Type dialog select Merge a range of revisions.
3. In the From: field enter the full repository URL of your working copy folder. This should come up as the default URL.
4. In the Revision range to merge field enter the list of revisions to roll back (or use the log dialog to select them as described above).
5. Make sure the Reverse merge checkbox is checked.
6. In the Merge options dialog accept the defaults.
7. Cliquez sur Fusionner pour terminer la fusion.

You have reverted the changes within your working copy. Check that the results are as expected, then commit the changes.

B.4.3. Utiliser `svndumpfilter`

Puisque TortoiseSVN ne perd jamais de données, vos révisions « annulées » existent toujours comme révisions intermédiaires dans le dépôt. Seule la révision HEAD a été changée à un état précédent. Si vous voulez faire que les révisions disparaissent complètement de votre dépôt, 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 dépôt 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/1.8/svn.reposadmin.maint.html].

B.5. Compare deux révisions d'un fichier ou d'un répertoire

Si vous voulez comparer deux révisions dans l'historique d'un fichier, par exemple les révisions 100 et 200 du même fichier, utilisez simplement TortoiseSVN → Voir le journal pour lister l'historique des révisions pour ce fichier. Choisissez les deux révisions que vous voulez comparer puis utilisez Menu contextuel → Comparer les révisions.

Si vous voulez comparer le même fichier dans deux arborescences différentes, par exemple dans la version trunk et dans une branche, vous pouvez utiliser l'explorateur de dépôt pour ouvrir les deux arborescences, sélectionnez le fichier dans les deux emplacements, puis utilisez Menu contextuel → Comparer les révisions.

Si vous voulez comparer deux arborescences pour voir ce qui a changé, par exemple le tronc et une version tagguée, vous pouvez utiliser TortoiseSVN → Graphique de révision. Sélectionnez les deux noeuds à comparer, utilisez ensuite Menu contextuel → Comparer les révisions HEAD. Cela montrera une liste des fichiers modifiés et vous pouvez alors choisir des fichiers individuelements pour voir les changements en détail. Vous pouvez également exporter une arborescence contenant tous les fichiers modifiés ou simplement une liste de tous les fichiers modifiés. Lisez Section 4.11.3, « Comparer des dossiers » pour plus d'information. Autrement utilisez Menu contextuel → Comparaison unifiée des révisions HEAD pour voir un résumé de toutes les différences, avec un contexte minimal.

B.6. Inclure un sous-projet commun

Sometimes you will want to include another project within your working copy, perhaps some library code. There are at least 4 ways of dealing with this.

B.6.1. Utiliser svn:externals

Définit la propriété `svn:externals` sur un dossier de votre projet. Cette propriété consiste en une ou plusieurs lignes; chaque ligne comporte le nom d'un sous-dossier que vous voulez utiliser comme dossier d'extraction pour du code commun et l'URL du dépôt que vous voulez extraire là. Pour des détails plus complets, référez-vous à Section 4.19, « Éléments externes ».

Livre le nouveau dossier. Maintenant quand vous mettrez à jour, Subversion récupérera une copie de ce projet de son dépôt vers votre copie de travail. Les sous-dossiers seront créés automatiquement au besoin. Chaque fois que vous mettrez à jour votre copie de travail principale, vous recevrez aussi la dernière version de tous les projets externes.

Si le projet externe est dans le même dépôt, 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 dépôt différent, les changements que vous faites au projet externe seront signalés quand vous livrerez le projet principal, mais vous devrez livrer ces changements externes séparément.

Parmi les trois méthodes décrites, c'est la seule qui n'a pas besoin d'installation côté client. Une fois que les projets externes sont spécifiés dans les propriétés du dossier, tous les clients recevront les dossiers remplis lors de la mise à jour.

B.6.2. Utiliser une copie de travail nichée

Créez un nouveau dossier dans votre projet qui contiendra 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

If you use the same common core code in several projects, and you do not want to keep multiple working copies of it for every project that uses it, you can just check it out to a separate location which is related to all the other projects which use it. For example:

```
C:\Projects\Proj1
C:\Projects\Proj2
C:\Projects\Proj3
C:\Projects\Common
```

and refer to the common code using a relative path, e.g. `..\..\Common\DSPcore`.

If your projects are scattered in unrelated locations you can use a variant of this, which is to put the common code in one location and use drive letter substitution to map that location to something you can hard code in your projects, e.g. Checkout the common code to `D:\Documents\Framework` or `C:\Documents and Settings\{login}\My Documents\framework` then use

```
SUBST X: "D:\Documents\framework"
```

to create the drive mapping used in your source code. Your code can then use absolute locations.

```
#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.6.4. Ajouter le projet au référentiel

The maybe easiest way is to simply add the project in a subfolder to your own project working copy. However this has the disadvantage that you have to update and upgrade this external project manually.

To help with the upgrade, TortoiseSVN provides a command in the explorer right-drag context menu. Simply right-drag the folder where you unzipped the new version of the external library to the folder in your working copy, and then select Context Menu → SVN Vendorbranch here. This will then copy the new files over to the target folder while automatically adding new files and removing files that aren't in the new version anymore.

B.7. Créer un raccourci vers un dépôt

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 archivé l'ajout, tout ce que vous avez à faire est d'utiliser TortoiseSVN → Annuler l'ajout... pour annuler l'ajout. Vous devriez ensuite ajouter le(s) fichier(s) à la liste des fichiers ignorés pour ne pas le(s) ajouter à nouveau par inadvertance.

Si les fichiers sont déjà dans le dépôt, ils doivent être supprimés du dépôt et ajoutés à la liste des éléments ignorés. Heureusement TortoiseSVN a un raccourci commode pour ce faire. TortoiseSVN → Retirer la version et ajouter à la liste des ignorés marquera d'abord le fichier/dossier pour la suppression sur le dépôt, en gardant la copie locale. Il ajoute également cet élément à la liste des éléments ignorés de sorte qu'il ne sera pas ajouté de nouveau dans Subversion par erreur. Une fois ceci fait vous avez juste besoin de livrer le dossier parent.

B.9. Retirer une copie de travail du contrôle de version

If you have a working copy which you want to convert back to a plain folder tree without the `.svn` directory, you can simply export it to itself. Read Section 4.27.1, « Retirer une copie de travail du contrôle de version » to find out how.

B.10. Retirer une copie de travail

Si vous avez une copie de travail dont vous n'avez plus besoin, comment voulez-vous en débarrasser proprement? Facile - il suffit de supprimer dans l'explorateur Windows! Les copies de travail sont privées des entités locales, et ils sont autonomes. Suppression d'une copie de travail dans l'Explorateur Windows n'affecte pas les données dans le référentiel du tout.

Annexe C. Astuces 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.

Un bon guide sur la façon de faire se trouvé dans l'article 314934 de la base de connaissances de Microsoft : http://support.microsoft.com/?kbid=314934.

TortoiseSVN must be installed under *Computer Configuration* and not under *User Configuration*. This is because TortoiseSVN needs the 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 12 from Microsoft on each computer you want to install TortoiseSVN as per user.

You can customize the MSI file if you wish so that all your users end up with the same settings. TSVN settings are stored in the registry under `HKEY_CURRENT_USER\Software\TortoiseSVN` and general Subversion settings (which affect all Subversion clients) are stored in config files under `%APPDATA%\Subversion`. If you need help with MSI customization, try one of the MSI transform forums or search the web for « MSI transform ».

C.2. Rediriger la vérification de mise à niveau

TortoiseSVN vérifie s'il existe une nouvelle version disponible régulièrement. Si une telle version existe, une notification est affichée dans la boîte de dialogue de livraison.

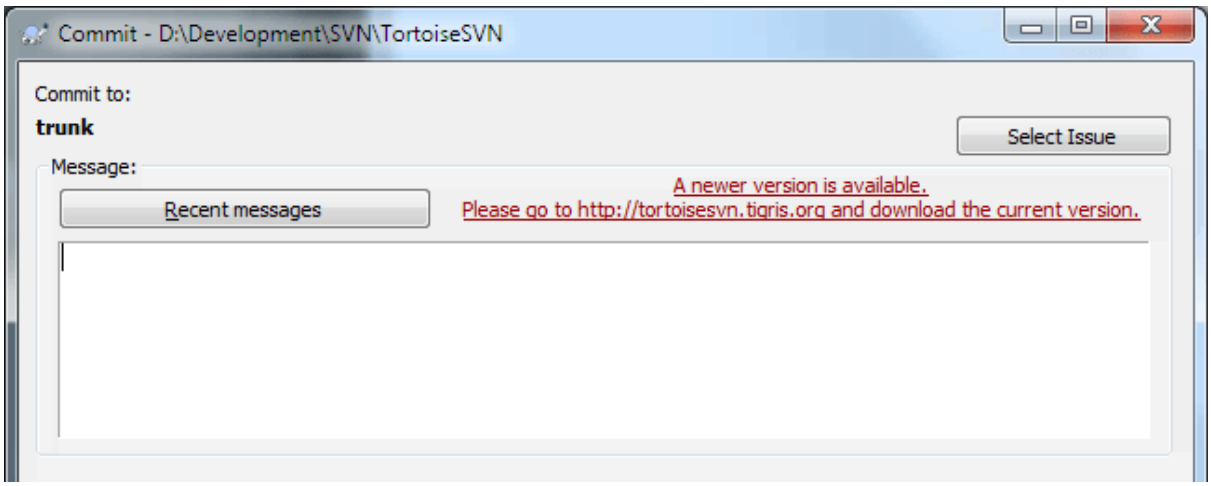


Figure C.1. La boîte de dialogue de livraison, montrant la notification de mise à jour

Si vous êtes responsable de beaucoup d'utilisateurs sur votre domaine, vous pouvez souhaitez que vous utilisateurs n'utilisent que des versions que vous avez approuvées et pas toujours la dernière version en date. Vous ne souhaitez probablement pas que cette notification s'affiche pour que vos utilisateur n'effectuent pas la mise à jour 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:

```
1.9.1.6000
A new version of TortoiseSVN is available for you to download!
http://192.168.2.1/downloads/TortoiseSVN-1.9.1.6000-svn-1.9.1.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 commit dialog. You can write there whatever you want. Just note that the space in the commit 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 commit 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. Définir 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 `SVN_ASP_DOT_NET_HACK`, puisque cela a causé beaucoup de problèmes et de confusion pour les utilisateurs qui installent toujours *tout*, sans forcément savoir à quoi cela sert.

But the feature is still available in TortoiseSVN and other svn clients. To enable it you have to set the Windows environment variable named `ASPDOTNETHACK` to `1`. Actually, the value of that environment variable doesn't matter: if the variable exists the feature is active.

Important

Veuillez noter que cette modification est uniquement nécessaire si vous utilisez toujours VS.NET2002. Toutes les versions ultérieures de Visual Studio ne requièrement *pas* l'activation de cette modification ! Donc à moins que vous n'utilisiez ce vieil outil, N'UTILISEZ PAS CETTE MODIFICATION !

C.4. Désactiver les entrées du menu contextuel

Depuis la version 1.5.0 et ultérieure, TortoiseSVN vous permet de désactiver (en fait, masquer) des entrées du menu contextuel. Comme il s'agit d'une caractéristique qui ne doit pas être utilisé à la légère, mais seulement s'il y a une raison impérieuse, il n'y a pas d'interface graphique pour cela et ça doit être fait directement dans le Registre. Cela peut être utilisé pour désactiver certaines commandes pour les utilisateurs qui ne devraient pas les utiliser. Mais veuillez noter que les entrées du menu contextuel sont seulement cachées dans *l'explorateur*, et que les commandes sont toujours disponibles par d'autres moyens, par exemple la ligne de commande ou même d'autres boîtes de dialogue dans TortoiseSVN lui-même!

Les clés de registre qui contiennent les informations pour savoir quels menus contextuels afficher sont `HKEY_CURRENT_USER\Software\TortoiseSVN\ContextMenuEntriesMaskLow` et `HKEY_CURRENT_USER\Software\TortoiseSVN\ContextMenuEntriesMaskHigh`.

Chacune de ces entrées de Registre est une valeur `DWORD`, chaque bit correspondant à une entrée de menu spécifique. Un bit activé signifie que l'entrée de menu correspondante est désactivée.

Valeur	Entrée du menu
0x0000000000000001	Extraire
0x0000000000000002	Mettre à jour
0x0000000000000004	Livrer
0x0000000000000008	Ajouter
0x0000000000000010	Revenir en arrière
0x0000000000000020	Nettoyer
0x0000000000000040	Résoudre

Valeur	Entrée du menu
0x0000000000000080	Aller sur...
0x0000000000000100	Importer
0x0000000000000200	Exporter
0x0000000000000400	Créer un dépôt 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 dépôt
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
0x0000000100000000	Fusionner Tous
0x0000000200000000	Différences avec la version précédente
0x0000000400000000	Coller
0x0000000800000000	Mettre à jour la copie de travail
0x0000001000000000	Comparer ultérieurement
0x0000002000000000	Diff with 'filename'
0x0000004000000000	Unified diff
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"`.

There's a special command that does not require the parameter `/command:abcd` but, if nothing is specified on the command line, starts the project monitor instead. If `/tray` is specified, the project monitor starts hidden and only adds its icon to the system tray.

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 `*`.

You can also specify a file which contains a list of paths, separated by newlines. The file must be in UTF-16 format, without a *BOM* [https://en.wikipedia.org/wiki/Byte-order_mark]. If you pass such a file, use `/pathfile` instead of `/path`. To have TortoiseProc delete that file after the command is finished, you can pass the parameter `/deletepathfile`. If you don't pass `/deletepathfile`, you have to delete the file yourself or the file gets left behind.

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.

Pour fermer la boîte de dialogue de progression automatiquement à la fin d'une commande sans utiliser le réglage permanent, vous pouvez lui passer le paramètre `/closeonend`.

- `/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

Pour fermer la boîte de dialogue de progression pour des opérations locales, s'il n'ya eu aucune erreur ni conflit, passez le paramètre `/closeforlocal`.

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. Additional options can be set:
	• `/startrev:xxx`,
	• `/endrev:xxx`,
	• `/limit:xxx` limits the amount of fetched messages
	• `/strict` enables the 'stop-on-copy' checkbox,
	• `/merge` enables the 'include merged revisions' checkbox,
	• `/datemin:"{datestring}"` sets the start date of the filter, and
	• `/datemax:"{datestring}"` sets the end date of the filter. The date format is the same as used for svn date revisions.
	• `/findstring:"filterstring"` fills in the filter text,
	• `/findtext` forces the filter to use text, not regex, or
	• `/findregex` forces the filter to use regex, not simple text search, and
	• `/findtype:X` with X being a number between 0 and 511. The numbers are the sum of the following options:
	  • `/findtype:0` tout filtrer
	  • `/findtype:1` filtrer par messages
	  • `/findtype:2` filtrer par chemin
	  • `/findtype:4` filtrer par auteurs
	  • `/findtype:8` filtrer par révision
	  • `/findtype:16` non utilisé
	  • `/findtype:32` filtre par ID de bug
	  • `/findtype:64` non utilisé
	  • `/findtype:128` filtre par date
	  • `/findtype:256` filtre par intervalle de date
	• If `/outfile:path\to\file` is specified, the selected revisions are written to that file when the log dialog is closed. The revisions are written in the same format as is used to specify revisions in the merge dialog.
	An svn date revision can be in one of the following formats:
	• {2006-02-17}

Commande	Description
	• {15:30} • {15:30:00.200000} • {"2006-02-17 15:30"} • {"2006-02-17 15:30 +0230"} • {2006-02-17T15:30} • {2006-02-17T15:30Z} • {2006-02-17T15:30-04:00} • {20060217T1530} • {20060217T1530Z} • {20060217T1530-0500}
:checkout	Opens the checkout dialog. The `/path` specifies the target directory and the `/url` specifies the URL to checkout from. If you specify the key `/blockpathadjustments`, the automatic checkout path adjustments are blocked. The `/revision:XXX` specifies the revision to check out. If you specify `/outfile:"path/to/file"` the specified file will contain three lines after a checkout. The first line is the checkout path, the second line the url and the third the revision.
:import	Opens the import dialog. The `/path` specifies the directory with the data to import. You can also specify the `/logmsg` switch to pass a predefined log message to the import dialog. Or, if you don't want to pass the log message on the command line, use `/logmsgfile:chemin`, where `chemin` points to a file containing the log message.
:update	Updates the working copy in `/path` to HEAD. If the option `/rev` is given then a dialog is shown to ask the user to which revision the update should go. To avoid the dialog specify a revision number `/rev:1234`. Other options are `/nonrecursive`, `/ignoreexternals` and `/includeexternals`. The `/stickydepth` indicates that the specified depth should be sticky, creating a sparse checkout. The `/skipprechecks` can be set to skip all checks that are done before an update. If this is specified, then the Voir le journal button is disabled, and the context menu to show diffs is also disabled after the update.
: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	Cleans up interrupted or aborted operations and unlocks the working copy in `/path`. You also have to pass the `/cleanup` to actually do the cleanup. Use `/noui` to prevent the result dialog from popping up (either telling about the cleanup being finished or showing an error message). `/noprogressui` also disables the

Commande	Description
	progress dialog. `/nodlg` disables showing the cleanup dialog where the user can choose what exactly should be done in the cleanup. The available actions can be specified with the options `/cleanup` for status cleanup, `/breaklocks` to break all locks, `/revert` to revert uncommitted changes, `/delunversioned`, `/delignored`, `/refreshshell`, `/externals`, `/fixtimestamps` and `/vacuum`.
: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 dépôt dans `/path`
:switch	Ouvre la fenêtre de dialogue . Le `/path` spécifie le répertoire cible et `/url`l'URL vers lequel se fera le changement.
:export	Exporter la copie de travail dans `/path` dans un autre répertoire. Si le `/path` pointe vers un répertoire non versionné, une boîte de dialogue vous demandera une URL à l'exportation vers le répertoire dans `/path` . Si vous spécifiez la clé `/blockpathadjustments` , Les ajustements automatiques du chemin d'exportation sont bloquées.
:dropexport	Exports the working copy in `/path` to the directory specified in `/droptarget`. This exporting does not use the export dialog but executes directly. The option `/overwrite` specifies that existing files are overwritten without user confirmation, and the option `/autorename` specifies that if files already exist, the exported files get automatically renamed to avoid overwriting them. The option `/extended` can specify either `modifications locales` to only export files that got changed locally, or `non versionnés` to also export all unversioned items as well.
:dropvendor	Copies the folder in `/path` recursively to the directory specified in `/droptarget`. New files are added automatically, and missing files get removed in the target working copy, basically ensuring that source and destination are exactly the same. Specify `/noui` to skip the confirmation dialog, and `/noprogressui` to also disable showing the progress dialog.
:merge	Opens the merge dialog. The `/path` specifies the target directory. For merging a revision range, the following options are available: `/fromurl:URL`, `/revrange:string`. For merging two repository trees, the following options are available: `/fromurl:URL`, `/tourl:URL`, `/fromrev:xxx` and `/torev:xxx`.
:mergeall	Ouvre la fenêtre fusionner tout. Le `/path` spécifie le répertoire cible.
:copy	Brings up the branch/tag dialog. The `/path` is the working copy to branch/tag from. And the `/url` is the target URL. If the urls starts with a ^ it is assumed to be relative to the repository root. To already check the option `Basculer la copie de travail vers une nouvelle branche/un nouveau marqueur` you can pass the `/switchaftercopy` switch. To check the option `Créer des dossiers intermédiaires` pass the `/makeparents` switch. You can also specify the `/logmsg` switch to pass a predefined log message to the branch/tag dialog. Or, if you don't want to pass the log message on the command line, use `/logmsgfile:chemin`, where `chemin` points to a file containing the log message.
: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`.

Commande	Description
: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. If the specified file also has property modifications, the external diff tool is also started for each modified property. To prevent that, pass the option `/ignoreprops`. To explicitly set the revision numbers use `/startrev:xxx` and `/endrev:xxx`, and for the optional peg revision use `/pegrevision:xxx`. If `/blame` is set and `/path2` is not set, then the diff is done by first blaming the files with the given revisions. The parameter `/line:xxx` specifies the line to jump to when the diff is shown.
:shelve	Shelves the specified paths in a new shelf. The option `/shelfname:name` specifies the name of the shelf. An optional log message can be specified with `/logmsg:message`. If option `/checkpoint` is passed, the modifications of the files are kept.
:unshelve	Applies the shelf with the name `/shelfname:name` to the working copy path. By default the last version of the shelf is applied, but you can specify a version with `/version:X`.
:showcompare	Selon les URL et les versions à comparer, ceci affiche soit un contenu unifié des différences (si l'option `unifié` est utilisée), soit une boîte de dialogue avec une liste de fichiers qui ont changé, soit, si les URL pointent vers les fichiers de démarrage, lance le visualisateur de différence pour ces deux fichiers. Les options `url1`, `url2`, `revision1` et `revision2` doivent être précisées. Les options `pegrevision`, `ignoreancestry`, `blame` et `unified` sont facultatives. If the specified url also has property modifications, the external diff tool is also started for each modified property. To prevent that, pass the option `/ignoreprops`. If a unified diff is requested, an optional `prettyprint` option can be specified which will show the merge-info properties in a more user readable format.
: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 boîte de dialogue de vérification des modifications. Le `/path` indique le répertoire de la copie de travail. Si `/remote` est spécifié, la boite de dialogue contacte le dépôt directement au démarrage, au moment où l'utilisateur clique sur le bouton Vérifier le dépôt.
:repobrowser	Starts the repository browser dialog, pointing to the URL of the working copy given in `/path` or `/path` points directly to an URL. Une option additionnelle `/rev:xxx` peut être utilisée pour spécifier la révision que l'explorateur de dépôt doit afficher. Si l'option `/rev:xxx` est omise, elle prend la valeur HEAD par défaut. Si `/path` pointe vers une URL, `/projectpropertiespath:chemin/vers/copie/de/travail` spécifie le chemin à partir duquel lire et utiliser les propriétés de projet.

Commande	Description
	If `/outfile:path\to\file` is specified, the selected URL and revision are written to that file when the repository browser is closed. The first line in that text file contains the URL, the second line the revision in text format.
: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. Les options `/ignoreeol`, `/ignorespaces` et `/ignoreallspaces` sont également supportées.
: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	Creates a patch file for the path given in `/path`. To skip the file Save-As dialog you can pass `/savepath:chemin` to specify the path where to save the patch file to directly. To prevent the unified diff viewer from being started showing the patch file, pass `/noview`. If a unified diff is requested, an optional `prettyprint` option can be specified which will show the merge-info properties in a more user readable format.
:revisiongraph	Montre le graphe de révision pour le chemin donné dans `/path`. To create an image file of the revision graph for a specific path, but without showing the graph window, pass `/output:path` with the path to the output file. The output file must have an extension that the revision graph can actually export to. These are: `.svg`, `.wmf`, `.png`, `.jpg`, `.bmp` and `.gif`. Since the revision graph has many options that affect how it is shown, you can also set the options to use when creating the output image file. Pass these options with `/options:XXXX`, where `XXXX` is a decimal value. The best way to find the required options is to start the revision graph the usual way, set all user-interface options and close the graph. Then the options you need to pass on the command line can be read from the registry `HKCU\Software\TortoiseSVN\RevisionGraphOptions`.
: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	Shows the properties dialog for the path given in `/path`. Pour traiter les propriétés versionnées, cette commande requiert une copie de travail. Revision properties can be viewed/changed if `/path` is an URL and `/rev:XXX` is specified.

Commande	Description
	To open the properties dialog directly for a specific property, pass the property name as `/property:name`.
:sync	Exports/imports settings, either depending on whether the current settings or the exported settings are newer, or as specified. If a path is passed with `/path`, then the path is used to store or read the settings from. The parameter `/askforpath` will show a file open/save dialog for the user to chose the export/import path. If neither `/load` nor `/save` is specified, then TortoiseSVN determines whether to export or import the settings by looking at which ones are more recent. If the export file is more recent than the current settings, then the settings are loaded from the file. If the current settings are more recent, then the settings are exported to the settings file. If `/load` is specified, the settings are imported from the settings file. Si `/save` est spécifié, alors les paramètres utilisés actuellement seront exportés dans le fichier de configuration. Le paramètre `/local` est spécifié, alors les paramètres utilisés actuellement seront exportés dans le fichier de configuration.

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:\svn_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. Tsvncmd URL handler

En utilisant des URL spéciales, il est également possible d'appeler TortoiseProc à partir d'une page web.

TortoiseSVN enregistre un nouveau protocole `tsvncmd:` qui peut être utilisé pour créer des liens hypertexte qui exécutent des commandes TortoiseSVN. Les commandes et les paramètres sont les mêmes que lors de l'automatisation de TortoiseSVN en ligne de commande.

Le format de l'URL de `tsvncmd:` est le suivant :

`tsvncmd:command:cmd?parameter:paramvalue?parameter:paramvalue`

with `cmd` being one of the allowed commands, `parameter` being the name of a parameter like `path` or `revision`, and `paramvalue` being the value to use for that parameter. The list of parameters allowed depends on the command used.

Les commandes suivantes sont permises avec `tsvncmd:` URLs:

- `:update`
- `:commit`

- `:diff`
- `:repobrowser`
- `:checkout`
- `:export`
- `:blame`
- `:repostatus`
- `:revisiongraph`
- `:showcompare`
- `:log`
- `:properties`

Une exemple d'URL simple pourrait ressembler à ça :

```
<a href="tsvncmd:command:update?path:c:\svn_wc?rev:1234"> Mise à jour </ a>
```

ou dans un cas plus compliqué :

```
<a href="tsvncmd:command:showcompare?
url1:https://svn.code.sf.net/p/stefanstools/code/trunk/StExBar/src/setup/Setup.wxs?
url2:https://svn.code.sf.net/p/stefanstools/code/trunk/StExBar/src/setup/Setup.wxs?
revision1:188?revision2:189">compare</a>
```

D.3. Commandes de TortoiseIDiff

L'outil de comparaison d'images a quelques options en ligne de commande que vous pouvez utiliser pour contrôler la façon dont l'outil démarre. Le programme est appelé `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	Si spécifié, l'outil de comparaison d'images bascule vers le mode d'icône de recouvrement (alpha).
:fit	Si spécifié, l'outil de différenciation d'image fait concorder les deux images ensemble.
: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
```

D.4. Commandes TortoiseUDiff

The unified diff viewer has only two command line options:

Option	Description
:patchfile	Chemin vers le fichier diff unifié.
:p	Active le mode tuyau. Le diff unifié est lu depuis l'entrée de la console.

Tableau D.3. Liste des options disponibles

Examples (which should be entered on one line):

```
TortoiseUDiff.exe /patchfile:"c:\diff.patch"
```

If you create the diff from another command, you can use TortoiseUDiff to show that diff directly:

```
svn diff | TortoiseUDiff.exe /u
```

this also works if you omit the `/p` parameter:

```
svn diff | TortoiseUDiff.exe
```

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 incidents de TortoiseSVN de ceux de Subversion. Cette référence vous dit quelle commande essayer.

E.1. Conventions et règles de base

In the descriptions which follow, the URL for a repository location is shown simply as `URL`, and an example might be `https://svn.code.sf.net/p/tortoisesvn/code/trunk/`. The working copy path is shown simply as `PATH`, and an example might be `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 [-depth ARG] [--ignore-externals] [-r rev] URL PATH
```

Les éléments de profondeur de la zone de liste déroulante se rapportent à l'argument `-depth`.

Si Omettre les références 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_of_WC
svn update [-r rev] PATH
```

Mettre à jour plusieurs éléments n'est pas actuellement une opération atomique dans Subversion. Donc TortoiseSVN trouve d'abord la révision HEAD du dépôt 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 dépôt, 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_of_WC
svn update [-r rev] [-depth ARG] [--ignore-externals] PATH
```

Les éléments de profondeur de la zone de liste déroulante se rapportent à l'argument `-depth`.

Si Omettre les références 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 PATH
```

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 PATH...
```

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 "LogMessage" [-depth ARG] [--no-unlock] PATH...
```

`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 PATH
```

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] PATH
or
svn log -v -r M:N [--stop-on-copy] PATH
```

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 dépôt.

Si Arrêt à la copie/renommage est coché, utilisez le commutateur `--stop-on-copy`.

E.2.7. Vérifier les modifications

```
svn status -v PATH
or
svn status -u -v PATH
```

La vérification initiale du statut ne regarde que votre copie de travail. Si vous cliquez sur Vérifier le dépôt alors le dépôt 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 dépôt et analyse ensuite les données renvoyées.

E.2.9. Explorateur de dépôt

```
svn info URL_of_WC
svn list [-r rev] -v URL
```

Vous pouvez utiliser `svn info` pour déterminer la racine du dépôt, qui est le niveau supérieur affiché dans l'explorateur de dépôt. 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 dépôt.

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 PATH
```

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 PATH
```

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 dépôt. Il y a 3 boutons radio d'options :

- Révision HEAD dans le dépôt
- Révision spécifique dans le dépôt
- 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 références externes est coché, utilisez le commutateur `--ignore-externals`.

E.2.22. Relocaliser

```
svn switch --relocate URL_Depuis URL_Vers
```

E.2.23. Créer un dépôt ici

```
svnadmin create --fs-type fsfs PATH
```

E.2.24. Ajouter

```
svn add PATH...
```

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. Icônes superposées

Chaque fichier et dossier a une valeur de statut Subversion tel que rapporté par la bibliothèque de Subversion. Dans le client de ligne de commande, elles sont représentées par des codes d'une seule lettre, mais dans TortoiseSVN elles sont représentées graphiquement en utilisant les icônes d'avant-plan. Parce que le nombre d'avant-plan est très limité, chacun d'eux peut représenter une ou plusieurs valeurs de statut.

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 dépôt. 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 dépôt 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.

The *Needs Lock* overlay is used to indicate when a file has the `svn:needs-lock` property set.

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.

Un élément ne peut avoir qu'une seule valeur de statut Subversion. Par exemple, un fichier peut être modifié localement et il pourrait être marqué pour suppression dans le même temps. Subversion renvoie une valeur de statut unique - dans ce cas `supprimé`. Ces priorités sont définies au sein de Subversion lui-même.

Lorsque TortoiseSVN affiche le statut récursivement (le réglage par défaut), sur chaque dossier est affichée une icône reflétant son propre statut et celui de tous ses enfants. Pour afficher une icône qui *résumeemphasis> l'ensemble, nous utilisons l'ordre de prioriten conflit prenant la priorit*

In fact, you may find that not all of these icons are used on your system. This is because the number of overlays allowed by Windows is limited to 15. Windows uses 4 of those, and the remaining 11 can be used by other applications. If there are not enough overlay slots available, TortoiseSVN tries to be a *Good Citizen (TM)* and limits its use of overlays to give other apps a chance.

Since there are Tortoise clients available for other version control systems, we've created a shared component which is responsible for showing the overlay icons. The technical details are not important here, all you need to know is that this shared component allows all Tortoise clients to use the same overlays and therefore the limit of 11 available slots isn't used up by installing more than one Tortoise client. Of course there's one small drawback: all Tortoise clients use the same overlay icons, so you can't figure out by the overlay icons what version control system a working copy is using.

- `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.
- *Locked* is loaded if possible, but falls back to *Normal* if there are not enough slots.
- *Ajouté* est chargé si possible, mais retombe à *Mis à jour* s'il n'y a pas assez d'emplacements.

Annexe G. Paquetages linguistiques et correcteurs orthographiques

Le programme d'installation standard prend uniquement l'Anglais en charge, mais vous pouvez télécharger des packs de langue et des dictionnaires séparément après l'installation.

G.1. Packs de langue

The TortoiseSVN user interface has been translated into many different languages, so you may be able to download a language pack to suit your needs. You can find the language packs on our *translation status page* [https://tortoisesvn.net/translation_status_dev.html]. And if there is no language pack available, why not join the team and submit your own translation ;-)

Chaque pack de langue est empaqueté comme un installeur `.msi`. Exécutez juste le programme d'installation et suivez les instructions. Une fois l'installation terminée, la traduction sera disponible.

The documentation has also been translated into several different languages. You can download translated manuals from the *support page* [https://tortoisesvn.net/support.html] on our website.

G.2. Vérificateur d'orthographe

TortoiseSVN uses the Windows spell checker if it's available (Windows 8 or later). Which means that if you want the spell checker to work in a different language than the default OS language, you have to install the spell checker module in the Windows settings (`Settings > Time & Language > Region & Language`).

TortoiseSVN will use that spell checker if properly configured with the `tsvn:projectlanguage` project property.

In case the Windows spell checker is not available, TortoiseSVN can also use spell checker dictionaries from *OpenOffice* [https://openoffice.org] and *Mozilla* [https://mozilla.org].

L'installeur ajoute automatiquement les dictionnaires d'anglais américain et britannique. 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. Une fois l'installation terminée, le dictionnaire sera disponible aussi.

Or you can install the dictionaries yourself. If you have OpenOffice or Mozilla installed, you can copy those dictionaries, which are located in the installation folders for those applications. Otherwise, you need to download the required dictionary files from http://wiki.services.openoffice.org/wiki/Dictionaries.

Once you have got the dictionary files, you probably need to rename them so that the filenames only have the locale chars in it. Example:

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

Then just copy them into the `%APPDATA%\TortoiseSVN\dic` folder. If that folder isn't there, you have to create it first. TortoiseSVN will also search the `Languages` sub-folder of the TortoiseSVN installation folder (normally this will be `C:\Program Files\TortoiseSVN\Languages`); this is the place where the language packs put their files. However, the %APPDATA%-folder doesn't require administrator privileges and, thus, has higher priority. The next time you start TortoiseSVN, the spell checker will be available.

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

1. Vérifier le réglage `tsvn:projectlanguage`. Référez-vous à Section 4.18, « 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 le dialecte exact Windows ne fonctionne pas, essayez la langue « de base », e.g. `de_CH` (Allemand-Suisse) revient à `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.

Glossaire

Ajouter	Une commande Subversion utilisée pour ajouter un fichier ou un répertoire à votre copie de travail. Les nouveaux éléments sont ajoutés au dépôt 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 dépôt. 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 dépôt 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.
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 dépôt 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 dépôt et vous remettez vos modifications dans le dépôt en faisant une « Livraison ».
Copier	Dans un dépôt 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.
Dépôt	Un dépôt est un endroit central où sont stockées et entretenues les données. Un dépôt peut être un endroit où se trouvent plusieurs bases de données ou des fichiers pour la distribution sur un réseau, ou un dépôt peut être un emplacement directement accessible à l'utilisateur sans devoir traverser de réseaux.
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 dépôt.

FSFS
: Un système de gestion de fichier de Subversion pour dépôts. Peut être utilisé pour les partages réseaux. Utilisé par défaut pour les dépôts à partir de la version 1.2.

Fusionner
: Le procédé par lequel les modifications du dépôt 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
: Objet de la politique 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 dépôt 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 dépôt, en créant une nouvelle révision du dépôt.

Mettre à jour
: Cette commande Subversion récupère les derniers changements depuis le dépôt 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 dépôt.

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 dépôt 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.
Relocaliser	Si votre dépôt 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 dépôt 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 dépôt, mais le dépôt 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 dépôt. Chaque révision représente l'état de l'arborescence du dépôt à un certain point de son histoire. Si vous voulez revenir dans le temps, vous pouvez examiner le dépôt 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 *dépôt*.
Supprimer	Quand vous supprimez un élément versionné (et que vous livrez le changement), l'élément n'existe plus dans le dépôt après la révision livrée. Mais il existe bien sûr toujours dans les révisions précédentes du dépôt, 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 dépôt « svnserve ».
Verrouiller	Quand vous retirez un verrou sur un élément versionné, vous le marquez comme non livrable dans le dépôt, 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

W