TortoiseSVN

Клиент Subversion для Windows

Версия 1.6.6

Stefan Küng
Lübbe Onken
Simon Large

TortoiseSVN: Клиент Subversion для Windows: Версия 1.6.6

Stefan Küng, Lübbe Onken, Simon Large
Перевод: Vladimir Serdyuk (vserd@users.sourceforge.net), Станислав Петраков (stannic@gmail.com)

Опубликовано 2009/08/31 01:13:53 (r17052)

Содержание

Список иллюстраций

Список таблиц

Предисловие

- Вы работаете в команде?
- Случалось ли так, что вы работали с файлом, и кто-то ещё работал с ним же и в то же время? И не теряли ли вы внесённых вами в файл изменений из-за этого?
- Бывало ли так, что после сохранения файла вам хотелось отменить только что сделанные изменения? И не хотелось ли вам узнать, как файл выглядел некоторое время назад?
- Не желали ли вы узнать, когда именно появилась ошибка, обнаруженная в вашем проекте?

Если вы ответили «да» хотя бы на один из этих вопросов, тогда TortoiseSVN предназначена для вас! Продолжайте чтение, и вы узнаете, как TortoiseSVN может помочь вам в работе. Всё это не так сложно, как кажется.

1. Кому адресована эта книга

Эта книга написана для тех, кто, владея компьютерной грамотой, хочет использовать Subversion для управления своими данными, но чувствует себя неуютно, применяя для этого клиента командной строки. Поскольку TortoiseSVN - расширение оболочки Windows, предполагается, что пользователь знаком с Проводником Windows и знает, как его использовать.

2. Структура книги

Это Предисловие рассказывает немного о проекте TortoiseSVN, о сообществе участвующих в нём людей, условиях лицензирования для использования и распространения.

Глава 1, *Введение* рассказывает, что представляет собой TortoiseSVN, его возможности, историю возникновения и основы его установки на ваш ПК.

В главе Глава 2, *Основные понятия управления версиями* мы даём краткое введение в систему управления версиями *Subversion*, лежащую в основе TortoiseSVN. Оно позаимствовано из документации проекта Subversion и объясняет различные подходы к управлению версиями, и то, как работает Subversion.

В главе о Глава 3, *Хранилище* рассказывается о том, как создать локальное хранилище, полезное для проверки Subversion и TortoiseSVN в рамках одного компьютера. В ней также немного рассказывается об администрировании хранилища, что также относится и к хранилищам, расположенным на сервере. Здесь также есть раздел о том, как установить и настроить сервер, если он вам необходим.

Глава 4, *Руководство по ежедневному использованию* является наиболее важным разделом, поскольку описывает все основные возможности TortoiseSVN и способы их использования. Оно представлено в виде учебного пособия, которое начинает с создания рабочей копии, её изменения, фиксации изменений и т.д., а дальше переходит к более сложным вопросам.

Глава 5, *Программа SubWCRev* - это отдельная программа, идущая вместе с TortoiseSVN, которая может извлекать информацию из вашей рабочей копии и записывать её в файл. Она пригодится для включения данных о сборке в ваши проекты.

Приложение Приложение B, *Как я могу...* отвечает на некоторые общие вопросы о решении задач, которые не освещены детально в каком-нибудь другом месте.

Раздел Приложение D, *Автоматизация TortoiseSVN* показывает, как диалоговые окна TortoiseSVN могут быть вызваны из командной строки. Это будет полезно при написании сценариев, в которых, тем не менее, необходимо взаимодействие с пользователем.

Приложение E, *Справочник соответствия с интерфейсом командной строки* показывает, как соотносятся команды TortoiseSVN и их эквиваленты в клиенте командной строки Subversion `svn.exe`.

3. TortoiseSVN бесплатен!

TortoiseSVN бесплатен. Вам не нужно платить за его использование, и вы можете применять его любым удобным вам способом. TortoiseSVN разработан под лицензией GNU General Public License (GPL).

TortoiseSVN - проект с открытым исходным кодом (Open Source). Это означает, что вы имеете полный доступ на чтение к исходному коду этой программы. Вы можете просмотреть его по ссылке *http://tortoisesvn.tigris.org/svn/tortoisesvn/*. У вас будут запрошены имя пользователя и пароль. В качестве имени пользователя введите `guest`, пароль нужно оставить пустым. Самая последняя версия (над которой мы работаем в данный момент) находится в `/trunk/`, ранее выпущенные версии находятся в `/tags/`.

4. Сообщество

Обе программы: и TortoiseSVN, и Subversion, разработаны сообществом людей, работающих в этих проектах. Это люди из разных стран со всего света, и они объединились для создания замечательных программ.

5. Благодарности

Тиму Кемпу (Tim Kemp)
: за основание проекта TortoiseSVN

Стефану Кунгу (Stefan Küng)
: за тяжёлый труд по реализации того, чем TortoiseSVN является сейчас

Люббе Онкену (Lübbe Onken)
: за прекрасные значки, логотипы, отлов ошибок, за перевод и координацию деятельности по переводу

Саймону Ладжу (Simon Large)
: за работу над документацией и отлов ошибок

Книге о Subversion (The Subversion Book)
: за прекрасное введение в Subversion и главу 2, которую мы сюда скопировали

Проекту Tigris Style (The Tigris Style project)
: за некоторые стили, использованные в этой документации

Нашим помощникам
: за исправления, сообщения об ошибках, новые идеи и за помощь, оказанную другим - в виде ответов на вопросы в нашем списке рассылки

Нашим дарителям
: за многие часы удовольствия от присланной нам музыки

6. Используемая терминология

Для облегчения чтения документации, имена всех экранов и меню TortoiseSVN выделены другим шрифтом. Например, Диалог журнала ревизий.

Выбор меню обозначен стрелкой. TortoiseSVN → Журнал означает: выберите *Журнал* из контекстного меню *TortoiseSVN*.

Использование локального контекстного меню какого-либо из диалоговых окон TortoiseSVN будет показываться следующим образом: Контекстное меню → Сохранить как...

Кнопки пользовательского интерфейса обозначаются так: Нажмите ОК для продолжения.

Действия пользователя показаны при помощи полужирного шрифта. **Alt+A**: нажмите клавишу **Alt** на вашей клавиатуре и, удерживая её нажатой, нажмите клавишу **A**. Перетаскивание правой кнопкой: нажмите правую кнопку мыши и, удерживая её нажатой, *перетащите* элементы в новое место.

Вывод системы и клавиатурный ввод также показан при помощи `отличающегося` шрифта.

Важно

Важные примечания отмечены значком.

Подсказка

Подсказки, делающие вашу жизнь проще.

Предостережение

Места, где надо быть осмотрительнее в том, что вы делаете.

Внимание

Необходимо проявить исключительную осторожность, возможно повреждение данных или другие неприятности при игнорировании этих предупреждений.

Глава 1. Введение

Управление версиями - это искусство управления изменениями информации. Этот инструмент давно стал критически важным для программистов, обычно тратящих свое время на создание небольших изменений в программе, некоторые из которых надо на другой день убрать или проверить. А теперь вообразите команду таких программистов, работающих одновременно, да ещё и над одними и теми же файлами! - и вы сможете понять, зачем нужна хорошая система для *управления потенциальным хаосом*.

1.1. Что такое TortoiseSVN?

TortoiseSVN - это бесплатный, с открытыми исходными кодами клиент системы управления версиями *Subversion*. Это означает, что TortoiseSVN управляет файлами и папками во времени. Файлы хранятся в центральном *хранилище*, которое очень похоже на обычный файловый сервер, за исключением того, что в нём запоминается каждое изменение, сделанное в ваших файлах и папках. Это позволяет восстанавливать старые версии файлов, и изучать историю того, как и когда изменялись ваши данные, и кем это делалось. Поэтому многие считают Subversion и системы управления версиями вообще своеобразными «машинами времени».

Некоторые системы контроля версий являются также и системами управления конфигурацией программ (software configuration management - SCM). Такие системы специально созданы для управления деревьями исходного кода, и имеют множество возможностей, специфичных для разработки программ, таких как непосредственное понимание языков программирования, или предоставление инструментов для сборки программ. Однако Subversion не является такой системой, она является системой общего назначения, которая может быть использована для управления *любым* набором файлов, включая и исходные коды программ.

1.2. История TortoiseSVN

В 2002 году Тим Кемп (Tim Kemp) обнаружил, что Subversion - очень хорошая система управления версиями, но ей не хватает хорошего клиента с графическим интерфейсом. Идея реализации клиента Subversion как расширения оболочки Windows была навеяна похожим клиентом для системы CVS, TortoiseCVS.

Тим изучил исходники TortoiseCVS и взял их за основу для TortoiseSVN. Он начал проект, зарегистрировал домен `tortoisesvn.org` и опубликовал исходный код. В это время Стефан Кунг (Stefan Küng) искал хорошую и бесплатную систему управления версиями, и обнаружил Subversion и исходный код TortoiseSVN. Поскольку TortoiseSVN всё ещё было невозможно использовать, он присоединился к проекту и начал программировать. Вскоре он переписал большинство существующего кода и начал добавлять команды и новые возможности, пока ничего из первоначального кода не осталось.

Со временем Subversion становилась всё более стабильной и привлекала всё больше и больше пользователей, которые начинали использовать TortoiseSVN для доступа к Subversion. Число пользователей быстро росло (и продолжает расти каждый день). Именно тогда Люббе Онкен (Lübbe Onken) предложил помощь в создании некоторых симпатичных значков и логотипа для TortoiseSVN. Он также взял на себя заботу о веб-сайте и стал заведовать переводами.

1.3. Возможности TortoiseSVN

Что делает TortoiseSVN таким хорошим клиентом Subversion? Вот краткий список возможностей:

Интеграция с оболочкой

TortoiseSVN интегрируется непосредственно в оболочку Windows (т.е. в Проводник). Это значит, что вы можете работать с уже знакомыми инструментами, и вам не надо переключаться на другое приложение каждый раз, когда вам необходимы функции для управления версиями!

И вам даже не обязательно использовать именно Проводник. Контекстные меню TortoiseSVN работают во многих других файловых менеджерах, и в диалогах для открытия файлов, используемых в большинстве стандартных Windows-приложений. Однако вы должны учитывать, что TortoiseSVN изначально разработан как расширение для Проводника Windows, и, возможно, в других приложениях интеграция будет не полной, например, могут не отображаться пометки на значках.

Пометки на значках
: Статус каждого версированного файла и папки отображается при помощи маленькой пометки поверх основного значка. Таким образом, вы сразу можете видеть состояние вашей рабочей копии.

Простой доступ к командам Subversion
: Все команды Subversion доступны из контекстного меню Проводника. TortoiseSVN добавляет туда собственное подменю.

Поскольку TortoiseSVN является клиентом Subversion, мы хотели бы показать и некоторые из возможностей самой Subversion:

Версирование папок
: CVS отслеживает только историю отдельных файлов, тогда как Subversion реализует «виртуальную» версионную файловую систему, которая отслеживает изменения в целых деревьях папок во времени. Файлы *и* папки являются версированными. В результате, есть команды **переместить** и **копировать**, реально выполняемые на стороне клиента и работающие непосредственно с файлами и папками.

Атомарные фиксации
: Фиксация сохраняется в хранилище либо полностью, либо не сохраняется вообще. Это позволяет разработчикам фиксировать изменения, собранные в логически связанные части.

Версированные метаданные
: Каждый файл и папка имеет прикреплённый невидимый набор «свойств». Вы можете создавать и сохранять произвольные пары ключ/значение для собственных нужд. Свойства тоже версируются во времени, как и содержимое файла.

Возможность выбора сетевого уровня
: В Subversion есть абстрагируемое понятие доступа к хранилищу, которое упрощает реализацию новых сетевых механизмов. «Усовершенствованный» сетевой сервер Subversion является модулем для веб-сервера Apache, который использует для взаимодействия диалект HTTP под названием WebDAV/DeltaV. Это даёт Subversion большие преимущества в стабильности и совместимости, и предоставляет различные ключевые возможности без дополнительных затрат: проверка личности (аутентификация), проверка прав доступа (авторизация), сжатие потока данных при передаче, просмотр хранилища. Также доступна меньшая, автономная версия сервера Subversion, взаимодействующая по собственному протоколу, который легко может быть туннелирован через ssh.

Единый способ обработки данных
: Subversion получает различия между файлами при помощи бинарного разностного алгоритма, который работает одинаково как с текстовыми (читаемыми человеком), так и с бинарными (не читаемыми человеком) файлами. Оба типа файлов содержатся в хранилище в сжатом виде, а различия передаются по сети в обоих направлениях.

Эффективные ветки и метки
: Стоимость создания веток и меток не обязательно должна быть пропорциональна размеру проекта. Subversion создаёт ветки и метки, просто копируя проект с использованием механизма, похожего на жёсткие ссылки в файловых системах. Благодаря этому, операции по созданию веток и меток происходят за одинаковое, очень малое время и занимают очень мало места в хранилище.

Расширяемость
: Subversion не имеет исторического багажа. Она реализована в виде набора совместно используемых библиотек на языке C с хорошо определёнными API. Это делает Subversion чрезвычайно удобной в сопровождении системой, пригодной для взаимодействия с другими приложениями и языками программирования.

1.4. Установка TortoiseSVN

1.4.1. Требования к системе

TortoiseSVN работает на Windows 2000 SP2, Windows XP или более поздней версии ОС. Начиная с TortoiseSVN 1.2.0, Windows 98, Windows ME и Windows NT4 больше не поддерживаются, но вы можете загрузить старые версии, если они вам действительно нужны.

Если вы обнаружите любую проблему во время или после установки TortoiseSVN, пожалуйста, сначала прочитайте Приложение A, *Часто задаваемые вопросы (ЧаВо, FAQ)*.

1.4.2. Установка

TortoiseSVN поставляется с простой в использовании программой установки. Запустите файл установщика и следуйте инструкциям, об остальном позаботится установщик.

Важно

У вас должны быть права администратора системы для установки TortoiseSVN.

1.4.3. Языковые пакеты

Интерфейс пользователя TortoiseSVN переведен на множество различных языков, так что есть возможность загрузить языковый пакет, который вам лучше подойдёт. Вы можете найти языковые пакеты на нашей *страничке состояния переводов* [http://tortoisesvn.net/translation_status]. И, если вашего языка нет в этом списке, вы могли бы присоединиться к команде и предложить свой перевод ;-)

Каждый языковой пакет упакован в `.exe` установщик. Просто запустите программу установки и следуйте инструкциям. Перевод станет доступен после перезагрузки.

1.4.4. Проверка правописания

TortoiseSVN содержит возможность проверки правописания, которая позволяет проверять ваши сообщения при фиксации. Это особенно полезно, если язык ведения проекта для вас не родной. Проверка правописания использует те же файлы словарей, которые используют *OpenOffice* [http://openoffice.org] и *Mozilla* [http://mozilla.org].

Установщик автоматически добавляет словари US и UK english (английский для США и английский для Великобритании). Если вам нужны другие языки, простейший путь - установить один из языковых пакетов TortoiseSVN. При этом будут установлены нужные файлы словарей и локализованный интерфейс пользователя TortoiseSVN. Словари будут доступны после перезагрузки.

Или вы можете установить словари самостоятельно. Если вы используете OpenOffice или Mozilla, вы можете скопировать эти словари из папок, в которых установлены эти программы. Иначе вам надо будет загрузить необходимые файлы словарей с *http://wiki.services.openoffice.org/wiki/Dictionaries*

После того, как у вас будут файлы словарей, возможно, вам понадобиться переименовать их так, чтобы имя файла содержало только символы, обозначающие язык и локализацию. Например:

- `en_US.aff`
- `en_US.dic`

Затем скопируйте их в подпапку `bin` установочной папки TortoiseSVN. Обычно это `C:\Program Files\TortoiseSVN\bin`. Если вы не желаете загромождать подпапку `bin`, вы можете поместить эти файлы в папку `C:\Program Files\TortoiseSVN\Languages`. Если эта папка отсутствует, необходимо сначала её создать. Когда вы в следующий раз запустите TortoiseSVN, будет доступна проверка правописания.

Если вы устанавливаете несколько словарей, TortoiseSVN использует следующие правила для выбора того, какой из них использовать.

1. Проверить параметр `tsvn:projectlanguage`, задающий язык проекта. Для информации об установке свойств проекта прочитайте Раздел 4.17, «Установки проекта».
2. Если язык проекта не задан, или этот язык не установлен, попробовать язык, соответствующий локализации Windows.
3. Если полное наименование локализации Windows не работает, попробовать «базовый» язык, например, вместо `de_CH` (Немецкий-Швейцария) применить `de_DE` (Немецкий).
4. Если ничего из этого не сработало, тогда язык по умолчанию - английский, включённый в стандартную установку.

Глава 2. Основные понятия управления версиями

Эта глава - слегка изменённая версия такой же главы из книги о Subversion. Размещённая в Сети версия книги о Subversion доступна по адресу *http://svnbook.red-bean.com/*.

Эта глава является кратким неформальным введением в Subversion. Если управление версиями для вас в новинку, эта глава определённо для вас. Мы начнём с обсуждения основных понятий управления версиями, перейдём к определённым идеям, лежащим в основе Subversion, и покажем несколько простых примеров использования Subversion.

Несмотря на то, что примеры этой главы показывают людей, совместно использующих набор исходных кодов программ, помните, что Subversion может управлять набором файлов любого типа, она не ограничена только тем, чтобы помогать в работе одним компьютерным программистам.

2.1. Хранилище

Subversion - это централизованная система для совместного использования информации. В её основе лежит *хранилище*, являющееся центром хранения данных. Хранилище хранит информацию в форме *дерева файловой системы* - типичной иерархии файлов и папок. Любое количество *клиентов* подключаются к хранилищу, а затем читают или записывают эти файлы. Записывая данные, клиент делает информацию доступной для остальных; читая данные, клиент получает информацию от других.

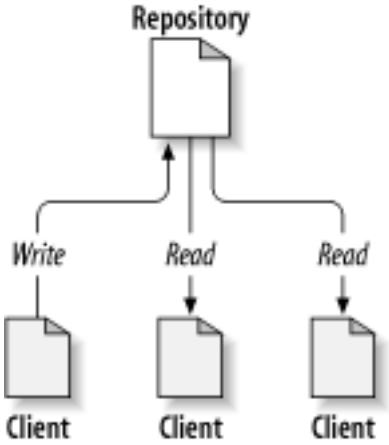


Рисунок 2.1. Типичная система Клиент/Сервер

Почему мы заостряем на этом внимание? Пока это звучит как определение типичного файл-сервера. И действительно, хранилище *является* разновидностью файл-сервера, однако не совсем обычного. Что делает хранилище Subversion особенным - это то, что он *запоминает каждое внесённое изменение*, когда-либо записанное в него: любое изменение любого файла, и даже изменения в самом дереве каталогов, такие как добавление, удаление и реорганизация файлов и каталогов.

Когда клиент читает данные из хранилища, он обычно видит только последнюю версию дерева файловой системы. Но клиент также имеет возможность просмотреть *предыдущие* состояния файловой системы. Например, клиент может запросить такие данные как, «Что содержал этот каталог в прошлую среду?» или «Кто последним изменял этот файл и какие изменения он произвёл?» Вопросы подобного типа являются основными для любой *системы управления*

версиями: системы, разработанной для записи и отслеживания изменений информации во времени.

2.2. Модели версирования

Всем системам управления версиями приходиться решать одну фундаментальную проблему: как система будет позволять пользователям совместно использовать информацию, не давая им при этом наступать друг другу на пятки? Для пользователей может оказаться чересчур легко нечаянно перезаписать в хранилище изменения друг друга.

2.2.1. Проблема совместного использования файлов

Рассмотрим такой сценарий: предположим, что у нас есть два сотрудника, Гарри и Салли. Каждый из них решил отредактировать один и тот же файл из хранилища в одно и то же время. Если Гарри сохранит свои изменения первым, тогда, возможно, Салли (несколькими секундами позже) может непреднамеренно перезаписать их своей новой версией файла. Несмотря на то, что версия Гарри не будет потеряна навсегда (т.к. система помнит каждую версию), внесённые Гарри изменения *не будут* отражены в новой версии файла Салли, потому что она никогда не видела изменений Гарри, которые могла бы учесть. Работа Гарри фактически потеряна - или, по крайней мере, отсутствует в последней версии файла, - и, вероятно, непредумышленно. Как раз та ситуация, которой мы и хотим избежать!

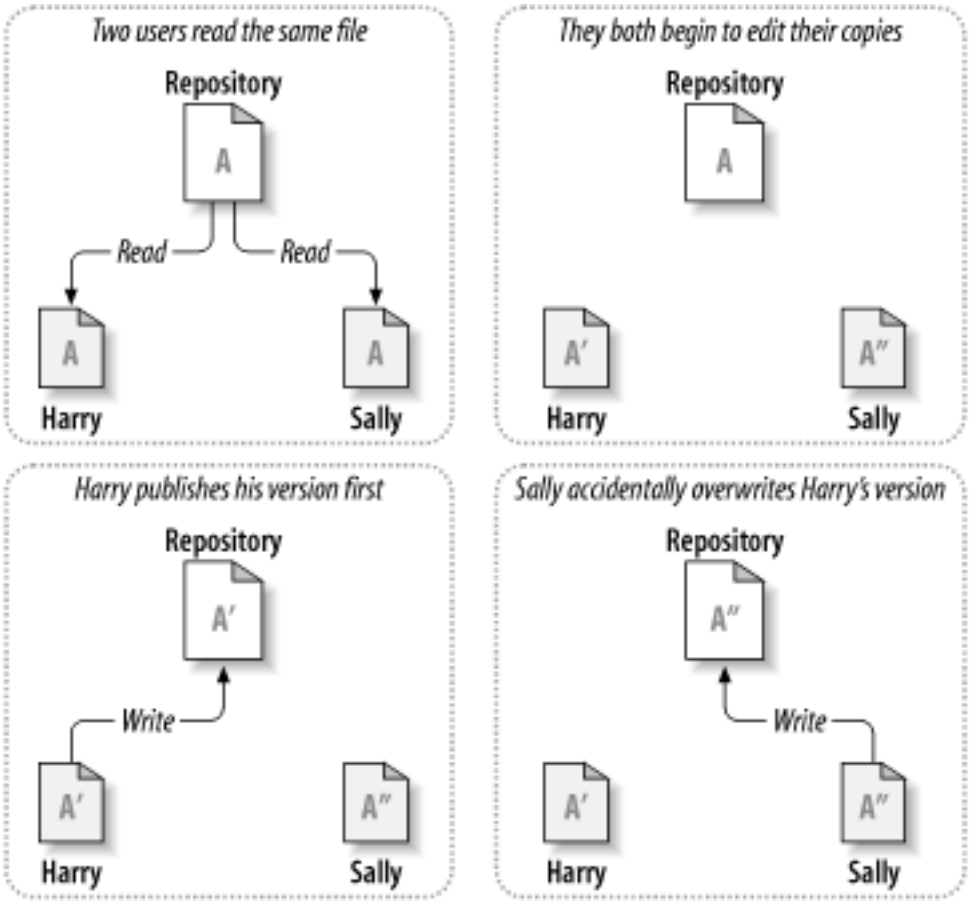


Рисунок 2.2. Проблема потери изменений

2.2.2. Модель Блокирование-Изменение-Разблокирование

Многие системы управления версиями используют для решения этой проблемы модель *блокирование-изменение-разблокирование*. В такой системе хранилище разрешает вносить изменения в файл только одному человеку за раз. До того, как Гарри сможет внести изменения

в файл, он должен сначала его *заблокировать*. Блокирование файла подобно взятию книги в библиотеке: если Гарри заблокировал файл, Салли не сможет сделать в нём никаких изменений. Хранилище отклонит её запрос, если она попытается заблокировать файл. Всё, что она может - читать файл и ждать, когда Гарри закончит свои изменения и снимет блокировку. После того, как Гарри разблокирует файл, его ход окончен, и теперь Салли, в свою очередь, сможет заблокировать и отредактировать.

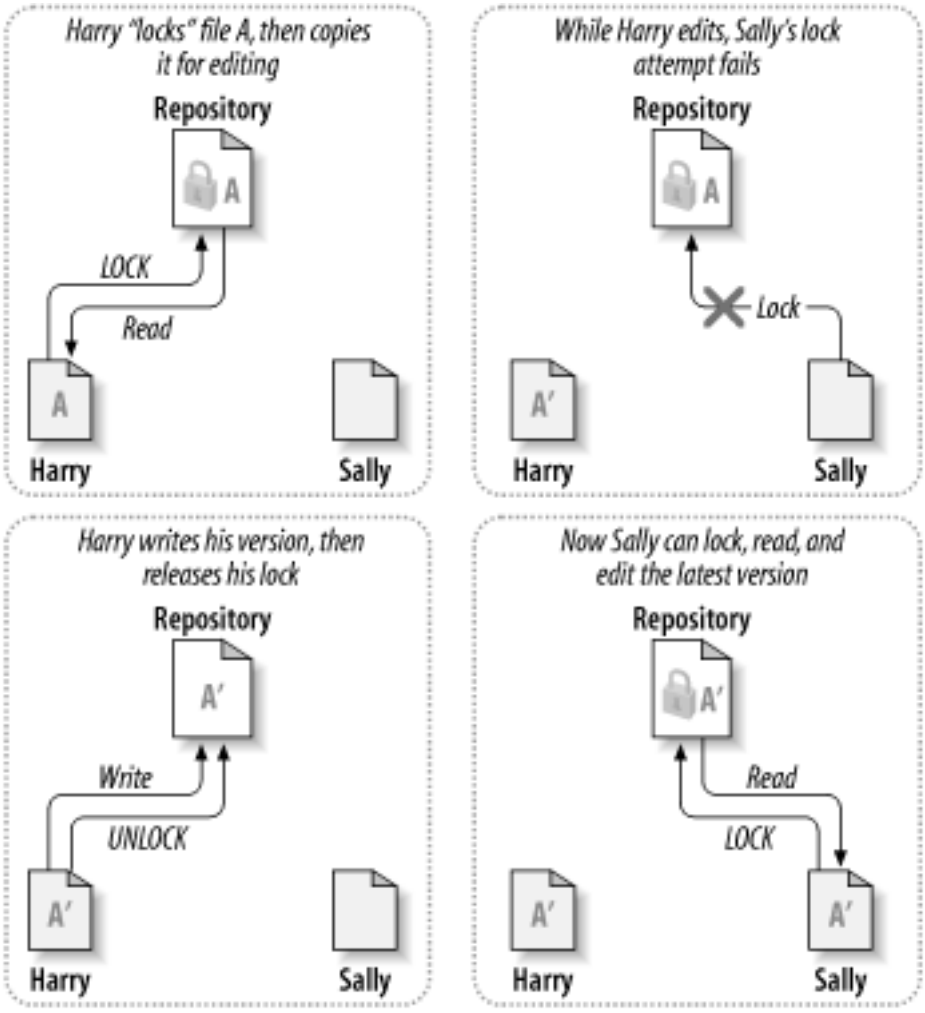


Рисунок 2.3. Модель Блокирование-Изменение-Разблокирование

Проблема с моделью блокирование-изменение-разблокирование состоит в том, что она накладывает некоторые ограничения и часто создаёт неудобства пользователям:

- *Блокирование может вызвать административные проблемы.* Иногда Гарри, заблокировав файл, забывает об этом. Между тем, поскольку Салли всё ещё ждёт, когда она сможет приступить к редактированию файла, её руки связаны. А потом Гарри уезжает в отпуск. Теперь Салли для снятия блокировки Гарри должна обратиться к администратору. Ситуация приводит к ненужной задержке и потере времени.

- *Блокирование может вызвать излишнюю поочерёдность.* Что, если Гарри редактирует начало текстового файла, а Салли хочет просто подправить окончание этого же файла? Эти изменения вообще не пересекаются. Они могли бы легко редактировать файл одновременно и никакого вреда это бы не принесло (предполагая корректное слияние изменений). В этой ситуации им не надо делать свои ходы по очереди.

- *Блокирование может вызвать ложное чувство безопасности.* Предположим, что Гарри заблокировал и редактирует файл A, в то время, как Салли заблокировала и редактирует файл B.

Но допустим, что A и B зависят друг от друга и сделанные в каждом изменения семантически не совместимы. Неожиданно A и B вместе больше не работают. Блокирующая система бессильна в предотвращении проблемы - вместо этого она обеспечила ложное чувство безопасности. Гарри и Салли запросто могут представить, что, блокируя файлы, каждый начинает безопасную изолированную задачу и поэтому это представление изначально препятствует обсуждению их несовместимых изменений.

2.2.3. Модель Копирование-Изменение-Слияние

Subversion, CVS и другие системы управления версиями используют модель *копирование-изменение-слияние* в качестве альтернативы блокированию. В этой модели клиент каждого пользователя считывает из хранилища проект и создаёт персональную *рабочую копию* - локальное отражение файлов и каталогов хранилища. После этого пользователи работают параллельно, изменяя свои личные копии. В конце концов, личные копии сливаются в новую, финальную версию. Обычно система управления версиями помогает в слиянии, но, разумеется, в конечном итоге за его корректное выполнение всё равно отвечает человек.

Вот пример: скажем, и Гарри, и Салли создали свои рабочие копии одного и того же проекта, скопировав их из хранилища. Они работают одновременно, и делают изменения в файле `A` в своих рабочих копиях. Первой свои изменения в хранилище сохраняет Салли. Затем, когда Гарри пытается сохранить свои изменения, хранилище информирует его, что его файл A *устарел*. Другими словами, файл A в хранилище был как-то изменён с тех пор, как Гарри получил его. Поэтому Гарри просит своего клиента *слить (merge)* любые изменения из хранилища с его рабочей копией файла A. Возможно, что изменения Салли не пересекаются с его собственными, и, поскольку теперь в его рабочей копии объединены оба набора изменений, он записывает её обратно в хранилище.

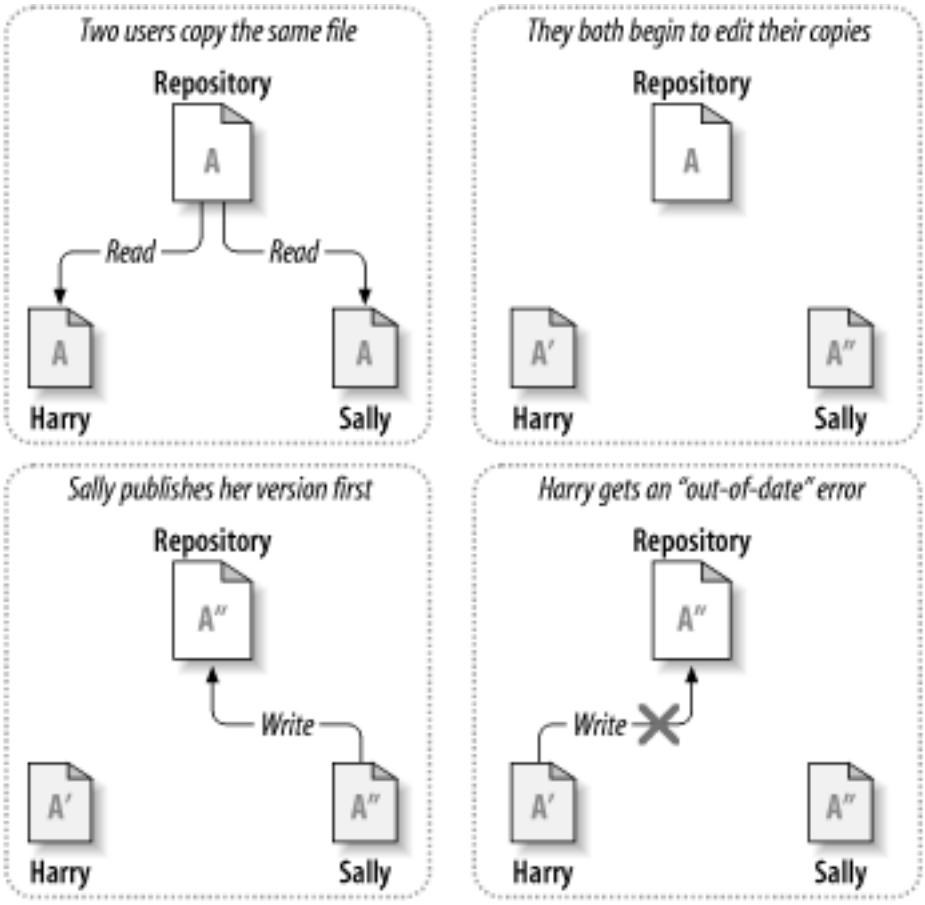


Рисунок 2.4. Модель Копирование-Изменение-Слияние

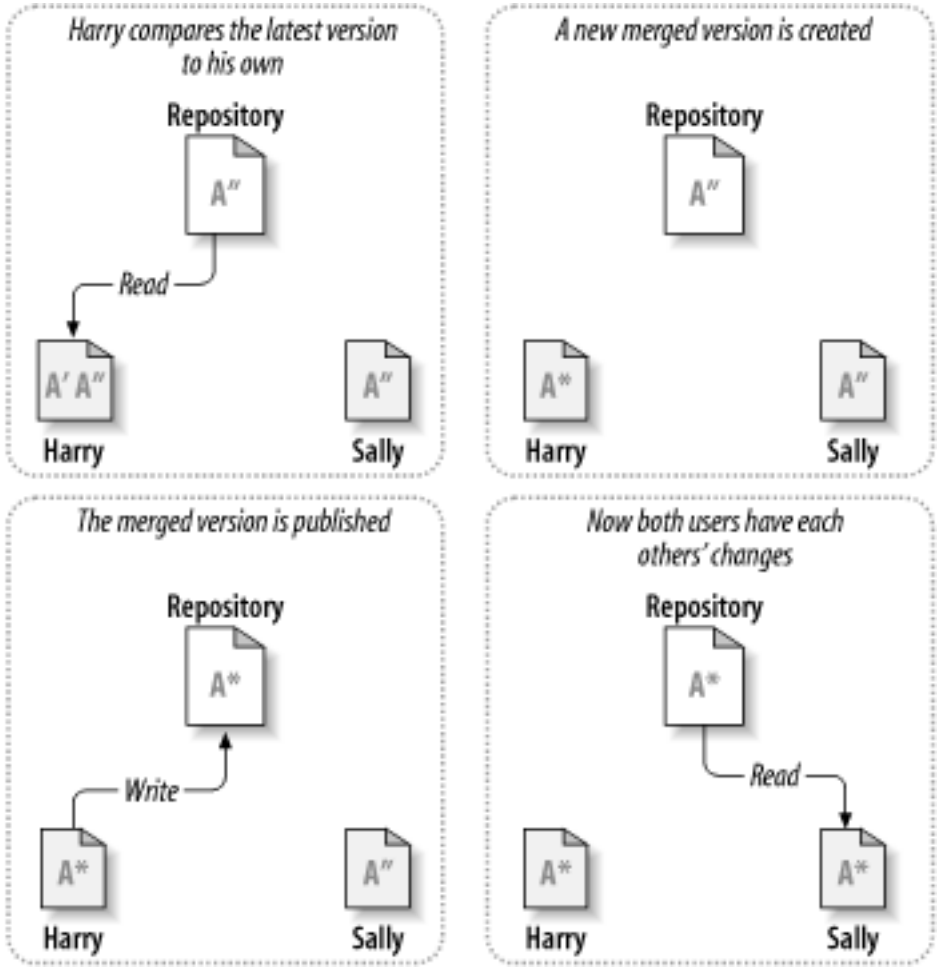


Рисунок 2.5. ...Копирование-Изменение-Слияние. Продолжение

Но что будет, если изменения Салли всё-таки *пересекаются* с изменениями Гарри? Что происходит в этом случае? Эта ситуация, называемая *конфликтом*, обычно не такая уж большая проблема. Когда Гарри просит объединить свои изменения с изменениями из хранилища, его копия файла A помечается как находящаяся в состоянии конфликта: он имеет возможность видеть оба набора конфликтующих изменений, и вручную выбирать между ними. Обратите внимание, программа не может автоматически разрешать конфликты, только человек способен понять и сделать необходимый осмысленный выбор. Когда Гарри вручную разрешил пересекающиеся изменения (возможно, путём их обсуждения с Салли!), он может безопасно сохранить объединённый файл обратно в хранилище.

Модель копирование-изменение-слияние может выглядеть немного хаотично, но на практике она отлично работает. Пользователи могут работать параллельно, никогда не ожидая друг друга. При работе над одними и теми же файлами обычно оказывается, что большинство одновременно вносимых изменений вообще не пересекаются; конфликты бывают редко. И время, потраченное на разрешение конфликтов, значительно меньше времени, отнимаемого системой с блокированием.

В конце концов, всё сводится к одному решающему фактору: взаимодействию пользователей. При плохом взаимодействии пользователей, увеличивается количество и семантических, и синтаксических конфликтов. Нет такой системы, которая сможет заставить пользователей общаться, и нет системы, которая может обнаруживать семантические конфликты. Не стоит успокаиваться ложным обещанием блокирующей системы как-то предотвращать конфликты; на практике, блокирование снижает производительность как ничто другое.

Однако, есть одна распространённая ситуация, когда модель блокирование-изменение-разблокирование оказывается лучше: если вы имеете дело с файлами, не поддающимися слиянию. Например, если ваше хранилище содержит некоторые графические изображения, и два человека изменяют их в одно и тоже время, то нет возможности слить эти изменения вместе. Всё равно, либо Гарри, либо Салли, потеряют свои изменения.

2.2.4. Что же делает Subversion?

По умолчанию Subversion использует модель копирование-изменение-слияние, и в большинстве случаев это всё, что вам нужно. Однако, начиная с версии 1.2, Subversion также поддерживает блокирование файлов, так что если у вас есть необъединяемые файлы, или если руководство просто вынудило вас работать в режиме с блокировками, Subversion сможет предоставить вам такую возможность.

2.3. Subversion в действии

2.3.1. Рабочие копии

Вы уже читали о рабочих копиях, сейчас мы покажем, как клиент Subversion их создаёт и использует.

Рабочая копия Subversion - это обычное дерево папок в вашей локальной системе, содержащее набор файлов. Вы можете изменять эти файлы по своему усмотрению, и, если это исходные коды, вы можете скомпилировать программу из них обычным способом. Ваша рабочая копия - это ваша собственная личная рабочая область: Subversion никогда не вносит изменения, сделанные другими, также как и не передаёт другим изменения, сделанные вами, до тех пор, пока вы сами явно не скажете ей сделать это.

После того, как вы произвели некоторые изменения в файлах вашей рабочей копии и убедились, что они работают правильно, вы можете воспользоваться представленными в Subversion командами для *публикации* ваших изменений, чтобы сделать их доступными другим людям, работающим вместе с вами над проектом (путём записи в хранилище). Когда другие люди публикуют свои изменения, Subversion предоставляет вам команды для слияния этих изменений с файлами в вашей рабочей папке (путём чтения из хранилища).

Рабочая копия также содержит несколько дополнительных файлов, создаваемых и обслуживаемых Subversion, помогающих при выполнении этих команд. В частности, каждая папка в вашей рабочей копии содержит папку с именем `.svn`, называемую также *административной папкой* рабочей копии. Файлы в каждой административной папке помогают Subversion распознавать, какие файлы содержат неопубликованные изменения, и какие файлы устарели из-за того, что было сделано другими.

Типичное хранилище Subversion часто содержит файлы (или исходный код) нескольких проектов, обычно каждый проект - это подпапка в дереве файловой системы хранилища. При таком подходе, пользовательская рабочая копия будет соответствовать какому-то поддереву в хранилище.

Например, предположим, что у вас есть хранилище с двумя программными проектами.

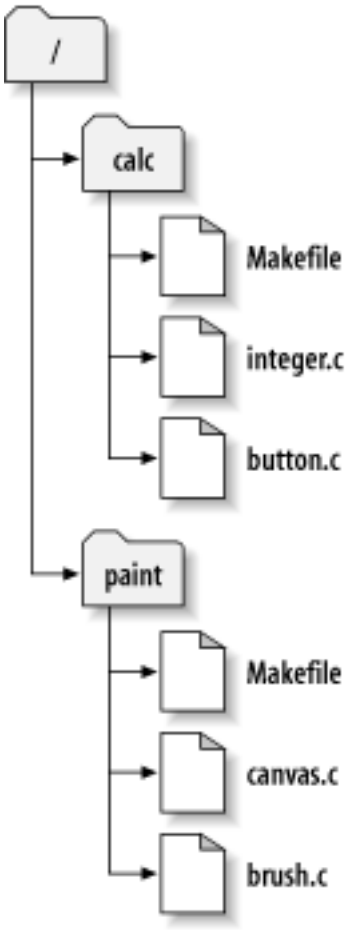


Рисунок 2.6. Файловая система хранилища

Другими словами, корневая папка хранилища содержит две папки: `paint` и `calc`.

Для получения рабочей копии, вы должны *извлечь* некоторое поддерево хранилища. (Термин *извлечение* (check out) по-английски может звучать как что-то, связанное с блокированием или резервированием ресурсов, но это не так: оно просто создаёт для вас личную копию проекта).

Предположим, вы вносите изменения в `button.c`. Так как в папке `.svn` запоминается дата модификации файла и исходное содержимое, Subversion может узнать, что вы изменили файл. Однако, Subversion не делает ваши изменения доступными другим, пока вы явно не скажете ей об этом. Действие по опубликованию ваших изменений, обычно известно как *фиксация* (или *внесение*) изменений в хранилище.

Для обнародования ваших изменений, вы должны использовать команду Subversion **фиксировать (commit)**.

Теперь ваши изменения в `button.c` были зафиксированы в хранилище; если другой пользователь извлечёт рабочую копию `/calc`, он увидит ваши изменения в последней версии файла.

Предположим, вы работаете вместе с Салли, которая извлекла рабочую копию `/calc` в то же время, что и вы. Когда вы фиксируете ваши изменения в `button.c`, рабочая копия Салли остаётся неизменной, Subversion изменяет рабочие копии только по запросу пользователя.

Для приведения своего проекта в актуальное состояние, Салли может попросить Subversion *обновить* её рабочую копию, используя команду **обновить (update)**. В результате, в её рабочую копию будут внесены как ваши изменения, так и изменения других, зафиксированные с момента извлечения Салли своей рабочей копии.

Обратите внимание, Салли не надо указывать, какие файлы обновлять, Subversion использует информацию в папке `.svn`, а затем информацию из хранилища, для определения того, какие файлы нуждаются в обновлении.

2.3.2. Адреса URL хранилища

Хранилища Subversion могут быть доступны посредством множества различных методов - с локального диска или через различные сетевые протоколы. Описание расположения хранилища, однако, всегда является разновидностью URL. Схема URL показывает метод доступа:

Схема	Метод доступа
`file://`	Прямой доступ к хранилищу на локальном или сетевом диске.
`http://`	Доступ через протокол WebDAV к Subversion, работающем на сервере Apache.
`https://`	Тоже самое, что и `http://`, но с шифрованием SSL
`svn://`	Не аутентифицируемый TCP/IP доступ через собственный протокол к серверу `svnserve`.
`svn+ssh://`	Аутентифицируемый, зашифрованный TCP/IP доступ через собственный протокол к серверу `svnserve`.

Таблица 2.1. URL для доступа к хранилищу

В большинстве случаев, для URL в Subversion используется стандартный синтаксис, позволяющий указывать имя сервера и номер порта в URL. Метод доступа `file://` обычно используется для локального доступа, хотя он может быть использован с путями UNC для доступа к узлам по сети. В этом случае URL имеет форму `file://имя-компьютера/путь/к/хранилищу`. Для локальной машины часть `имя-компьютера` должна быть либо пропущена, либо указана как `localhost`. По этой причине, локальные пути обычно указывают с тремя косыми чертами (/), `file:///путь/к/хранилищу`.

Также, пользователи схемы `file://` на платформе Windows вынуждены использовать неофициальный «стандарт» синтаксиса для доступа к хранилищам, находящимся на той же машине, но на дисках, отличных от текущего рабочего диска пользователя. Будет работать любой из двух приведённых ниже синтаксиса путей URL (`X` обозначает диск, на котором находится хранилище):

```
file:///X:/путь/к/хранилищу
...
file:///X|/путь/к/хранилищу
...
```

Обратите внимание, в URL используется обычная (прямая) косая черта, хотя в исходной (не URL) форме путей в Windows используется обратная косая черта.

Вы можете безопасно получить доступ к FSFS-хранилищу на разделяемом сетевом ресурсе, но вы *не можете* получить доступ к BDB-хранилищу таким образом.

Внимание

Не создавайте и не обращайтесь к хранилищу на основе Berkeley DB, которое находится на разделяемом сетевом ресурсе. Оно *не может* существовать в рамках удалённой файловой системы. И даже в том случае, если у вас назначена буква для сетевого диска. Если вы пытаетесь использовать Berkeley DB на разделяемом сетевом ресурсе, результаты непредсказуемы - вы можете увидеть загадочные ошибки

> сразу же, или могут пройти месяцы, прежде чем вы обнаружите, что база данных хранилища коварно испорчена.

2.3.3. Ревизии

Команда **svn commit** может опубликовать изменения в любом количестве файлов и папок как одну атомарную транзакцию. В вашей рабочей копии вы можете изменить содержимое файла, создать, удалить, переименовать и скопировать файлы и папки, а затем зафиксировать весь набор изменений как единое целое.

Каждая фиксация в хранилище обрабатывается как атомарная транзакция: либо сохраняются все изменения, либо не сохраняется ни одно из них. Subversion старается поддерживать эту атомарность, несмотря на сбои программ, аварийные отказы систем, на сетевые проблемы и действия других пользователей.

Каждый раз при выполнении фиксации в хранилище создаётся новое состояние дерева файловой системы, называемое *ревизией*. Каждой ревизии назначается уникальный целочисленный номер, на единицу больший, чем у предыдущей ревизии. Начальная ревизия в только что созданном хранилище имеет номер ноль, и не содержит ничего, кроме пустой корневой папки.

Удачный способ мысленного представления хранилища - представить его в виде серии деревьев. Вообразите массив номеров ревизий, начинающийся с 0, и растущий слева направо. Под каждым номером ревизии расположено дерево файловой системы, и каждое дерево - это «снимок» состояния хранилища после каждой фиксации.

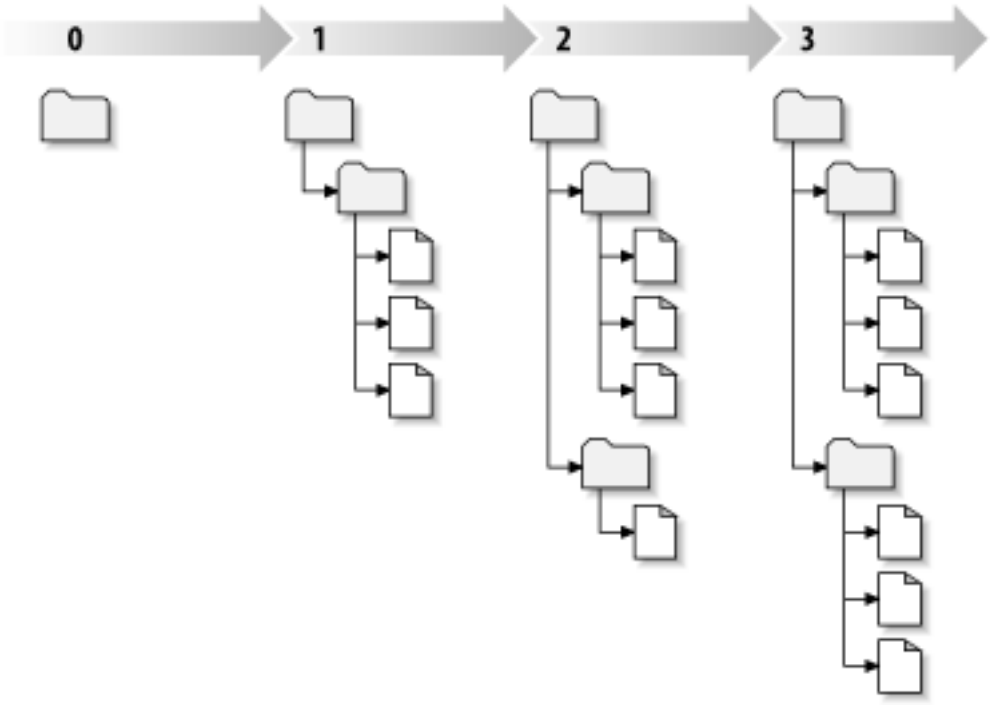


Рисунок 2.7. Хранилище

Общие номера ревизий

В отличие от многих других систем управления версиями, номера ревизий в Subversion относятся к *деревьям целиком*, а не к отдельным файлам. Каждый номер ревизии обозначает целое дерево - некоторое состояние хранилища после зафиксированного изменения. Иначе говоря, можно считать, что ревизия N представляет состояние файловой системы хранилища после выполнения N-ой фиксации. Когда пользователь Subversion говорит о "ревизии 5

`foo.c`", это в действительности означает "`foo.c`, каким он был в ревизии 5". Обратите внимание, что ревизии N и M файла, вообще говоря, *не обязательно* отличаются!

Важно помнить то, что рабочие копии не всегда соответствуют какой-то одной ревизии в хранилище; они могут содержать файлы из разных ревизий. Например, вы извлекли рабочую копию из хранилища, в котором самая последняя ревизия имеет номер 4:

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

На данный момент рабочая папка полностью соответствует ревизии 4 в хранилище. Допустим, что вы внесли изменения в `button.c`, и зафиксировали эти изменения. При отсутствии других фиксаций ваша фиксация создаст ревизию под номером 5, и теперь ваша рабочая копия выглядит следующим образом:

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

Предположим, что после этого Салли фиксирует изменения в `integer.c`, создавая ревизию 6. Если вы воспользуетесь **svn update** для приведения своей рабочей копии в актуальное состояние, то она станет выглядеть так:

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

Изменения, внесенные Салли в `integer.c`, будут отражены в вашей рабочей копии, также как и ваши изменения будут присутствовать в `button.c`. В этом примере текст `Makefile` в ревизиях 4, 5 и 6 идентичен, однако, Subversion всё равно присваивает файлу `Makefile` в вашей рабочей копии номер ревизии 6, чтобы показать, что файл в актуальном состоянии. Таким образом, после того как вы выполните полное обновление вашей рабочей копии, она будет соответствовать точно одной ревизии в хранилище.

2.3.4. Как рабочие копии отслеживают хранилище

В служебной папке `.svn/` для каждого файла рабочей папки Subversion записывает информацию о двух важнейших свойствах:

- на какой ревизии основан ваш рабочий файл (это называется *рабочая ревизия* файла), и
- дату и время, когда локальная копия последний раз обновлялась из хранилища.

Основываясь на этой информации, и взаимодействуя с хранилищем, Subversion может сказать, в каком из следующих четырех состояний находится рабочий файл:

Не изменялся и не устарел
: Файл не изменялся в рабочей папке, и в хранилище не фиксировались изменения этого файла со времени его рабочей ревизии. Команды **фиксировать (commit)** и **обновить (update)** ничего делать не будут.

Изменён локально и не устарел
: Файл был изменён в рабочей папке, и в хранилище не фиксировались изменения этого файла со времени его базовой ревизии. Существующие локальные изменения не были зафиксированы в хранилище, поэтому команда **фиксировать (commit)** для файла преуспеет в опубликовании ваших изменений, а команда **обновить (update)** ничего делать не будет.

Не изменялся и устарел
: Файл в рабочей папке не изменялся, но был изменён в хранилище. Со временем, файл должен быть обновлён для соответствия текущей публичной ревизии. Команда **фиксировать (commit)** ничего делать не будет, а команда **обновить (update)** внесёт последние изменения в вашу рабочую копию.

Изменён локально и устарел
: Файл был изменён как в рабочей папке, так и в хранилище. Команда **фиксировать (commit)** потерпит неудачу с ошибкой *устарел (out-of-date)*. Файл необходимо сначала обновить; команда **обновить (update)** попытается объединить опубликованные изменения с локальными. Если Subversion не сможет выполнить объединение в приемлемой форме самостоятельно, то заботу о разрешении конфликта она оставит пользователю.

2.4. Подводя итоги

В этой главе мы рассмотрели несколько основных понятий Subversion:

- Мы ввели понятия центрального хранилища, клиентской рабочей копии и массива соответствующих ревизиям деревьев в хранилище.
- Мы рассмотрели на нескольких простых примерах, как при помощи Subversion два сотрудника могут публиковать и получать изменения, сделанные друг другом, используя модель "копирование-изменение-слияние".
- Мы немного поговорили о том, как Subversion отслеживает изменения и управляет информацией в рабочей копии.

Глава 3. Хранилище

Каким бы протоколом вы ни пользовались для доступа к своим хранилищам, вам в любом случае потребуется создать хотя бы одно хранилище. Это может быть сделано как при помощи клиента Subversion для командной строки, так и при помощи TortoiseSVN.

Если вы ещё не создали хранилище Subversion, самое время этим заняться.

3.1. Создание хранилища

Вы можете создать хранилище или в формате FSFS, или в более старом формате Berkeley Database (BDB). Формат FSFS обычно быстрее и более лёгок в администрировании, также он сейчас без проблем работает на сетевых разделяемых ресурсах и в Windows 98. Формат BDB одно время рассматривался как более стабильный - просто потому, что использовался дольше, но, поскольку FSFS используется для реальной работы вот уже несколько лет, это соображение теперь не существенно. Прочтите *Choosing a Data Store (Выбор формата хранилища)* [http://svnbook.red-bean.com/en/1.5/svn.reposadmin.planning.html#svn.reposadmin.basics.backends] в Книге о Subversion для дополнительной информации.

3.1.1. Создание хранилища при помощи клиента командной строки

1. Создайте пустую папку с именем SVN, (например, `D:\SVN\`), которая будет корневой папкой для всех ваших хранилищ.
2. Создайте другую папку `MyNewRepository` внутри `D:\SVN\`.
3. Откройте командную строку (или окно эмуляции DOS), перейдите в `D:\SVN\` и введите

```
svnadmin create --fs-type bdb MyNewRepository
```

или

```
svnadmin create --fs-type fsfs MyNewRepository
```

Теперь у вас есть новое хранилище, расположенное в `D:\SVN\MyNewRepository`.

3.1.2. Создание хранилища при помощи TortoiseSVN

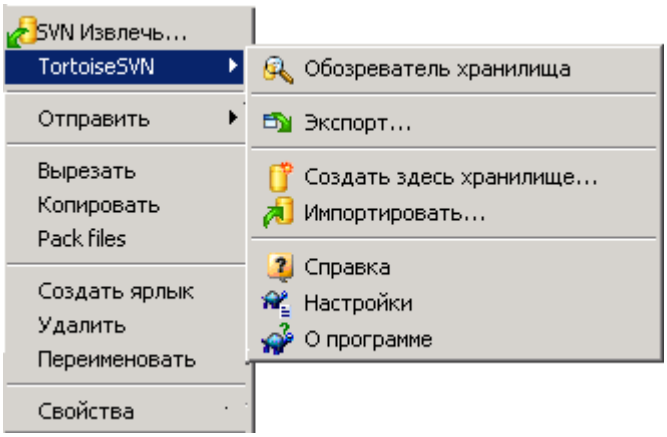


Рисунок 3.1. Меню TortoiseSVN для неверсированных папок

1. Откройте Проводник Windows
2. Создайте новую папку и назовите её, например, `SVNRepository`
3. Щёлкните правой кнопкой мыши на вновь созданной папке, и выберите TortoiseSVN → Создать здесь хранилище....

   Хранилище будет создано внутри новой папки. *Не редактируйте эти файлы самостоятельно!!!*. В случае возникновения ошибок убедитесь, что папка пуста и доступна для записи.

Подсказка

TortoiseSVN больше не предоставляет возможность создавать хранилища, использующие BDB, хотя их всё ещё возможно создать при помощи клиента командной строки. Хранилища FSFS обычно более легки в обслуживании, а также делают для нас более лёгкой поддержку и TortoiseSVN - из-за проблем с совместимостью разных версий BDB.

Будущие версии TortoiseSVN не будут поддерживать доступ к хранилищам BDB при помощи `file://` из-за этих проблем с совместимостью, хотя TortoiseSVN, конечно, будет поддерживать этот формат хранилища при доступе к серверу по протоколам `svn://`, `http://` или `https://`. По этой причине, мы настоятельно рекомендуем, чтобы все новые хранилища, доступ к которым должен осуществляться при помощи протокола `file://`, создавались в формате FSFS.

Мы также предлагаем, чтобы вы совсем не использовали доступ при помощи `file://`, кроме как в целях локального тестирования. Использование сервера более безопасно и более надёжно для всех задач, за исключением применения его для единственного разработчика.

3.1.3. Локальный доступ к хранилищу

Для доступа к вашему локальному хранилищу вам необходим путь к этой папке. Только запомните, что Subversion принимает все пути к хранилищам в виде `file:///C:/хранилищеSVN/`. Обратите внимание, что везде используется прямая косая черта.

Для доступа к хранилищу, находящемуся на сетевом разделяемом ресурсе, вы можете подключить этот ресурс как диск, или использовать путь UNC. Пути UNC используются в виде `file://ИмяСервера/путь/к/хранилищу`. Обратите внимание, что здесь только 2 ведуших прямых косых черты.

В версиях SVN до 1.2, пути UNC имели более запутанный вид `file:///\ИмяСервера/путь/к/хранилищу`. Хотя этот вид до сих пор поддерживается, использовать его не рекомендуется.

Внимание

Не создавайте и не обращайтесь к хранилищу на основе Berkeley DB, которое находится на разделяемом сетевом ресурсе. Оно *не может* существовать в рамках удалённой файловой системы. И даже в том случае, если у вас назначена буква для сетевого диска. Если вы пытаетесь использовать Berkeley DB на разделяемом сетевом ресурсе, результаты непредсказуемы - вы можете увидеть загадочные ошибки сразу же, или могут пройти месяцы, прежде чем вы обнаружите, что база данных хранилища коварно испорчена.

3.1.4. Доступ к хранилищу на сетевом ресурсе

Хотя в теории возможно разместить хранилище FSFS на сетевом ресурсе и предоставить доступ нескольким пользователям посредством протокола `file://`, это то, что совершенно точно *не рекомендуется* делать. На самом деле мы *весьма* не одобряем этого, и такое использование нами не поддерживается.

Во первых, вы предоставляете каждому пользователю непосредственный доступ на запись к хранилищу, и любой пользователь может случайно удалить всё хранилище или испортить его каким-либо другим способом.

Во вторых,не все протоколы разделения файлов по сети поддерживают блокировку, необходимую для Subversion, и вы можете обнаружить, что ваше хранилище повреждено. Возможно, это случится не сразу, но когда-нибудь два пользователя попробуют получить доступ к хранилищу в одно и то же время.

В третьих, разрешения на доступ к файлам должны указываться очень аккуратно. Возможно, вам удастся это сделать на "родном" разделяемом ресурсе Windows, но на SAMBA это особенно трудно.

Доступ при помощи `file://` предназначен только для локального использования одним пользователем, особенно для тестирования и отладки. Когда вы желаете предоставить общий доступ к хранилищу, вам *на самом деле* необходимо настроить подходящий сервер, и это не настолько сложно, как вы можете подумать. Прочтите Раздел 3.5, «Доступ к хранилищу» для рекомендаций по выбору и настройке сервера.

3.1.5. Организация данных в хранилище

Перед тем, как импортировать данные в хранилище, сначала подумайте о том, как вы хотите их организовать данные. Если использовать один из рекомендуемых способов, в дальнейшем вам будет намного легче.

Есть несколько стандартных рекомендуемых способов организации хранилища. Большинство людей создают папку `trunk`, в которой ведётся «основная линия» разработки, папку `branches`, содержащую копии ответвлений, и папку `tags` для копий меток. Если хранилище содержит только один проект, тогда их часто создают как папки верхнего уровня:

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

Если хранилище содержит несколько проектов, их часто упорядочивают по ответвлениям:

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

...или по проектам:

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

Упорядочивание по проектам имеет смысл, если проекты не сильно связаны, и каждый из них извлекается индивидуально. Для связанных проектов, где вы можете пожелать извлекать все

проекты за раз, или где все проекты связанны друг с другом в один распространяемый пакет, часто лучшим является упорядочивание по ответвлениям. В этом случае у вас только один ствол для извлечения, и легче видимы взаимосвязи между подпроектами.

В случае принятия подхода, когда `/trunk /tags /branches` - папки верхнего уровня, незачем говорить, что для каждого ответвления и метки копируется весь ствол, и в некотором роде эта структура предоставляет наибольшую гибкость.

Для несвязанных проектов вы можете предпочесть использовать раздельные хранилища. Когда вы фиксируете изменения, изменяется номер ревизии для всего хранилища, а не номер ревизии проекта. Когда два несвязанных проекта совместно используют одно хранилище, могут образовываться большие промежутки в номерах ревизий. Проекты Subversion и TortoiseSVN обладают одним адресом сетевого узла, но у них полностью раздельные хранилища, делающие возможной независимую разработку, без путаницы по поводу номеров сборок.

Конечно, вы вольны игнорировать эти обычные схемы. Вы можете реализовать любые виды изменений - всё, что лучше работает для вас или для вашей команды. Помните, что выбранный вами вариант не является чем-то незыблемым. Вы можете реорганизовать хранилище в любое время. Поскольку ответвления и метки - это обычные папки, TortoiseSVN может перемещать или переименовывать их, как вы пожелаете.

Переход с одного способа организации на другой - это вопрос выполнения серии перемещений на сервере. Если вам не нравится, как организованы данные в хранилище, просто переставьте папки.

Итак, если вы пока ещё не создали основную структуру папок в вашем хранилище, это надо сделать сейчас. Есть два способа этого достичь: если вы просто желаете создать структуру `/trunk /tags /branches` (/ствол /метки /ответвления), вы можете воспользоваться обозревателем хранилища для создания этих трёх папок (за три отдельных фиксации). Если же вы желаете создать более разветвлённую иерархию, то проще сначала создать структуру папок на диске и импортировать её за одну фиксацию, вот так:

1. создайте новую пустую папку на вашем жёстком диске
2. создайте желаемую структуру папок верхнего уровня внутри этой папки - но пока не помещайте в них никаких файлов!
3. импортируйте эту структуру в хранилище путём правого щелчка на папке и выбора TortoiseSVN → Импортировать... Тем самым вы импортируете вашу временную папку в корень хранилища и создадите основу для организации данных хранилища.

Обратите внимание: сама папка, которую вы импортируете, в хранилище не появляется, только её содержимое. Например, создайте следующую структуру папок:

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

Импортируйте `C:\Temp\New` в корень хранилища, который станет выглядеть следующим образом:

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

3.2. Резервирование хранилища

Какой бы тип хранилища вы не использовали, жизненно важно, чтобы вы делали регулярные резервные копии, и чтобы вы их проверяли. При сбое сервера у вас, возможно, останется доступ к последней версии ваших файлов, но без хранилища вся история будет потеряна навсегда.

Простейший (но не рекомендуемый) путь - просто скопировать папку хранилища на резервный носитель. Однако, в этом случае вы должны быть абсолютно уверены, что ни один процесс не имел доступа к данным. В этом контексте, доступ означает вообще *любой* доступ. В хранилище BDB происходит запись даже тогда, когда производится операция, которая, казалось бы, требует только чтения, такая как получение статуса. Если к вашему хранилищу производился хоть какой-то доступ во время копирования (остался открытым веб-обозреватель, WebSVN, и т.д.), резервная копия будет бесполезна.

Рекомендуемый метод - это запуск

```
svnadmin hotcopy путь/к/хранилищу путь/к/резервной/копии --clean-logs
```

для создания копии вашего хранилища безопасным способом. Затем эту копию можно резервировать. Опция `--clean-logs` не обязательна, но удаляет любые излишние файлы журналов, когда вы резервируете хранилище BDB, и тем самым сохраняет немного места.

Утилита `svnadmin` устанавливается автоматически при установке клиента командной строки Subversion. Если вы устанавливаете инструменты командной строки на ПК с Windows, то из доступных вариантов лучше загрузить установщик для Windows. Он сжат более эффективно, чем `.zip`-версия, так что объем закачки меньше, и он для вас установит необходимые пути. Вы можете загрузить последнюю версию клиента командной строки с *http://Subversion.tigris.org/servlets/ProjectDocumentList?folderID=91*.

3.3. Скрипты ловушек, выполняемые на стороне сервера

Ловушка - это программа, запускаемая по какому-либо событию в хранилище, такому как создание новой ревизии или изменение неверсированного свойства. Каждой ловушке передаётся достаточно информации для того, чтобы узнать, что это за событие, какие объекты затронуты, а также имя пользователя, инициировавшего событие. В зависимости от вывода или возвращаемого значения ловушки, программа ловушки может продолжить действие, прекратить его, или приостановить некоторым образом. За дополнительной информацией о реализованных ловушках обращайтесь, пожалуйста, к главе *Hook Scripts (Скрипты ловушек)* [http://svnbook.red-bean.com/en/1.5/svn.reposadmin.create.html#svn.reposadmin.create.hooks] в книге о Subversion.

Эти скрипты ловушек выполняются сервером, на котором расположено хранилище. TortoiseSVN также позволяет настроить скрипты ловушек, выполняемые локально клиентом при наступлении определённых событий. Для получения дополнительной информации смотрите Раздел 4.30.8, «Скрипты ловушек, выполняемые на стороне клиента».

Примеры скриптов ловушек вы можете найти в папке `hooks` хранилища. Эти скрипты подходят для серверов Unix/Linux, но они должны быть модифицированы, если у вас Windows-сервер. Ловушка может быть как пакетным, так и исполняемым файлом. Пример ниже показывает пакетный файл, который может быть использован для реализации ловушки pre-revprop-change (перед-изменением-свойства_ревизии).

```
rem Only allow log messages to be changed.
rem Допускать изменения только сообщений журнала.
if "%4" == "svn:log" exit 0
echo Property '%4' cannot be changed >&2
exit 1
```

Заметьте, что всё, посылаемое в stdout, отбрасывается. Если вы желаете, чтобы в диалоге 'Фиксация отклонена' появилось сообщение, вам надо послать его в stderr. В пакетном файле это достигается при помощи `>&2`

3.4. Ссылки для извлечения

Если вы желаете сделать своё хранилище Subversion доступным для других, вы можете разместить ссылку на него на вашем веб-сайте. Один из путей достижения большей доступности - размещение *ссылки для извлечения* для других пользователей TortoiseSVN.

При установке TortoiseSVN регистрируется новый протокол `tsvn:`. Когда пользователи TortoiseSVN щёлкают по такой ссылке, автоматически открывается диалог извлечения с уже заполненным URL-адресом хранилища.

Для размещения таких ссылок на вашей собственной HTML-странице необходимо добавить код вроде этого:

```
<a href="tsvn:http://project.domain.org/svn/trunk">
</a>
```

Конечно, будет ещё лучше, если вы добавите подходящую картинку. Вы можете использовать *логотип TortoiseSVN* [http://tortoisesvn.tigris.org/images/TortoiseCheckout.png] или воспользоваться собственной картинкой.

```
<a href="tsvn:http://project.domain.org/svn/trunk">
<img src=TortoiseCheckout.png></a>
```

Вы можете также сделать так, чтобы ссылка указывала на конкретную ревизию, например

```
<a href="tsvn:http://project.domain.org/svn/trunk?100">
</a>
```

3.5. Доступ к хранилищу

Для использования TortoiseSVN (или любого другого клиента Subversion) вам необходимо место, где будут располагаться хранилища. Вы можете содержать ваши хранилища локально и обращаться к ним, используя протокол `file://`, или же вы можете разместить их на сервере и получать доступ к ним посредством протоколов `http://` или `svn://`. Оба этих протокола для работы с сервером могут ещё и шифроваться, в этом случае это будут протоколы `https://` или `svn+ssh://`, или же вы можете воспользоваться `svn://` с SASL.

Если вы используете общедоступную службу размещения проектов (хостинг), такую как *Google Code* [http://code.google.com/hosting/], или ваш сервер уже кем-либо настроен, то вам больше ничего делать не нужно. Переходите прямо к Глава 4, *Руководство по ежедневному использованию*.

Если у вас нет сервера и вы работаете в одиночку, или если вы только оцениваете Subversion и TortoiseSVN самостоятельно,тогда локальные хранилища, вероятно, будут лучшим выбором. Просто создайте хранилище на вашем собственном ПК, как описано ранее в Глава 3, *Хранилище*. Вы можете пропустить остаток этой главы и сразу перейти к Глава 4, *Руководство по ежедневному использованию*, чтобы узнать, как приступить к их использованию.

Если вы думаете разместить многопользовательское хранилище на сетевом ресурсе, подумайте ещё раз. Чтобы узнать, почему мы считаем, что это плохая идея, прочтите Раздел 3.1.4, «Доступ к хранилищу на сетевом ресурсе». Установка и настройка сервера не так трудна, как кажется, и предоставит вам как большую надёжность, так и, вероятно, большую скорость.

Следующие разделы являются пошаговым руководством по установке и настройке такого сервера на компьютере с Windows. Конечно, вы можете поднять сервер и на Linux-машине, но это выходит за пределы этого руководства. Более подробную информацию о параметрах сервера Subversion, а также о том, как выбрать наилучшую архитектуру для вашей ситуации, можно найти в Книге о Subversion в разделе *Server Configuration (Конфигурирование сервера)* [http://svnbook.red-bean.com/en/1.5/svn.serverconfig.html].

3.6. Сервер на основе Svnserve

3.6.1. Введение

Subversion содержит Svnserve - облегчённый автономный сервер, использующий собственный протокол поверх обычного TCP/IP соединения. Он идеально подходит для небольших инсталляций, или там, где не может быть использован полнофункциональный сервер Apache.

В большинстве случаев svnserve легче настроить и он работает быстрее, нежели сервер на базе Apache, хотя в нём нет некоторых расширенных возможностей. А теперь, когда добавлена поддержка SASL, его легко можно сделать также и безопасным.

3.6.2. Установка svnserve

1. Загрузите последнюю версию Subversion по адресу *http://subversion.tigris.org/servlets/ProjectDocumentList?folderID=91*. Или загрузите заранее настроенный установщик с сайта CollabNet по адресу *http://www.collab.net/downloads/subversion*. Этот установщик настроит запуск svnserve в качестве службы Windows, а также в него включено несколько инструментов, которые вам понадобятся, если вы собираетесь использовать для безопасности SASL.

2. Если у вас уже установлена какая-либо из версий Subversion и запущен svnserve, то перед тем, как продолжить, вам необходимо его остановить.

3. Запустите установщик Subversion. В случае, если вы запустили установку на вашем сервере (рекомендуемый вариант), можете пропустить шаг 4.

4. При помощи Проводника Windows откройте папку, где установлена Subversion (обычно `C:\Program Files\Subversion`) и в папке `bin` найдите файлы `svnserve.exe`, `intl3_svn.dll`, `libapr.dll`, `libapriconv.dll`, `libapriutil.dll`, `libdb*.dll`, `libeay32.dll` и `ssleay32.dll` и скопируйте эти файлы (или сразу всю папку `bin`) в папку на вашем сервере, например `c:\svnserve`

3.6.3. Запуск svnserve

Теперь, когда svnserve установлен, надо запустить его на вашем сервере. Простейший подход - выполнить из командной строки или при помощи ярлыка Windows следующее:

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

svnserve запустится и начнёт ожидать входящие запросы на порту 3690. Опция --daemon запускает svnserve в фоновом режиме (daemon), и он будет выполняться, пока его специально не завершат.

Если вы ещё не создали хранилище, следуйте указаниям, приведённым в разделе установки сервера Apache Раздел 3.7.4, «Настройка».

Чтобы проверить, работает ли svnserve, используйте TortoiseSVN → Обозреватель хранилища для просмотра хранилища.

Предполагая, что хранилище расположено в `c:\repos\TestRepo`, а сервер называется `localhost`, введите:

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

в ответ на запрос обозревателя хранилища.

Вы можете также усилить безопасность и сократить время на ввод URL при работе с svnserve, используя параметр --root для указания местоположения корневой папки и ограничения доступа только её пределами:

```
svnserve.exe --daemon --root диск:\путь\к\корню\хранилища
```

Применительно к предыдущей проверке, svnserve будет запускаться так:

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

И URL в обозревателе хранилища TortoiseSVN сократится до:

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

Обратите внимание, параметр --root также необходим, если хранилище и svnserve находятся в разных разделах или на разных дисках вашего сервера.

Svnserve может обслуживать любое количество хранилищ - просто разместите их где-либо ниже только что заданной корневой папки, и обращайтесь к ним по URL-адресу, определённому относительно этого корня.

Внимание

Не создавайте и не обращайтесь к хранилищу на основе Berkeley DB, которое находится на разделяемом сетевом ресурсе. Оно *не может* существовать в рамках удалённой файловой системы. И даже в том случае, если у вас назначена буква для сетевого диска. Если вы пытаетесь использовать Berkeley DB на разделяемом сетевом ресурсе, результаты непредсказуемы - вы можете увидеть загадочные ошибки сразу же, или могут пройти месяцы, прежде чем вы обнаружите, что база данных хранилища коварно испорчена.

3.6.3.1. Запуск svnserve как службы

Как правило, запускать svnserve как обычную программу - не самый лучший способ: это означает, всегда должен быть пользователь, вошедший в систему на сервере, и что надо не забывать запускать её каждый раз после перезагрузки. Есть лучший способ - запуск и исполнение svnserve как службы Windows. Начиная с Subversion 1.4, svnserve может быть установлен как обычная служба Windows.

Для того, чтобы установить svnserve как обычную службу Windows, выполните следующую команду, создающую службу, автоматически запускаемую при запуске Windows (всё в одной строке):

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

Если где-нибудь в пути встречаются пробелы, необходимо заключить путь в заэкранированные кавычки, вот так:

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

Вы также можете добавить описание после создания службы, которое будет показываться в оснастке 'Управление службами Windows'.

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

Обратите внимание на довольно необычный формат командной строки, используемой `sc`. В парах `ключ= значение` не должно быть пробела между ключом и =, но должен быть пробел перед значением.

Подсказка

Теперь Microsoft рекомендует, чтобы службы выполнялись под учётной записью или Локальной службы, или Сетевой службы. Обратитесь к *The Services and Service Accounts Security Planning Guide* [http://www.microsoft.com/technet/security/topics/serversecurity/serviceaccount/default.mspx]. Для создания службы под учётной записью Локальной службы, добавьте к вышеприведённому примеру следующее:

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

Заметьте, вы должны будете предоставить учётной записи Локальной службы соответствующие права и к Subversion, и к хранилищам. Такие же права потребуются и любым приложениям, которые используются скриптами ловушек. Встроенная группа для этого называется просто "СЛУЖБА" (в английской версии - "LOCAL SERVICE").

После установки службы вам необходимо запустить её при помощи оснастки управления службами (это понадобится только в этот раз; она будет запускаться автоматически при перезагрузке сервера).

За более подробной информацией обращайтесь к url="http://svn.collab.net/repos/svn/trunk/notes/windows-service.txt">*Windows Service Support for Svnserve (Поддержка службы Windows программой Svnserve)*

Если вы установили более раннюю версию svnserve при помощи программы-обёртки `SvnService`, и теперь вы желаете использовать вместо неё встроенную поддержку, вам необходимо отменить регистрацию этой обёртки как службы (не забудьте сначала её остановить!). Просто используйте команду

```
svnservice -remove
```

для удаления записи в реестре, относящейся к этой службе.

3.6.4. Элементарная аутентификация в svnserve

Настройки svnserve по умолчанию предоставляют анонимный доступ только для чтения. Это значит, что вы можете использовать URL `svn://` для извлечения и обновления, или воспользоваться обозревателем хранилища TortoiseSVN для просмотра хранилища, но у вас не будет возможности зафиксировать изменения.

Для разрешения доступа на запись в хранилище, вам необходимо отредактировать файл `conf/svnserve.conf` в папке вашего хранилища. Этот файл управляет конфигурацией svnserve, а также содержит полезную документацию.

Вы можете разрешить анонимный доступ на запись простой установкой:

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

Однако, вы не узнаете, кто произвёл изменения в хранилище, поскольку свойство `svn:author` будет пустым. Вы также не сможете управлять тем, кто может вносить изменения в хранилище. Это довольно рискованная настройка!

Одним из путей для преодоления этого является создание базы данных паролей:

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

Где `userfile` - файл, находящийся в той же папке, что и `svnserve.conf`. Этот файл может располагаться где-нибудь в другом месте вашей файловой системы (это может пригодиться, когда у вас есть несколько хранилищ с одинаковыми правами доступа) и может быть указан по абсолютному пути, или по относительному пути от папки `conf`. Если вы указываете путь, он должен быть написан `/в/стиле/unix`. Использование \ или букв дисков работать не будет. Файл `userfile` должен иметь следующую структуру:

```
[users]
username = password
...
```

Этот пример запрещает любой доступ для анонимных (неаутентифицированных) пользователей, и предоставляет доступ для чтения-записи пользователям, перечисленным в `userfile`.

Подсказка

Если у вас есть несколько хранилищ, использующих одну базу данных паролей, то применение области аутентификации сделает жизнь пользователей легче, поскольку TortoiseSVN может запоминать учётные данные и вам достаточно будет ввести их только однажды. Дополнительная информация содержится в Книге о Subversion, а именно в разделах *Create a 'users' file and realm (Создание файла 'users' и области аутентификации)* [http://svnbook.red-bean.com/en/1.5/svn.serverconfig.svnserve.html#svn.serverconfig.svnserve.auth.users] и *Client Credentials Caching (Кэширование клиентских учётных данных)* [http://svnbook.red-bean.com/en/1.5/svn.serverconfig.netmodel.html#svn.serverconfig.netmodel.credcache]

3.6.5. Улучшение безопасности при помощи SASL

3.6.5.1. Что такое SASL?

Простой уровень аутентификации и безопасности Сайрус (Cyrus Simple Authentication and Security Layer, SASL) - это ПО с открытым исходным кодом, написанное в университете Карнеги-Меллона. Оно добавляет общие возможности по аутентификации и шифрованию в любой сетевой протокол, и, начиная с Subversion 1.5, сервер svnserve и клиент TortoiseSVN знают, как использовать эту библиотеку.

Более полное обсуждение доступных параметров приводится в книге о Subversion в разделе *Using svnserve with SASL (Использование svnserve с SASL)* [http://svnbook.red-bean.com/en/1.5/svn.serverconfig.svnserve.html#svn.serverconfig.svnserve.sasl]. Если же вам нужен простой способ настройки безопасных аутентификации и шифрования на Windows-сервере, чтобы обеспечить безопасный доступ к вашему хранилищу через большой опасный Интернет, то читайте дальше.

3.6.5.2. Аутентификация при помощи SASL

Для включения механизмов SASL на сервере, вам необходимо сделать три вещи. Во-первых, создайте раздел `[sasl]` в файле `svnserve.conf` вашего хранилища со следующей парой ключ-значение:

```
use-sasl = true
```

Во-вторых, создайте файл с названием `svn.conf` в подходящем месте - обычно это папка, в которой установлена Subversion.

В-третьих, создайте два новых ключа реестра, служащих для того, чтобы сообщить SASL, где искать необходимые вещи. Создайте раздел реестра с названием `[HKEY_LOCAL_MACHINE \SOFTWARE\Carnegie Mellon\Project Cyrus\SASL Library]` и поместите в него два новых строковых параметра: `SearchPath`, указывающий путь к папке, содержащей дополнительные подключаемые модули `sasl*.dll` (обычно это папка установки Subversion), и `ConfFile`, задающий папку, содержащую файл `svn.conf`. Если вы использовали установщик CollabNet, то эти параметры реестра для вас уже были созданы.

Исправьте файл `svn.conf`, чтобы он содержал следующее:

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

Последняя строка показывает местоположение базы данных аутентификации, которая является файлом с названием `sasldb`. Он может быть где угодно, но довольно удобное место - это в родительской папке хранилища. Убедитесь, что у службы svnserve есть доступ для чтения к этому файлу.

Если svnserve уже запущена, вам понадобится перезапустить её для того, чтобы она прочитала обновлённую конфигурацию.

Теперь, когда эти настройки произведены, всё, что вам необходимо сделать - это создать несколько пользователей и паролей. Для этого вам понадобится программа `saslpasswd2`. Если вы использовали установщик CollabNet, эта программа будет в папке установки. Выполните команду вроде этой:

```
saslpasswd2 -c -f C:\TortoiseSVN\sasldb -u область имя_пользователя
```

Опция `-f` задаёт местоположение базы данных, `область` должна быть такой же, как и значение, указанное в файле `svnserve.conf` вашего хранилища, и `имя_пользователя` - это именно то, что вы и подозреваете. Обратите внимание: область не должна содержать символов пробела.

Вы можете вывести список пользователей, содержащихся в базе данных, при помощи программы `sasldblistusers2`.

3.6.5.3. Шифрование при помощи SASL

Для включения и выключения различных уровней шифрования вы можете указать два значения в файле `svnserve.conf` вашего хранилища:

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

Переменные `min-encryption` и `max-encryption` управляют уровнем шифрования, запрашиваемого сервером. Для полного отключения шифрования установите оба значения в 0. Для включения простой проверки контрольной суммы данных (т.е. для предотвращения искажения и обеспечения целостности данных без шифрования), установите оба значения в 1. Если вы желаете разрешить (но не требовать) шифрование, установите минимальное значение в 0, а максимальное - в какое-либо значение длины ключа в битах. Для безусловного требования шифрования, задайте в обоих значениях числа, большие 1. В нашем предыдущем примере мы требовали, чтобы клиенты использовали шифрование с длиной ключа по крайней мере 128 бит, но не более 256 бит.

3.6.6. Аутентификация при помощи svn+ssh

Другой путь для аутентификации пользователей сервера на основе svnserve - использование безопасной оболочки (SSH) для туннелирования запросов. Её не так просто настроить, как SASL, но она может быть полезной в некоторых случаях.

При этом подходе, svnserve не работает в фоновом режиме, наоборот, безопасная оболочка запускает для вас svnserve как пользователь, которого она аутентифицировала. Для того, чтобы это работало, вам необходим демон безопасной оболочки на вашем сервере.

Основной метод настройки сервера приведён в Приложение G, *Организация защиты Svnserve при помощи SSH*. Вы можете найти другую информацию на тему SSH в ЧаВо (FAQ) при помощи поиска по слову «SSH».

Дополнительную информацию о svnserve можно найти в *Книге о Subversion* [http://svnbook.red-bean.com].

3.6.7. Авторизация с учётом пути в svnserve

Начиная с Subversion версии 1.3, svnserve поддерживает ту же схему авторизации с учётом пути `mod_authz_svn`, которая доступна с сервером Apache. Вам необходимо отредактировать файл `conf/svnserve.conf` в папке вашего хранилища и добавить строку, ссылающуюся на файл авторизации.

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

Здесь, `authz` - файл, созданный вами для задания разрешений доступа. Вы можете использовать отдельный файл для каждого хранилища, или же вы можете использовать один и тот же файл для нескольких хранилищ. Формат файла описан в Раздел 3.7.6, «Авторизация с учётом пути».

3.7. Сервер на основе Apache

3.7.1. Введение

Наиболее гибкая из всех возможных установок Subversion - это установка на основе Apache. Хотя её несколько сложнее устанавливать и настраивать, она предлагает преимущества, которые другие серверы предоставить не могут:

WebDAV
: Сервер Subversion на основе Apache использует протокол WebDAV, который также поддерживается многими другими программами. Вы можете, например, подключить хранилище как «веб-папку» в Проводнике Windows и обращаться к нему как к любой другой папке в файловой системе.

Просмотр хранилища при помощи веб-обозревателя
: Вы можете указать в веб-обозревателе URL вашего хранилища и просматривать его содержимое даже без установленного клиента Subversion. Это предоставляет доступ к вашим данным значительно большему кругу пользователей.

Аутентификация
: Вы можете использовать любой механизм аутентификации, поддерживаемый Apache, включая SSPI и LDAP.

Безопасность
: Поскольку Apache очень стабилен и безопасен, вы автоматически получаете ту же степень безопасности для вашего хранилища, включая шифрование SSL.

3.7.2. Установка Apache

Первое, что вам необходимо перед установкой Apache - это компьютер под управлением Windows 2000, Windows XP+SP1, Windows 2003, Vista или Server 2008.

Внимание

Обратите внимание: использование Windows XP без пакета обновления 1 приводит к передаче фиктивных сетевых данных и вследствие этого может испортить ваше хранилище!

1. Загрузите последнюю версию веб-сервера Apache по адресу *http://httpd.apache.org/download.cgi*. Убедитесь, что вы загрузили версию 2.2.x - версии 1.3.xx работать не будут!

   Установщик для Apache в виде msi-файла можно найти, щёлкнув на `other files`, после чего перейти к `binaries/win32`. Возможно, вам лучше выбрать msi-файл `apache-2.2.x-win32-x86-openssl-0.9.x.msi` (тот, который включает OpenSSL).

2. Как только у вас будет установщик Apache2, вы можете запустить его, и он проведёт вас через процесс установки. Убедитесь, что правильно ввели URL сервера (если у вас нет DNS-имени для сервера, введите IP-адрес). Я рекомендую установить Apache *для всех пользователей, на 80 порту, как службу*. Примечание: если у вас уже запущен IIS или любая другая программа, принимающая запросы на 80 порту, установка может завершиться неудачей. Если это произошло, перейдите в папку для программ (часто это `C:\Program Files\`), далее в `\Apache Group\Apache2\conf` и найдите файл `httpd.conf`. Отредактируйте его, так чтобы `Listen 80` было изменено на свободный порт, например `Listen 81`. После этого перезапустите установку - на этот раз она должна завершиться без проблем.

3. Теперь проверьте работоспособность веб-сервера Apache, задав в веб-обозревателе адрес `http://localhost/` - должен показаться веб-сайт по умолчанию.

Предостережение

Если вы решили установить Apache как службу, будьте внимательны, поскольку по умолчанию он запускается с правами локальной системной учётной записи. Будет более безопасно, если вы создадите отдельную учётную запись, под которой будет запускаться Apache.

Убедитесь, что для учётной записи на сервере, под которой запускается Apache, есть явная запись в списке контроля доступа папки хранилища (правая кнопка на папке | свойства | безопасность) с полными правами доступа. Иначе пользователи не смогут зафиксировать свои изменения.

Даже если Apache запускается под локальной системной учётной записью, права на папку всё равно должны быть указаны (в данном случае - для учётной записи SYSTEM).

Если же у Apache нет этих прав, ваши пользователи получат сообщение об ошибке «В доступе отказано (Access denied)», которая в журнале ошибок Apache появляется как ошибка 500.

3.7.3. Установка Subversion

1. Загрузите последнюю версию исполняемых файлов Subversion под Win32 для Apache. Убедитесь, что вы получили правильную версию для интеграции с вашей версией Apache, иначе вы получите невразумительное сообщение об ошибке при перезапуске. Если у вас Apache версии 2.2.x, то пройдите по ссылке *http://subversion.tigris.org/servlets/ProjectDocumentList?folderID=8100*.

2. Запустите установщик Subversion и следуйте инструкциям. Если установщик Subversion обнаружит установленный сервер Apache, то вы уже почти закончили. Если не сможет - вам надо будет выполнить несколько дополнительных шагов.

3.

   Используя Проводник Windows, скопируйте из папки, где установлена Subversion (обычно `c:\program files\Subversion`), в папку с модулями Apache (обычно `c:\program files\apache group\apache2\modules`) файлы `/httpd/mod_dav_svn.so` и `mod_authz_svn.so`.

4. Скопируйте файлы `/bin/libdb*.dll` и `/bin/intl3_svn.dll` из папки, где установлена Subversion, в папку bin Apache.

5. Откройте конфигурационный файл Apache (обычно `C:\Program Files\Apache Group \Apache2\conf\httpd.conf`) при помощи текстового редактора, такого как Блокнот, и внесите следующие изменения:

   Разкомментируйте (убрав символ '#') следующие строки:

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

   Добавьте следующие две строки в конце секции `LoadModule`:

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

3.7.4. Настройка

Итак, вы установили Apache и Subversion, но Apache до сих пор не знает, как работать с клиентами Subversion, такими как TortoiseSVN. Для того, чтобы Apache знал, какой URL должен быть использован для хранилищ Subversion, вы должны отредактировать конфигурационный файл Apache (обычно `c:\program files\apache group\apache2\conf\httpd.conf`) при помощи любого текстового редактора (например, Блокнота):

1. В конец файла настроек добавьте следующие строки:

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

   Эти настройки сообщают Apache о том, что все ваши хранилища Subversion физически располагаются внутри папки `D:\SVN`. Хранилища предоставляются внешнему миру по URL: `http://MyServer/svn/` . Доступ разрешён только известным пользователям/паролям, перечисленным в файле `passwd`.

2. Для создания файла `passwd`, откройте командную строку (окно эмуляции DOS), перейдите в папку `apache2` (обычно `c:\program files\apache group\apache2`) и создайте файл путём ввода:

```
bin\htpasswd -c passwd <username>
```

Это создаёт файл с именем `passwd`, который используется для аутентификации. Дополнительные пользователи могут быть добавлены при помощи

```
bin\htpasswd passwd <username>
```

3. Перезапустите службу Apache.
4. Откройте в вашем веб-обозревателе `http://MyServer/svn/MyNewRepository` (где `MyNewRepository` - имя ранее созданного хранилища Subversion). Если всё сделано правильно, у вас будут запрошены имя пользователя и пароль, после чего вы сможете увидеть содержимое вашего хранилища.

Краткое объяснение того, что вы только что ввели:

Настройка	Пояснение
<Location /svn>	означает, что хранилища Subversion доступны по URL `http://MyServer/svn/`
DAV svn	сообщает Apache, какой модуль будет ответственен за обслуживание этого URL, - в данном случае модуль Subversion.
SVNListParentPath on	Для Subversion версии 1.3 или более новой, это указание разрешает выдавать перечень всех доступных хранилищ в папке `SVNParentPath`.
SVNParentPath D:\SVN	предписывает Subversion искать хранилища в папке `D:\SVN`
SVNIndexXSLT "/svnindex.xsl"	Используется для того, чтобы просмотр через веб-обозреватель выглядел более симпатично.
AuthType Basic	для включения базовой аутентификации, т.е. имя_пользователя/пароль
AuthName "Subversion repositories"	используется как информация для пользователя в диалоге аутентификации, сообщая, для доступа к какой области требуется эта аутентификация
AuthUserFile passwd	указывает, какой файл паролей использовать при аутентификации
AuthzSVNAccessFile	расположение файла доступа для путей внутри хранилища Subversion
Require valid-user	предписывает, что только пользователям, предоставившим правильные имя_пользователя/пароль, будет разрешён доступ к URL

Таблица 3.1. Настройки Apache в `httpd.conf`

Но это только пример. Есть ещё очень много возможностей того, что вы можете сделать с веб-сервером Apache.

- Если вы желаете, чтобы доступ для чтения к вашему хранилищу был у всех, а доступ для записи - только у некоторых пользователей, вы можете изменить строку

```
Require valid-user
```

на

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

- Использование файла `passwd` ограничивает или разрешает доступ сразу ко всем вашим хранилищам. Если вам необходим лучший контроль за тем, какие пользователи имеют доступ к каждой папке внутри хранилища, вы можете разкомментировать строку

  ```
  #AuthzSVNAccessFile svnaccessfile
  ```

  и создать файл доступа Subversion. Apache будет проверять, чтобы только дозволенные пользователи имели доступ к папке `/svn`, и после этой проверки будет передавать имя пользователя в модуль `AuthzSVNAccessFile` для того, чтобы можно было осуществить более тонко настроенный доступ на основе правил из файла доступа Subversion. Обратите внимание, пути указываются как `хранилище:путь` или просто `путь`. Если вы не укажете конкретное хранилище, это правило доступа будет применено ко всем хранилищам в папке `SVNParentPath`. Формат файла политики авторизации, используемого в `mod_authz_svn`, описан в Раздел 3.7.6, «Авторизация с учётом пути»

- Для того, чтобы сделать просмотр хранилища при помощи веб-обозревателя более 'симпатичным', разкомментируйте строку

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

  и поместите файлы `svnindex.xsl`, `svnindex.css` и `menucheckout.ico` в корневую папку документов Apache (обычно `C:/Program Files/Apache Group/Apache2/htdocs`). Папка задаётся при помощи указания `DocumentRoot` в файле конфигурации Apache.

  Вы можете скачать эти три файла прямо из нашего хранилища исходного кода по ссылке *http://tortoisesvn.tigris.org/svn/tortoisesvn/trunk/contrib/other/svnindex*. (Раздел 3, «TortoiseSVN бесплатен!» рассказывает, как получить доступ к хранилищу исходного кода TortoiseSVN).

  В xsl-файле из хранилища TortoiseSVN реализована одна хитрость: при просмотре хранилища при помощи веб-обозревателя справа от каждой папки показывается значок. Если щёлкнуть по нему, будет запущен диалог извлечения TortoiseSVN для этого URL.

3.7.5. Работа с несколькими хранилищами

Если вы используете указание `SVNParentPath`, тогда вам не надо изменять конфигурационный файл Apache каждый раз при добавлении нового хранилища Subversion. Просто создайте новое хранилище в той же папке, где и первое хранилище, и всё! В моей компании у меня есть прямой доступ к этой папке на сервере посредством SMB (обычный доступ к файлам в Windows). Поэтому я просто создаю там новую папку, выполняю команду TortoiseSVN TortoiseSVN → Создать здесь хранилище... и у нового проекта уже есть свой дом...

Если вы используете Subversion версии 1.3 или более поздней, вы можете использовать указание `SVNListParentPath on` для того, чтобы Apache создавал перечень всех доступных проектов при попытке открыть веб-обозревателем родительскую папку, а не конкретное хранилище.

3.7.6. Авторизация с учётом пути

Модуль `mod_authz_svn` позволяет детально управлять правами доступа, основываясь на именах пользователей и на путях в хранилище. Эта возможность доступна при работе с сервером Apache, а начиная с Subversion версии 1.3 она также доступна и в svnserve.

Файл для примера может выглядеть подобно этому:

```
[groups]
admin = john, kate
devteam1 = john, rachel, sally
devteam2 = kate, peter, mark
docs = bob, jane, mike
training = zak
# Default access rule for ALL repositories
# Everyone can read, admins can write, Dan German is excluded.
# Правило доступа по умолчанию для ВСЕХ хранилищ
# У всех есть права на чтение, у группы admin есть права на запись,
# у Dan German - нет допуска.
[/]
* = r
@admin = rw
dangerman =
# Allow developers complete access to their project repos
# Разрешить разработчикам полный доступ к хранилищам,
# содержащим их проекты
[proj1:/]
@devteam1 = rw
[proj2:/]
@devteam2 = rw
[bigproj:/]
@devteam1 = rw
@devteam2 = rw
trevor = rw
# Give the doc people write access to all the docs folders
# Предоставить пользователям, создающим документацию,
# доступ для записи к папкам с документацией (docs)
[/trunk/doc]
@docs = rw
# Give trainees write access in the training repository only
# Предоставить стажёрам доступ для записи только к
# учебному хранилищу
[TrainingRepos:/]
@training = rw
```

Заметьте, проверка каждого пути может быть очень затратной операцией, особенно в случае журнала ревизий. Сервер проверяет каждый изменённый путь в каждой ревизии на возможность чтения, что может повлечь существенные затраты времени для ревизий, затрагивающих большое количество файлов.

Аутентификация и авторизация - отдельные процессы. Если пользователь хочет получить доступ по какому-либо пути в хранилище, он должен удовлетворить *обоим* наборам требований: обычным требованиям аутентификации и требованиям авторизации из файла доступа.

3.7.7. Аутентификация при помощи домена Windows

Как вы могли уже заметить, вам необходимо сделать запись имя_пользователя/пароль в файле `passwd` для каждого пользователя отдельно. И если (в целях безопасности) вы желаете, чтобы ваши пользователи периодически меняли свои пароли, вы должны будете выполнять эти изменения вручную.

Однако, есть решение этой проблемы - по крайней мере, если вы обращаетесь к хранилищу из ЛВС с контроллером домена Windows: `mod_auth_sspi`!

Первоначально модуль SSPI был предложен Syneapps вместе с исходным кодом, но его разработка была остановлена. Впрочем, не отчаивайтесь: сообществом работа над ним была возобновлена, и было реализовано несколько улучшений. Теперь этот проект располагается на *SourceForge* [http://sourceforge.net/projects/mod-auth-sspi/].

- Скачайте модуль, соответствующий вашей версии Apache, и скопируйте файл `mod_auth_sspi.so` в папку модулей Apache.
- Отредактируйте конфигурационный файл Apache: добавьте строку

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

  в секцию `LoadModule`. Убедитесь, что вы вставили эту строку *перед* строкой

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

- Для того чтобы это размещение Subversion использовало этот тип аутентификации, вы должны изменить строку

  ```
  AuthType Basic
  ```

  на

  ```
  AuthType SSPI
  ```

  ещё вам надо добавить

  ```
  SSPIAuth On
  SSPIAuthoritative On
  SSPIDomain <domaincontroller>
  SSPIOmitDomain on
  SSPIUsernameCase lower
  SSPIPerRequestAuth on
  SSPIOfferBasic On
  ```

  внутри блока `<Location  /svn>`. Если у вас нет контроллера домена, оставьте имя контроллера домена в виде `<domaincontroller>`.

Заметьте, что если вы производите аутентификацию при помощи SSPI, то вам больше не нужна строка `AuthUserFile` для определения файла паролей. Вместо этого Apache аутентифицирует ваши имя пользователя и пароль при помощи домена Windows. Дополнительно вам необходимо будет обновить список пользователей в файле доступа Subversion (`svnaccessfile`) для того, чтобы они и там указывались как `ДОМЕН\имя_пользователя`.

Важно

Аутентификация SSPI работает только для соединений, защищённых SSL (т.е. https). Если вы используете только обычные http-соединения с сервером, она работать не будет.

Для включения SSL на вашем сервере, смотрите Раздел 3.7.9, «Защита сервера при помощи SSL».

Подсказка

Файлы `AuthzSVNAccessFile` в Subversion учитывают регистр в именах пользователей (`JUser` отличается от `juser`).

В мире Microsoft домены Windows и имена пользователей не зависят от регистра. Но даже в такой ситуации некоторые сетевые администраторы предпочитают создавать учётные записи пользователей ВСмешанномСтиле (например, `JUser`).

Эта разница может затронуть вас при использовании аутентификации SSPI, поскольку домены Windows и имена пользователей передаются в Subversion в том виде, в каком пользователь набрал их при запросе. Internet Explorer часто передаёт имя пользователя Apache автоматически, используя тот же регистр, который был использован при создании учётной записи.

В итоге, вам может понадобиться по крайней мере две записи в вашем файле `AuthzSVNAccessFile` для каждого пользователя - одна строчными буквами, другая - в том виде, в каком Internet Explorer передаёт её Apache. Вам также надо будет обучить пользователей тому, чтобы они вводили свои учётные данные для доступа к хранилищам посредством TortoiseSVN в нижнем регистре.

Журнал ошибок и журнал доступа Apache - ваши лучшие друзья в расшифровывании такого рода проблем, поскольку они помогут вам определить строку с именем пользователя, переданную в модуль `AuthzSVNAccessFile` Subversion. Возможно, вам придётся поэкспериментировать с точным форматом строки пользователя в файле `svnaccessfile` (например, `ДОМЕН\пользователь` или `ДОМЕН//пользователь`) для того, чтобы всё работало.

3.7.8. Множественные поставщики аутентификации

Возможно использование более одного поставщика аутентификации для хранилища Subversion. Для этого необходимо сделать каждый тип аутентификации не-авторитарным (non-authoritative), чтобы Apache проверял несколько поставщиков на подходящие имя_пользователя/пароль.

Обычный сценарий - использование двух поставщиков аутентификации: во-первых - домена Windows, во-вторых - файла `passwd` для предоставления возможности доступа к SVN пользователям, не зарегистрированным в домене Windows.

- Для включения сразу обоих поставщиков аутентификации: и домена Windows, и файла `passwd`, добавьте следующие строки в блоке `<Location>` конфигурационного файла Apache:

```
AuthBasicAuthoritative Off
SSPIAuthoritative Off
```

Вот пример полной конфигурации Apache для совмещённой аутентификации через домен Windows и файл `passwd`:

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

  AuthName "Subversion repositories"
  AuthzSVNAccessFile svnaccessfile.txt
```

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

# Htpasswd Logins.
  AuthType Basic
  AuthBasicAuthoritative Off
  AuthUserFile passwd

  Require valid-user
</Location>
```

3.7.9. Защита сервера при помощи SSL

Хотя в Apache 2.2.x и есть поддержка OpenSSL, по умолчанию она отключена. Вам необходимо включить её вручную.

1. В файле конфигурации Apache разкомментируйте строки:

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

   и в конце

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

   после чего измените строку

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

   на

   ```
   SSLMutex default
   ```

2. Далее вам надо создать сертификат SSL. Для этого откройте командную строку (окно эмуляции DOS) и перейдите в папку Apache (например, `C:\program files\apache group \apache2`) и введите следующую команду:

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

   У вас будет запрошена парольная фраза. Пожалуйста, не используйте просто слова, используйте целые предложения, например, часть стихотворения. Чем длиннее фраза, тем лучше. Ещё вам надо будет ввести URL вашего сервера. Все другие вопросы необязательны, но мы рекомендуем ответить также и на них.

   Обычно файл `privkey.pem` создаётся автоматически, но если этого не произошло, вам надо ввести эту команду для его генерации:

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

   Потом введите команды

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

и (одной строкой)

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

потом (одной строкой)

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

Это создаст сертификат со сроком действия в 4000 дней. И, наконец, введите (одной строкой):

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

Эти команды создали несколько файлов в папке `conf` Apache (`server.der.crt`, `server.csr`, `server.key`, `.rnd`, `privkey.pem`, `server.cert`).

3. Перезапустите службу Apache.
4. Введите в вашем веб-обозревателе `https://servername/svn/project` ...

SSL и Internet Explorer

Если вы обезопасили свой сервер при помощи SSL и используете аутентификацию при помощи домена Windows, вы обнаружите, что просмотр хранилища при помощи Internet Explorer больше не работает. Не волнуйтесь - это только Internet Explorer не способен провести аутентификацию. В других обозревателях такой проблемы нет, так что TortoiseSVN и любой другой клиент Subversion по-прежнему способны проводить аутентификацию.

Если вам всё равно необходимо просматривать хранилище при помощи IE, у вас есть следующие варианты:

- задайте отдельное указание `<Location /путь>` в конфигурационном файле Apache, и добавьте `SSPIBasicPreferred On`. Это позволит IE аутентифицироваться, но зато другие обозреватели и Subversion не будут способны аутентифицироваться для этого размещения.
- Попробуйте также просмотр с незашифрованной аутентификацией (без SSL). Странно, но у IE нет каких-либо проблем с аутентификацией, если соединение не защищено при помощи SSL.
- Часто в "стандартных" настройках SSL в виртуальном узле SSL Apache есть следующее указание:

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

  Есть (были?) веские причины для такой конфигурации, см. *http://www.modssl.org/docs/2.8/ssl_faq.html#ToC49* Но если вы желаете использовать аутентификацию NTLM, вам необходимо использовать `keepalive`. Если вы разкомментируете

`SetEnvIf` полностью, вы сможете заставить IE аутентифицироваться при помощи аутентификации Windows с использованием SSL в Apache на платформе Win32 с загруженным модулем `mod_auth_sspi`.

Принудительное использование SSL при доступе

После того, как вы настроили SSL для большей безопасности хранилища, возможно, вы пожелаете отключить обычный доступ не через SSL (http), и оставить доступ только через https. Для этого вы должны добавить ещё одно указание в блок `<Location>` Subversion: `SSLRequireSSL`.

Пример блока `<Location>` будет выглядеть так:

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

3.7.10. Использование клиентских сертификатов с виртуальными SSL-узлами

Прислано в список рассылки TortoiseSVN Найджелом Грином (Nigel Green). Спасибо!

В некоторых серверных конфигурациях вам может быть необходимо настроить один сервер, включающий 2 виртуальных SSL-узла: один для общего доступа через веб (не требующий клиентского сертификата), другой должен быть безопасным, с запросом клиентского сертификата, на котором и работает сервер Subversion.

Добавление указания `SSLVerifyClient Optional` в секции *per-server* конфигурации Apache (т.е. вне всех блоков `VirtualHost` и `Directory`) заставляет Apache запрашивать сертификат клиента при начальном установлении связи с использованием SSL. Из-за ошибки в `mod_ssl` необходимо, чтобы сертификат запрашивался на этой стадии, поскольку это не сработает при повторном установлении SSL-соединения.

Решением будет добавление следующего указания к папке виртуального узла, к которой желательно ограничить доступ для использования Subversion

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

Это указание предоставляет доступ к папке только в случае, если сертификат клиента был получен и проверен успешно.

Резюмируя, соответствующие строки конфигурации Apache выглядят следующим образом:

```
SSLVerifyClient Optional

### Конфигурация ОБЩЕДОСТУПНОГО виртуального узла
### (не запрашивающего сертификат)

<VirtualHost 127.0.0.1:443>
```

```
  <Directory "путь_к_общедоступному_файловому_корню">
  </Directory>
</VirtualHost>

### Конфигурация виртуального узла SUBVERSION
### (требующего клиентский сертификат)
<VirtualHost 127.0.0.1:443>
  <Directory "корневой путь узла subversion">
    SSLRequire %{SSL_CLIENT_VERIFY} eq "SUCCESS"
  </Directory>

  <Location /svn>
    DAV svn
    SVNParentPath /путь_к_хранилищу
  </Location>
</VirtualHost>
```

Глава 4. Руководство по ежедневному использованию

В этом документе описано использование TortoiseSVN, как оно происходит изо дня в день. Это *не* введение в системы управления версиями, и *не* введение в Subversion (SVN). Эта глава более похожа на то, к чему вы можете обратиться, когда вы приблизительно знаете, что нужно сделать, но не помните точно, как это делается.

Если вам нужно введение в управление версиями с использованием Subversion, тогда мы рекомендуем прочитать чудесную книгу[1]: *Управление версиями в Subversion* [http://svnbook.red-bean.com/].

Работа над этим документом продолжается, также как и над TortoiseSVN и Subversion. Если вы нашли какие-нибудь ошибки, пожалуйста, сообщите о них в список рассылки, чтобы мы могли обновить документацию. Некоторые копии экранов в Руководстве по ежедневному использованию могут не соответствовать текущему состоянию программы. Пожалуйста, простите нас: мы работаем над TortoiseSVN в своё свободное время.

Для того, чтобы пользы от Руководства по ежедневному использованию:

- У вас уже должен быть установлен TortoiseSVN.
- Вы должны быть знакомы с системами управления версиями.
- Вы должны знать основы Subversion.
- Вы должны установить и настроить сервер и/или иметь доступ к хранилищу Subversion.

4.1. Приступая к работе

4.1.1. Пометки на значках

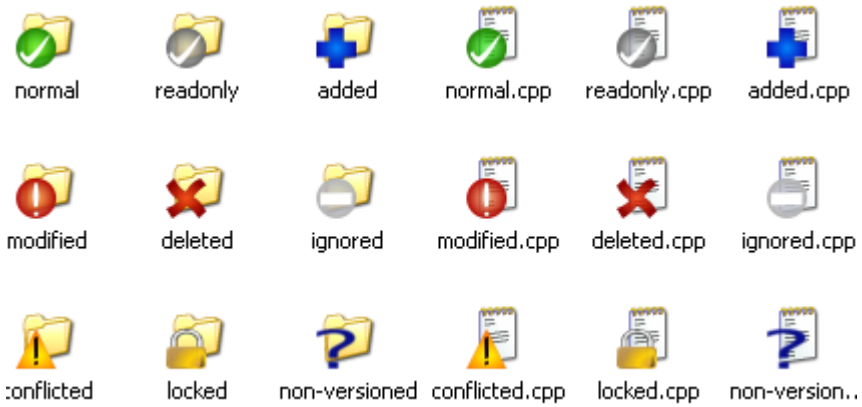


Рисунок 4.1. Проводник с пометками на значках

Одной из наиболее заметных особенностей TortoiseSVN являются пометки на значках, которые появляются для файлов в рабочей копии. Они сразу же показывают, какие файлы были изменены. Что обозначают различные пометки, можно посмотреть в Раздел 4.7.1, «Пометки на значках».

[1] В других частях этого документа эта книга фигурирует как "Книга о Subversion". Она доступна на английском, русском и некоторых других языках - прим. переводчика

4.1.2. Контекстные меню

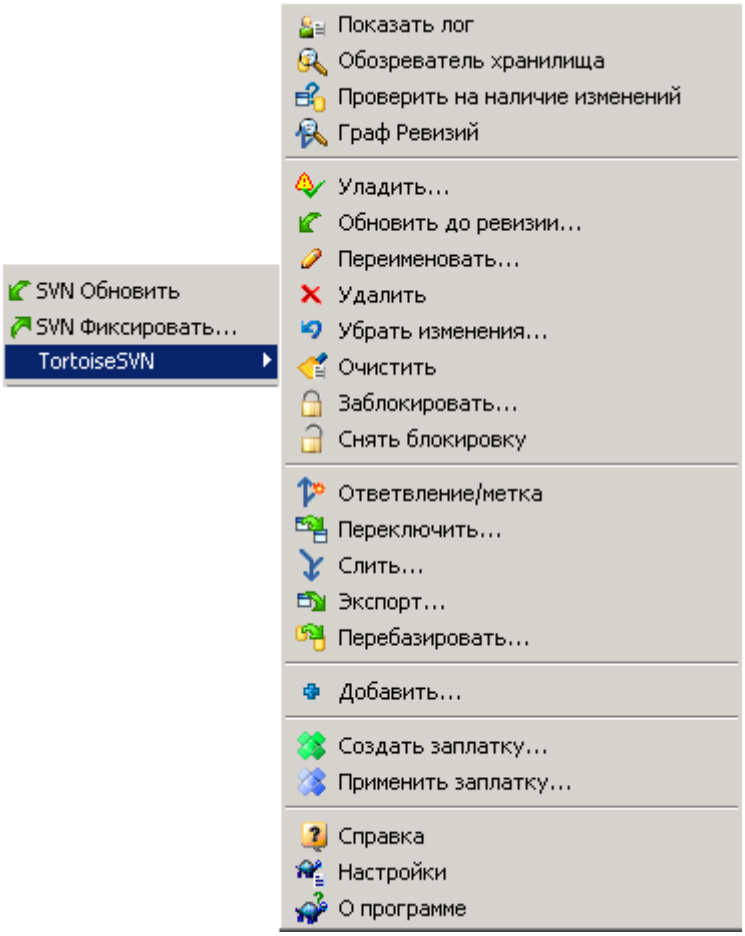


Рисунок 4.2. Контекстное меню для папки, находящейся под управлением версиями

Все команды TortoiseSVN вызываются из контекстного меню Проводника Windows. Большинство из них видно непосредственно, когда вы щелкаете правой клавишей мыши на файле или папке. Список доступных команд зависит от того, находятся ли файл, папка или их родительская папка под управлением версиями, или нет. Вы также можете увидеть меню TortoiseSVN как часть меню "Файл" Проводника.

Подсказка

Некоторые редко используемые команды доступны только в расширенном контекстном меню. Для его вызова необходимо удерживать клавишу **Shift** при правом щелчке.

В некоторых случаях вы можете видеть в меню несколько пунктов TortoiseSVN. Это не ошибка!

Рисунок 4.3. Меню "Файл" Проводника для ярлыка в версированной папке

Этот пример для неверсированного ярлыка внутри версиорованной папки, и меню "Файл" Проводника содержит *три* вхождения TortoiseSVN. Одно из них для папки, одно для ярлыка и одно для объекта, на который указывает ярлык. Для того, чтобы можно было отличить их друг от друга, значки имеют пометку в нижнем правом углу, показывающую, к какому объекту относится это вхождение меню: к файлу, к папке, к ярлыку или к нескольким выделенным элементам.

Если вы используете Windows 2000, вы обнаружите, что контекстные меню отображаются в виде простого текста, без значков в меню, показанных выше. Мы понимаем, что это работало в предыдущих версиях, но Microsoft поменяла в Vista способ работы обработчиков значков, что потребовало от нас использования другого метода отображения, который, к сожалению, не работает в Windows 2000.

4.1.3. Перетаскивание мышью

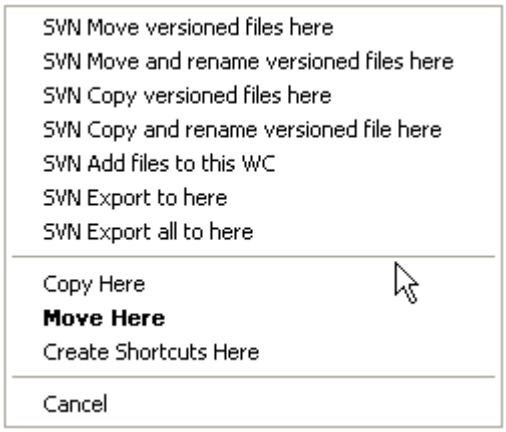


Рисунок 4.4. Меню при перетаскивании правой клавишей мыши для папки под управлением версиями

Другие команды становятся доступны как возможные варианты действий (обработчики перетаскивания) при перетаскивании правой клавишей мыши файлов или папок на новое место внутри рабочей копии, или при перетаскивании правой клавишей неверсированных файлов или папок в какую-либо папку, находящуюся под управлением версиями.

4.1.4. Общие клавиатурные сокращения

У некоторых общих операций есть хорошо известные клавиатурные сокращения Windows, но они не появляются на кнопках или в меню. Если у вас не получается выполнить что-то очевидное, вроде обновления вида, посмотрите здесь.

F1
: Конечно же, справка

F5
: Обновление текущего вида. Это, наверное, одна из самых полезных одноклавишных команд. Например: в Проводнике она обновляет пометки на значках в вашей рабочей копии; в диалоге фиксации она перепросматривает рабочую копию для обнаружения того, что ещё можно зафиксировать; в диалоге 'Журнал ревизий' она вновь связывается с хранилищем для проверки последних изменений.

Ctrl-A
: Выделить все. Она может быть использована, если вы получили сообщение об ошибке, и желаете скопировать его и вставить в письмо электронной почты. Используйте Ctrl-A для выбора сообщения об ошибке, а затем ...

Ctrl-C
: ... Скопируйте выделенный текст.

4.1.5. Аутентификация

Если вы пытаетесь подключиться к хранилищу, защищённому паролем, появится диалог аутентификации.

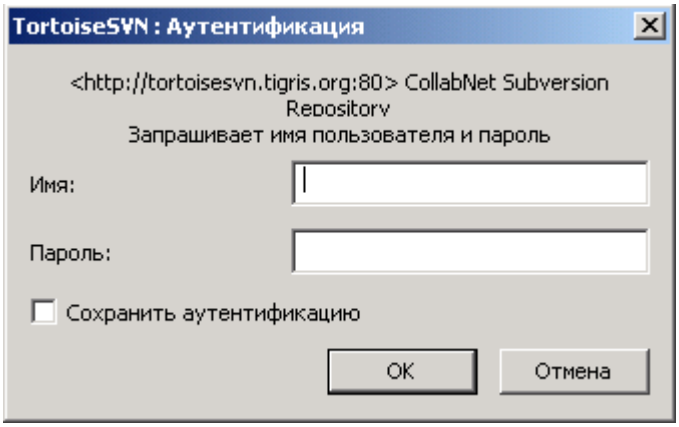


Рисунок 4.5. Диалог аутентификации

Введите ваше имя пользователя и пароль. При помощи флажка можно сделать так, чтобы эти данные сохранялись TortoiseSVN в папке по умолчанию Subversion: `%APPDATA%\Subversion\auth` в трёх подпапках:

- `svn.simple` содержит учётные данные для базовой аутентификации (имя пользователя/пароль).
- `svn.ssl.server` содержит серверные сертификаты SSL.
- `svn.username` содержит учётные данные для аутентификации только по имени пользователя (без пароля).

Очистить кэш аутентификации для *всех* серверов можно со страницы Сохранённые данные диалога настроек TortoiseSVN. Там находится кнопка, очищающая все закэшированные данные аутентификации из папок `auth` Subversion, так же как и любые аутентификационные данные, сохранённые в реестре более ранними версиями TortoiseSVN. Смотрите Раздел 4.30.6, «Настройки сохранённых данных ».

Некоторым нравится, чтобы данные аутентификации удалялись при выходе из Windwos или при выключении компьютера. Этого можно добиться при помощи выполняемого при выключении скрипта, удаляющего папку `%APPDATA%\Subversion\auth`, например,

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

Описание того, как установить такой скрипт, можно найти на *windows-help-central.com* [http://www.windows-help-central.com/windows-shutdown-script.html].

За более полной информацией о том, как настроить ваш сервер для аутентификации и управления доступом, обращайтесь к Раздел 3.5, «Доступ к хранилищу»

4.1.6. Разворачивание окон

Многим диалогам TortoiseSVN необходимо показывать большой объём информации, но довольно часто бывает полезно развернуть окно только по высоте, или только по ширине, вместо того, чтобы разворачивать его на весь экран. Для удобства эти функции реализованы путём быстрого вызова через стандартную кнопку Развернуть. Щёлкните по ней средней кнопкой мыши для разворачивания по вертикали, и правой кнопкой - для разворачивания по горизонтали.

4.2. Импорт данных в хранилище

4.2.1. Импорт

Если вы импортируете в существующее хранилище, которое уже содержит несколько проектов, то структура хранилища будет уже вполне определена. Если вы импортируете данные в новое хранилище, то имеет смысл подумать о том, как оно будет организовано. Прочтите Раздел 3.1.5, «Организация данных в хранилище» для дополнительных рекомендаций.

Этот раздел описывает команду импорта Subversion, которая предназначена для импорта иерархии папок в хранилище за один подход. Несмотря на то, что она это делает, у неё есть несколько недостатков:

- Нет способа выбрать включаемые файлы и папки, кроме как применяя настройки глобального исключения.
- Импортированная папка не становится рабочей копией. Вы должны выполнить извлечение для копирования файлов обратно с сервера.
- Можно легко импортировать совсем на не тот уровень папок в хранилище.

По этим причинам мы рекомендуем, чтобы вы вообще не использовали команду импорта, за исключением, конечно, выполнения двухшагового метода, описанного в Раздел 4.2.2, «Импорт на месте». Но поскольку вы уже здесь, вот как в основном работает импорт...

Перед тем, как вы импортируете ваш проект в хранилище, вам следует:

1. Удалить все файлы, которые не нужны для сборки проекта (временные и создаваемые компилятором файлы, такие как *.obj, скомпилированные исполняемые файлы, ...)
2. Упорядочить файлы в папки и подпапки. Хотя возможно переименовать/переместить файлы и позже, настоятельно рекомендуется, чтобы структура вашего проекта была сформирована перед импортом!

Теперь выберите самую верхнюю папку в структуре папок проекта в Проводнике Windows и сделайте правый щелчок для открытия контекстного меню. Выберите команду TortoiseSVN → Импорт..., которая откроет диалоговое окно:

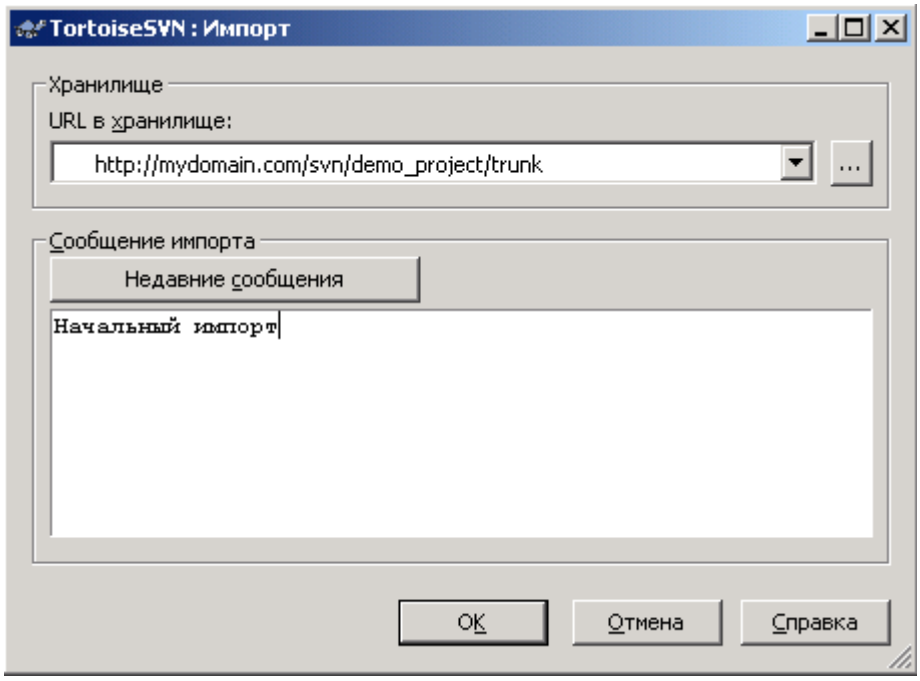


Рисунок 4.6. Диалог импорта

В этом диалоге вы должны ввести URL места в хранилище, в которое вы желаете импортировать проект. Очень важно понять, что сама импортируемая локальная папка в хранилище не появится, только её содержимое. Например, если у вас имеется следующая структура:

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

и вы импортируете `C:\Проекты\Widget` в `http://mydomain.com/svn/trunk`, то вы можете быть удивлены, что ваши подпапки попадут непосредственно в `ствол`, вместо того, чтобы остаться в подпапке `Widget`. Вам необходимо указать подпапку как часть URL,`http://mydomain.com/svn/trunk/Widget-X`. Обратите внимание: команда импорта автоматически создаёт подпапки в хранилище, если они не существуют.

Сообщение импорта используется как сообщение журнала.

По умолчанию, файлы и папки, соответствующие глобальным шаблонам игнорирования, *не импортируются*. Для изменения этого поведения, вы можете использовать флажок Включить игнорируемые файлы. Более детальная информация по установке глобальных шаблонов игнорирования приведена в Раздел 4.30.1, «Общие настройки».

Как только вы нажмёте ОК, TortoiseSVN импортирует в хранилище полностью всё дерево папок со всеми файлами. Сейчас проект помещён в хранилище под управление версиями. Пожалуйста помните, что та папка, которую вы импортировали, *НЕ НАХОДИТСЯ* под управлением версиями! Для получения *рабочей копии*, находящейся под управлением версиями, вы должны произвести извлечение только что импортированной версии. Или продолжайте чтение для того, чтобы узнать как импортировать папку на месте.

4.2.2. Импорт на месте

Предполагая, что хранилище уже у вас есть, и вы желаете добавить в него новую папку со всей её структурой, просто выполните следующие шаги:

1. При помощи обозревателя хранилища создайте новую папку проекта непосредственно в хранилище.
2. Извлеките эту новую папку поверх той папки, которую вы желаете импортировать. Появится предупреждение о том, что локальная папка не пуста. Теперь у вас есть версированная папка верхнего уровня с неверсированным содержимым.
3. Воспользуйтесь командой TortoiseSVN → Добавить... на новой версированной папке для добавления части или всего содержимого. Вы можете добавлять и убирать файлы, задавать свойства `svn:ignore` для папок и производить любые другие необходимые вам изменения.
4. Зафиксируйте папку верхнего уровня, и у вас получится новое версированное дерево и локальная рабочая копия, созданная из вашей существующей папки.

4.2.3. Особые файлы

Иногда вам необходимо внести под управление версиями файл с данными, отличающимися для каждого пользователя. Это означает, что у вас есть файл, который каждый разработчик/пользователь должен изменить для соответствия собственным настройкам. Но версирование такого рода файлов затруднено, поскольку каждый пользователь будет каждый раз фиксировать свои изменения в хранилище.

В таких случаях мы предлагаем использовать *шаблонные* файлы. Вы создаёте файл, содержащий все необходимые разработчикам данные, добавляете этот файл под управление версиями и пусть разработчики извлекают этот файл. После извлечения каждый разработчик должен *сделать копию*

этого файла и переименовать созданную копию, после чего эту копию можно изменять без проблем.

Например, вы можете посмотреть на скрипт сборки TortoiseSVN. Он вызывает файл с именем `TortoiseVars.bat`, который не существует в хранилище - только `TortoiseVars.tmpl`. `TortoiseVars.tmpl` - это шаблон файла, который каждый разработчик получает из хранилища и переименовывает его в `TortoiseVars.bat`. Внутри этого файла мы поместили комментарии, так что пользователи видят, какие строки они должны отредактировать и изменить в соответствии с их локальными настройками, чтобы это заработало.

И чтобы не сбивать с толку пользователей, мы также добавили файл `TortoiseVars.bat` в список игнорирования для его родительской папки, т.е. мы устанавливаем свойство Subversion `svn:ignore` так, чтобы оно включало это имя файла. Таким образом, оно не показывается как неверсированное при каждой фиксации.

4.3. Извлечение рабочей копии

Для получения рабочей копии вам надо произвести *извлечение* из хранилища.

Выберите в Проводнике Windows папку, в которой хотите разместить вашу рабочую копию. Сделайте правый щелчок для вызова контекстного меню и выберите команду TortoiseSVN → Извлечь..., после чего появится следующий диалог:

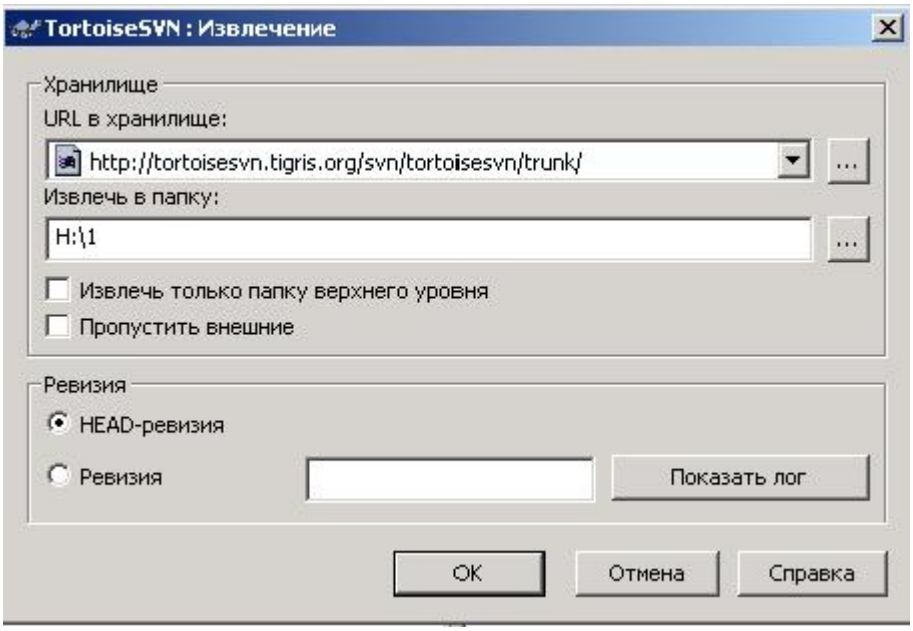


Рисунок 4.7. Диалог извлечения

Если ввести имя пока ещё несуществующей папки, то она будет создана.

4.3.1. Глубина извлечения

Вы можете выбрать *глубину* охвата при извлечении, определяющую глубину рекурсии для дочерних папок. Если вам необходимы всего лишь несколько разделов большого дерева, вы можете извлечь только папку верхнего уровня, после чего обновить выбранные папки рекурсивно.

Полностью рекурсивно
: Извлекает всё дерево целиком, включая все дочерние папки и подпапки.

Непосредственные потомки, включая папки
: Извлекает указанную папку, включая все файлы и дочерние папки, но не включая содержимое дочерних папок.

Только потомки-файлы
: Извлекает указанную папку, включая все файлы, но не включая дочерние папки.

Только этот элемент
: Извлекает только указанную папку. Не извлекает в неё файлы и дочерние папки.

Рабочая копия
: Сохраняет глубину, указанную в рабочей копии. Этот параметр не используется в диалоге извлечения, но является используемым по умолчанию во всех других диалогах, в которых присутствует глубина охвата.

Исключить
: Используется для уменьшения глубины рабочей копии после того, как папка уже была заполнена. Эта опция доступна только в диалоге Обновить до ревизии.

Если вы извлекли разреженную рабочую копию (т.е. выбрали для глубины извлечения значение, отличное от `полностью рекурсивно`), вы можете получить дополнительные неизвлечённые подпапки, используя обозреватель хранилища (Раздел 4.24, «Обозреватель хранилища») или диалог проверки на наличие изменений (Раздел 4.7.3, «Локальный и удалённый статус»).

При использовании обозревателя хранилища: выполните правый щелчок на извлечённой папке, после чего вызовите TortoiseSVN → Обозреватель хранилища. Найдите подпапку, которую вы желали бы добавить в рабочую копию, и выберите Контекстное меню → Обновить элемент до ревизии. Этот пункт меню будет виден, только если выбранный элемент пока ещё не существует в вашей рабочей копии, при том, что его родительская папка уже была извлечена.

В диалоге проверки на наличие изменений, сначала нажмите кнопку Проверить хранилище. Все существующие в хранилище файлы и папки, которые пока ещё не были вами извлечены, будут показаны в диалоге как `добавлен отдалённо`. Выполните правый щелчок на папке (папках), которую вы желали бы добавить в вашу рабочую копию, и примените команду Контекстное меню → Обновить.

Эта возможность очень полезна, когда вы желаете извлечь только некоторые части из большого дерева, но при этом желаете сохранить удобство обновления одной рабочей копии. Предположим, у вас есть большое дерево, в котором есть подпапки от `Проект01` до `Проект99`, и вы желаете извлечь только `Проект03`, `Проект25` и `Проект76/ПодПроект`. Выполните следующие шаги:

1. Извлеките родительскую папку с глубиной «Только этот элемент» Теперь у вас еть пустая папка верхнего уровня.

2. Выберите новую папку и воспользуйтесь пунктом TortoiseSVN → Обозреватель хранилища для отображения содержимого хранилища.

3. Выполните правый щелчок на `Проект03` и Контекстное меню → Обновить элемент до ревизии.... Не изменяя настройки по умолчанию, нажмите на ОК. Теперь эта папка у вас полностью заполнена.

   Повторите ту же процедуру для `Проект25`.

4. Перейдите к `Проект76/ПодПроект` и сделайте то же самое. На этот раз обратите внимание, что в папке `Проект76` нет ничего, кроме `ПодПроект`, который полностью заполнен. Subversion создала для вас все промежуточные папки, не заполняя их.

Изменение глубины рабочей копии

После того, как вы извлекли рабочую копию до определённой глубины, вы можете позже изменить эту глубину, чтобы получить больше или меньше содержимого при помощи Контекстное меню → Обновить до ревизии....

Использование старого сервера

Серверы версий до 1.5 не понимают запросов глубины рабочей копии, поэтому они не всегда могут эффективно обработать такие запросы. Команда по-прежнему будет работать, но более старый сервер может отправить все данные, оставляя клиенту заботу отфильтровывать то, что не требуется, и что может означать передачу большого количества данных по сети. По возможности, обновите свой сервер до версии 1.5.

Если проект содержит ссылки на внешние проекты, которые вы *не хотите* извлекать в этот раз, используйте флажок Пропустить внешние

Важно

Если отмечен флажок Пропустить внешние, или если вы желаете изменить значение глубины, то вы должны будете выполнить обновление вашей рабочей копии с использованием команды TortoiseSVN → Обновить до ревизии... вместо TortoiseSVN → SVN Обновить.... Стандартное обновление включает все внешние ссылки и сохраняет существующее значение глубины.

Рекомендуется извлекать только `ствол (trunk)` из дерева папок, или его подветку. Если вы в URL укажете родительский путь для дерева папок, это может привести к полному заполнению вашего жёсткого диска, поскольку вы получите копию всего дерева хранилища, включая каждое ответвление и метку вашего проекта!

Экспорт

Иногда бывает необходимо создать локальную копию без всех этих папок `.svn`, например, для создания запакованного архиватором файла (zipped tarball) исходного кода. Прочтите Раздел 4.26, «Экспорт рабочей копии Subversion», чтобы узнать, как это сделать.

4.4. Фиксация ваших изменений в хранилище

Отправка изменений, сделанных в вашей рабочей копии, называется *фиксацией*. Но перед фиксацией вы должны убедиться, что рабочая копия находится в актуальном состоянии. Можно либо сразу использовать TortoiseSVN → SVN Обновить..., либо можно сначала вызвать TortoiseSVN → Проверить на наличие изменений для просмотра файлов, которые были изменены локально или на сервере.

4.4.1. Диалог фиксации

Если ваша рабочая копия в актуальном состоянии, и конфликты отсутствуют, то вы готовы к фиксации ваших изменений. Выберите любой файл и/или папку, которые вы хотите зафиксировать, и вызовите TortoiseSVN → SVN SVN Фиксировать....

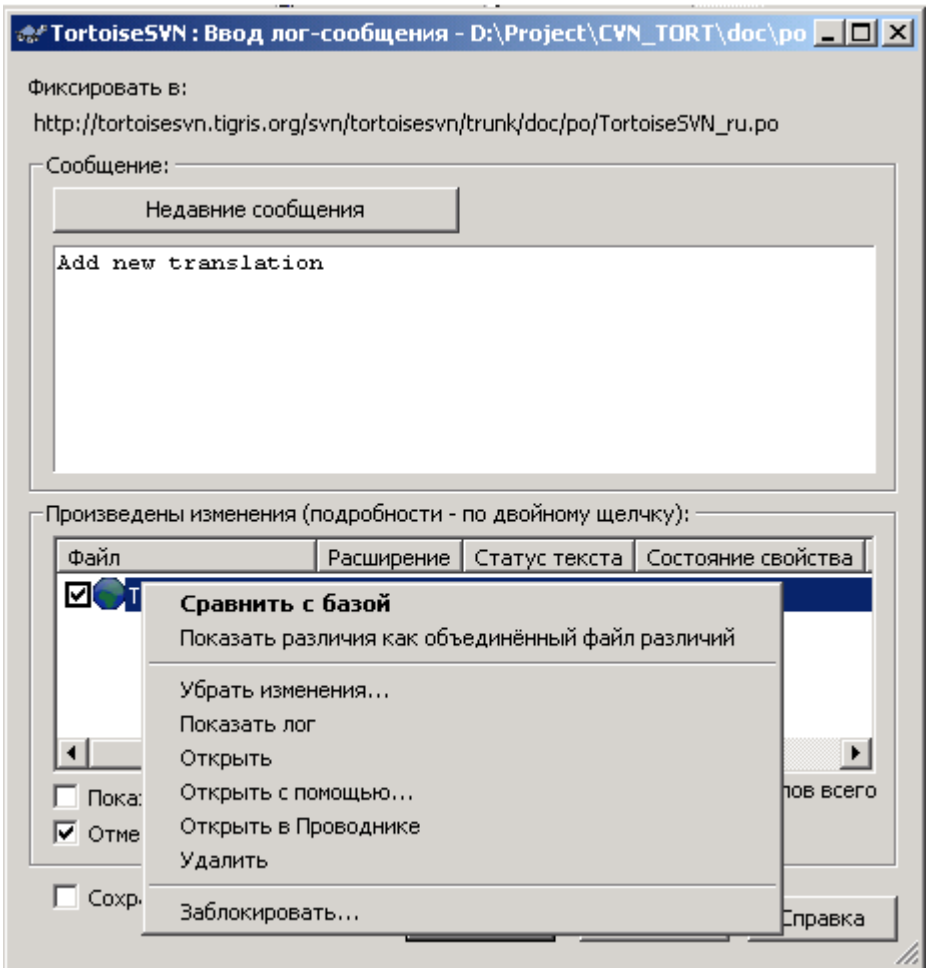


Рисунок 4.8. Диалог фиксации

Диалог фиксации покажет все изменённые файлы, включая добавленные, удалённые и неверсированные. Если вы не хотите фиксировать какой-то изменённый файл, просто разотметьте его. Если вы хотите включить неверсированный файл, тогда пометьте его для добавления к фиксации.

Элементы, переключенные на другие пути в хранилище, отмечаются маркером `(s)`. Возможно, вы переключили что-то, пока работали в ответвлении, и забыли переключить обратно в основной ствол. Этот маркер - предупредительный сигнал!

Что фиксировать: файлы или папки?

Когда вы фиксируете файлы, диалог фиксации отображает только выбранные вами файлы. Когда вы фиксируете папки, диалог фиксации самостоятельно отберёт изменённые файлы. Если вы забудете о созданном вами новом файле, при фиксации папки он всё равно будет обнаружен. Фиксация папки *не означает*, что каждый файл будет помечен как изменённый, это просто способ облегчить вам жизнь путём выполнения в помощь вам большего объёма работы.

Если вы изменили файлы, которые были включены из другого хранилища с использованием `svn:externals`, то эти изменения не могут быть включены в эту же атомарную фиксацию. Если это случиться, появится предупредительный символ под списком файлов, и подсказка объяснит, что эти внешние файлы должны быть зафиксированы отдельно.

Большое количество неверсированных файлов в диалоге фиксации

Если вы считаете, что в диалоге фиксации показывается слишком много неверсированных файлов (вроде генерируемых компилятором или резервных копий редактора), есть несколько способов с этим справиться. Вы можете:

- добавить файл (или шаблон его имени) в список игнорируемых файлов на странице настроек. Это затронет все ваши рабочие копии.
- добавить файл в список `svn:ignore`, используя TortoiseSVN → Добавить в список игнорирования. Это затронет только папку, для которой вы устанавливаете свойство `svn:ignore`. Вы можете изменить свойство `svn:ignore` для папки при помощи диалога свойств Subversion.

Для дополнительной информации прочтите Раздел 4.13, «Игнорирование файлов и папок».

Двойной щелчок на любом изменённом файле в диалоге фиксации запускает внешнюю утилиту сравнения для показа произведённых изменений. Как видно на рисунке, в контекстном меню доступны и другие возможности. Вы также можете перетащить файлы отсюда в другое приложение, такое как текстовый редактор или IDE.

Вы можете отмечать или разотмечать элементы при помощи щелчка на флажке, находящегося слева от этого элемента. Для папок возможно использование **Shift**-отметки для того, чтобы применить действие рекурсивно.

Отображаемые в нижней панели столбцы можно настроить. Если щёлкнуть правой кнопкой на заголовке любого столбца, появится контекстное меню, позволяющее выбрать отображаемые столбцы. Вы также можете изменить ширину столбца при помощи указателя перемещения, который появляется при прохождении указателя мыши через границу столбца. Эти настройки сохраняются, так что вы увидите заголовки в том же виде и в следующий раз.

По умолчанию, при фиксации изменений все удерживаемые вами блокировки файлов снимаются после успешного завершения операции. Если вы желаете оставить эти блокировки, убедитесь, что флажок Сохранить блокировки отмечен. Значение, используемое по умолчанию для этого флажка, берётся из параметра `no_unlock` конфигурационного файла Subversion. Прочтите Раздел 4.30.1, «Общие настройки», чтобы узнать, как редактировать конфигурационный файл Subversion.

Перетаскивание мышью

Вы можете перетаскивать файлы в диалог фиксации из других мест, при условии, что рабочие копии были извлечены из того же хранилища. Например, у вас может быть огромная рабочая копия с несколькими открытыми окнами проводника для просмотра далеко отстоящих друг от друга папок в иерархии. Если вы хотите избежать фиксации из папки верхнего уровня (с длительным обходом папок для обнаружения изменений), вы можете открыть диалог фиксации для одной папки и перетащить в него элементы из других окон для включения в ту же атомарную фиксацию.

Вы можете перетащить в диалог фиксации неверсированные файлы из рабочей копии, и они будут автоматически добавлены под управление версиями.

Исправление внешних переименований

Иногда файлы переименовываются вне Subversion, и тогда в списке файлов каждый из них присутствует как два: один отсутствующий, другой - неверсированный. Чтобы избежать потери истории файла, необходимо известить Subversion о том, что они связаны. Просто выберите оба файла: и со старым именем (отсутствующий), и с новым именем (неверсированный) и выполните Контекстное меню → Поправить переименование, чтобы обозначить эту пару файлов как переименование.

4.4.2. Группы изменений

Диалог фиксации поддерживает группы изменений Subversion, предназначенных для группировки связанных файлов. Об этой возможности рассказывает Раздел 4.8, «Группы изменений».

4.4.3. Исключение элементов из списка для фиксации

Иногда у вас есть версированные файлы, которые часто изменяются, но которые вы не хотели бы фиксировать на самом деле. Иногда это говорит о дефекте в вашем процессе сборки - почему эти файлы версированы? А не лучше использовать файлы шаблонов? Но иногда это неизбежно. Классическая причина - среда разработки изменяет пометку даты-времени файла проекта при каждой сборке. Файл проекта должен быть версированным, поскольку включает настройки для сборки, но его не нужно фиксировать только из-за того, что пометка даты-времени изменилась.

Чтобы помочь в затруднительных случаях вроде этого, была зарезервирована специальная группа изменений, называемая `ignore-on-commit`. Любой файл, добавленный в эту группу, не будет автоматически отмечаться в диалоге фиксации. Вы по-прежнему можете зафиксировать изменения в этом файле, но вы должны выбрать его в диалоге фиксации вручную.

4.4.4. Сообщения журнала при фиксации

Убедитесь, что ввели сообщение журнала, описывающее фиксируемые изменения. Впоследствии, при просмотре сообщений журнала проекта, это поможет вам увидеть, что и когда происходило. Сообщение может быть настолько длинным или коротким, как вы пожелаете; во многих проектах есть рекомендации о том, что должно быть в него включено, какой должен использоваться язык, а иногда в них даже задаётся строгий формат сообщения.

Вы можете применять простое форматирование в сообщениях журнала, используя соглашения, похожие на употребляемые в электропочте. Чтобы `текст` выглядел по-другому, можно использовать `*текст*` для жирного начертания, `_текст_` для подчёркивания и `^текст^` для курсива.

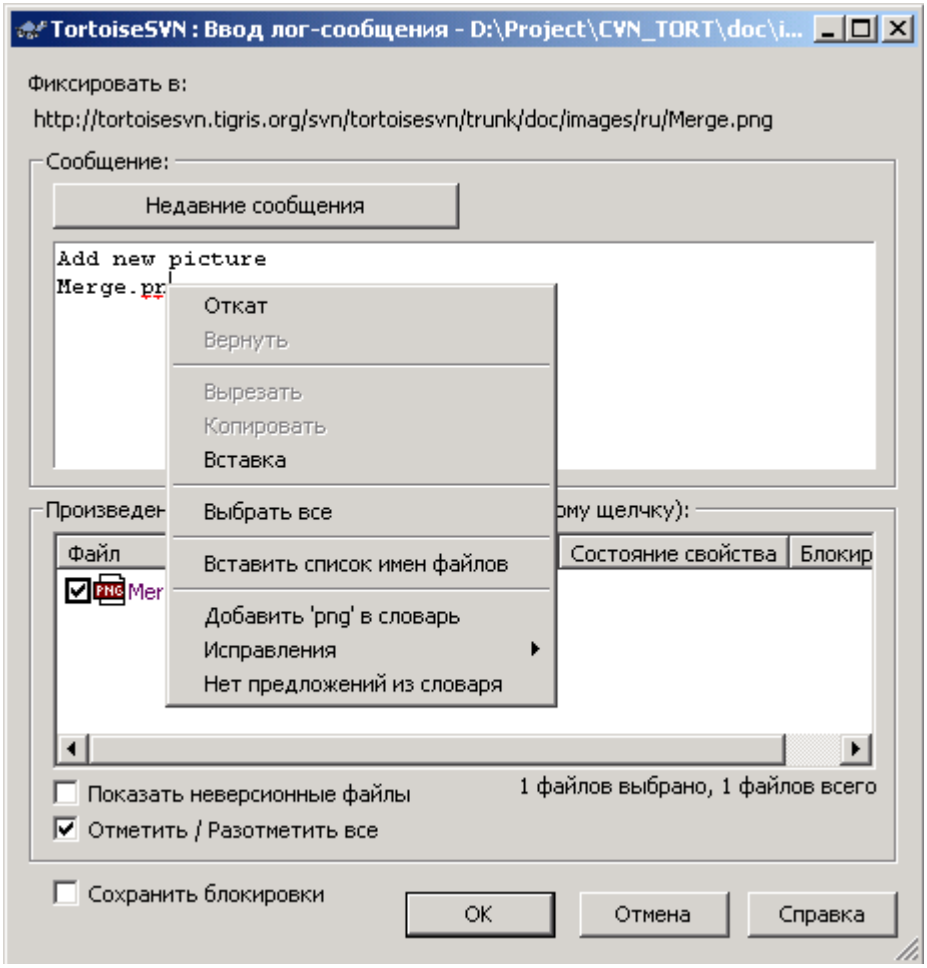


Рисунок 4.9. Проверка правописания в диалоге фиксации

В TortoiseSVN включена проверка правописания, чтобы помочь вам в написании грамотных сообщений журнала. Она выделяет любое неправильно написанное слово. Воспользуйтесь контекстным меню для доступа к предлагаемым исправлениям. Конечно, проверка правописания не знает *каждый* технический термин, который вы можете использовать, так что правильно написанные слова иногда могут показываться как ошибочные. Но не беспокойтесь - вы можете тут же добавить их в ваш персональный словарь при помощи контекстного меню.

Окно сообщения журнала включает также средство автозавершения имён файлов и функций. Оно использует регулярные выражения для извлечения имён классов и функций из (текста) фиксируемых файлов, а также самих имён файлов. Если слово, которое вы вводите, соответствует чему-либо в списке (после того, как вы набрали хотя бы 3 символа, или нажали **Ctrl+пробел**), появляется выпадающий список, позволяющий выбрать полное имя. Регулярные выражения, поставляемые вместе с TortoiseSVN, содержатся в папке `bin` установки TortoiseSVN. Вы можете задать собственные регулярные выражения и сохранить их в файле `%APPDATA%\TortoiseSVN\autolist.txt`. Конечно же, ваш частный список не будет перезаписан при обновлении установки TortoiseSVN. Если вы малознакомы с регулярными выражениями, взгляните на введение по адресу *http://en.wikipedia.org/wiki/Regular_expression*[2], а также на доступные в Сети документацию и учебный курс по адресу *http://www.regular-expressions.info/*.

Вы можете повторно использовать введённые ранее сообщения журнала: достаточно щёлкнуть на кнопку Недавние сообщения и появится список из нескольких последних введённых вами

[2]Есть статья об этом и в русском разделе Википедии: *http://ru.wikipedia.org/wiki/Регулярные_выражения* [http://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B3%D1%83%D0%BB%D1%8F%D1%80%D0%BD%D1%8B%D0%B5_%D0%B2%D1%8B%D1%80%D0%B0%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F] - прим. переводчика

для этой рабочей копии сообщений. Количество сохраняемых сообщений может быть указано в диалоге настроек TortoiseSVN.

Вы можете очистить все сохранённые сообщения фиксации со страницы Сохранённые данные настроек TortoiseSVN, или же вы можете убирать отдельные сообщения непосредственно в диалоге Предыдущие сообщения журнала при помощи клавиши **Delete**.

Если вы желаете включить имена фиксируемых файлов в сообщение журнала, вы можете воспользоваться специальным пунктом из вызываемого в области редактирования контекстного меню: Контекстное меню → Вставить список имен файлов.

Другой способ вставить имена в сообщение журнала - просто перетащить файлы из списка файлов в область редактирования.

Специальные свойства папок

Есть несколько специальных свойств для папок, которые предоставляют возможность более тонко настроить формат сообщений журнала и задать язык, используемый модулем проверки правописания. Прочтите раздел Раздел 4.17, «Установки проекта» для дополнительной информации.

Интеграция с инструментами отслеживания ошибок

Если у вас подключена и действует система отслеживания ошибок, вы можете указать одну или несколько проблем в поле ввода ID ошибки / N проблемы. Если вводится сразу несколько проблем, то они должны быть разделены запятыми. Или, если вы используете основанную на регулярных выражениях поддержку системы отслеживания ошибок, просто добавьте упоминания проблем как часть сообщения журнала. Больше узнать об этом можно, прочитав Раздел 4.28, «Интеграция с системами отслеживания ошибок/проблем».

4.4.5. Ход выполнения фиксации

После нажатия на OK появится диалог, отображающий ход выполнения фиксации.

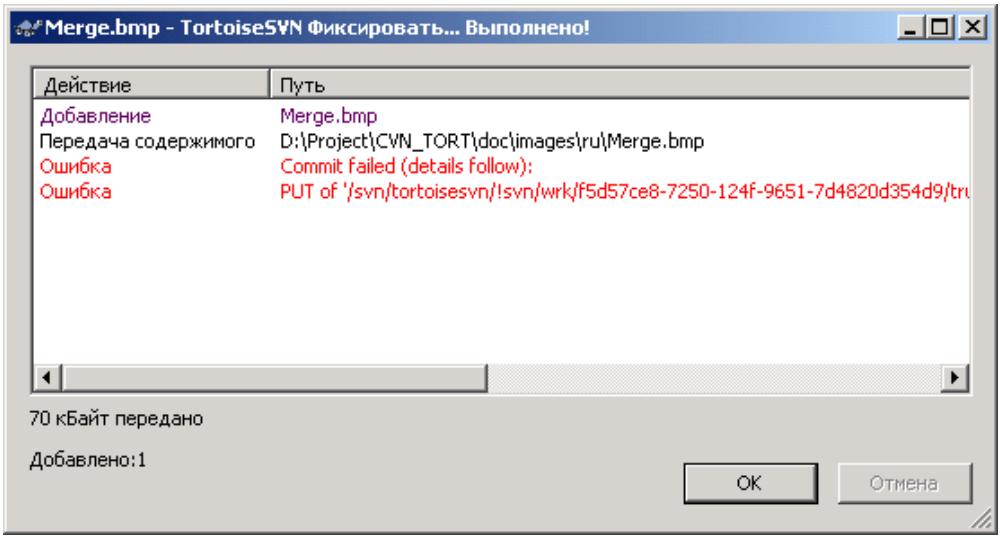


Рисунок 4.10. Диалог выполнения, отображающий ход выполнения фиксации

Различные действия, производимые при фиксации, в окне выполнения обозначаются разным цветом:

Голубой
: Фиксация изменений.

Пурпурный
: Фиксация новых добавлений.

Темно-красный
: Фиксация удалений или перемещений.

Чёрный
: Все другие элементы.

Это используемая по умолчанию цветовая схема, но вы можете настроить эти цвета в диалоге настроек. Для дополнительной информации смотрите Раздел 4.30.1.4, «Настройки цветов в TortoiseSVN».

4.5. Обновление вашей рабочей копии путём внесения изменений, которые сделаны другими

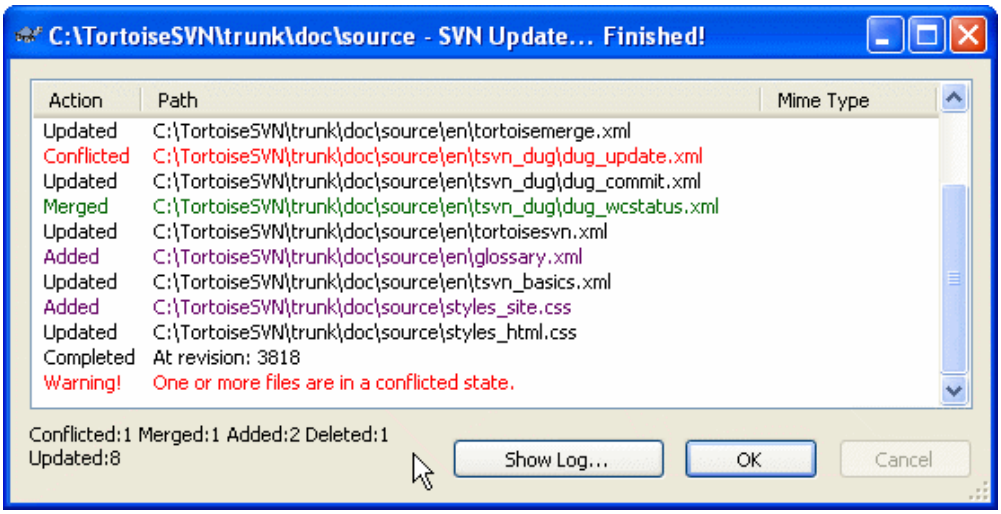


Рисунок 4.11. Окно выполнения, отображающее законченное обновление

Время от времени вы должны обеспечивать внесение изменений, сделанных другими, в вашу рабочую копию. Процесс внесения изменений с сервера в вашу локальную копию называется *обновлением*. Обновляться может одиночный файл, набор выбранных файлов или рекурсивно целые иерархии папок. Для обновления выберите файлы и/или папки, которые вы желаете обновить, щёлкните правой клавишей мыши и выберите из контекстного меню Проводника TortoiseSVN → Обновить.... Появится окно, отображающее ход выполнения обновления. Изменения, сделанные другими, будут слиты с вашими локальными файлами с сохранением любых изменений, которые вы произвели в этих же файлах. Обновление *не оказывает* влияния на хранилище.

Различные действия, производимые при обновлении, в окне выполнения обозначаются разным цветом:

Пурпурный
: Новый элемент, добавленный к вашей рабочей копии.

Темно-красный
: Избыточный элемент, удалённый из вашей рабочей копии, или отсутствующий элемент, заменённый в вашей рабочей копии.

Зеленый
: Изменения из хранилища, успешно слитые с вашими локальными изменениями.

Ярко-красный
: Изменения из хранилища, вызвавшие при слиянии с локальными изменениями конфликт, который вы должны уладить.

Чёрный
: Неизменённый элемент в вашей рабочей копии, обновлённый новой версией из хранилища.

Это используемая по умолчанию цветовая схема, но вы можете настроить эти цвета в диалоге настроек. Для дополнительной информации смотрите Раздел 4.30.1.4, «Настройки цветов в TortoiseSVN».

Если возникают *конфликты* в процессе выполнения обновления (это может произойти, если другие изменили те же строки в тех же файлах, что и вы, и эти изменения не совпадают), тогда диалог показывает эти конфликты красным цветом. Вы можете сделать двойной щелчок на этих строках для запуска внешней утилиты слияния для улаживания конфликтов.

После завершения обновления, в диалоге выполнения под списком файлов показывается сводка о количестве обновлённых, добавленных, удалённых, конфликтующих и т.д. элементов. Эту информацию можно скопировать в буфер обмена при помощи **Ctrl+C**.

Стандартная команда 'Обновить' не имеет параметров и просто обновляет вашу рабочую копию до ведущей (HEAD) ревизии хранилища, и это наиболее типичный способ её использования. Если же вам необходимо точнее управлять процессом обновления, то вместо стандартной команды надо использовать TortoiseSVN → Обновить до ревизии.... Эта команда позволяет обновить вашу рабочую копию до нужной ревизии, а не только до самой последней. Предположим, что ваша рабочая копия соответствует ревизии 100, а вы желаете привести её в состояние, в котором она была в ревизии 50 - просто обновите её до ревизии 50. В этом же диалоге можно выбрать *глубину* обновления текущей папки. Используемые термины описаны в Раздел 4.3.1, «Глубина извлечения». Глубина по умолчанию - Рабочая копия, которая сохраняет существующие установки глубины. Также можно проигнорировать все внешние проекты при обновлении (т.е. такие проекты, связь с которыми организована при помощи `svn:externals`).

Предостережение

После того, как вы обновили файл или папку до нужной ревизии, вы не должны делать изменения в этих файлах. Вы получите ошибку «устарел» при попытке их зафиксировать! Если вы хотите отменить изменения в файле и начать заново с ранней ревизии, вы можете откатиться к предыдущей ревизии в диалоге журнала ревизий. Для дополнительных инструкций и других возможных методов, ознакомьтесь с Раздел B.4, «Возвратиться к старым ревизиям в хранилище (откат)».

Обновить до ревизии может иногда пригодиться для просмотра того, как выглядел ваш проект в какой-то более ранней точке его истории. Но, вообще говоря, обновление отдельных файлов до более ранних ревизий является не очень хорошей идеей, поскольку приводит вашу рабочую копию в противоречивое состояние. Если обновляемый файл был переименован, вы можете даже обнаружить, что он просто исчез из вашей рабочей копии, потому что в более ранней ревизии не существовало файла с таким именем. Обращаем ваше внимание также на то, что на таком элементе показывается обычная зелёная пометка, и он неотличим от файлов, которые находятся в актуальном состоянии.

Если вам нужна просто локальная копия старой ревизии файла, лучше использовать команду Контекстное меню → Сохранить ревизию в... из диалога журнала для этого файла.

Несколько файлов/папок

Если выбрать несколько файлов и папок в Проводнике, и затем выполнить команду Обновить, все эти файлы/папки будут обновляться по очереди. TortoiseSVN обеспечивает, чтобы все файлы/папки из одного хранилища обновлялись точно до одной ревизии!, - даже если между этими обновлениями была выполнена другая фиксация.

Локальный файл уже существует

Иногда попытка обновления завершается неудачей с сообщением о существовании локального файла с таким же именем. Это обычно происходит, когда Subversion пытается извлечь новый версированный файл, и обнаруживает, что неверсированный файл с таким именем уже существует в вашей рабочей папке. Subversion никогда не перезапишет неверсированный файл - он может содержать что-либо, над чем вы работаете, и по стечению обстоятельств называться также, как и файл, недавно зафиксированный другим разработчиком.

Если вы получили это сообщение об ошибке, решение - просто переименовать неверсированный файл. После выполнения обновления, вы можете посмотреть, насколько нужен вам переименованный файл.

Если вы продолжаете получать сообщения об ошибках, воспользуйтесь командой TortoiseSVN → Проверить на наличие изменений для получения списка всех проблемных файлов. Этим способом вы сможете справиться с ними за один раз.

4.6. Улаживание конфликтов

Когда-нибудь у вас возникнет *конфликт* при обновлении/слиянии ваших файлов из хранилища или при переключении рабочей копии на другой URL. Существует два типа конфликтов:

конфликты файлов
: Конфликт файлов возникает, когда двое (или более) разработчиков изменили одни и те же строки файла.

конфликты деревьев
: Конфликт деревьев возникает, когда разработчик переместил/переименовал/удалил файл или папку, которые другой разработчик также переместил/переименовал/удалил или же только изменил.

4.6.1. Конфликты файлов

Конфликт файлов возникает, когда двое или более разработчиков изменили одни и те же строки файла. Поскольку Subversion ничего не знает о вашем проекте, она оставляет улаживание конфликта за разработчиками. При возникновении конфликта вам надо открыть данный файл, и найти строки, начинающиеся с символов <<<<<<<. Область конфликта выглядит следующим образом:

```
<<<<<<< имя файла
    ваши изменения
=======
    код из хранилища
>>>>>>> ревизия
```

Кроме того, для каждого файла с конфликтом Subversion помещает в ту же папку три дополнительных файла:

filename.ext.mine
: Это ваш файл, каким он был в рабочей копии перед выполнением обновления, т.е. без конфликтных отметок. В этом файле содержатся ваши последние в нём изменения, и ничего другого.

filename.ext.rСТАРАЯ_РЕВИЗИЯ
: Это файл, который был базовой ревизией перед обновлением вашей рабочей копии, т.е. это тот файл, который был извлечён до того, как вы начали вносить ваши последние изменения.

filename.ext.rНОВАЯ_РЕВИЗИЯ
: Это файл, который был получен с сервера клиентом Subversion при обновлении вашей рабочей копии. Этот файл соответствует ведущей (HEAD) ревизии хранилища.

Вы можете либо запустить внешнюю утилиту слияния / редактор конфликтов с помощью TortoiseSVN → Редактировать конфликты, либо использовать любой другой редактор для ручного улаживания конфликта. Вы должны решить, как должен выглядеть код, сделать необходимые изменения и сохранить файл.

После этого выполните команду TortoiseSVN → Улажено и зафиксируйте ваши изменения в хранилище. Пожалуйста, помните, что команда 'Улажено' на самом деле конфликты не улаживает. Она только удаляет файлы `filename.ext.mine` и `filename.ext.r*` для того, чтобы вы могли зафиксировать ваши изменения.

Если конфликты возникли в двоичных файлах, Subversion не пытается самостоятельно слить эти файлы. Локальный файл остаётся неизменённым (точно таким, как при последнем вашем изменении) и у вас есть файлы `filename.ext.r*`. Если вы хотите отменить ваши изменения, и сохранить версию из хранилища, используйте команду 'Убрать изменения'. Если вы хотите сохранить вашу версию и переписать версию в хранилище, используйте команду 'Улажено', после чего зафиксируйте вашу версию.

Вы можете использовать команду 'Улажено' для нескольких файлов, если, после щелчка правой клавишей на родительской папке, выбрать TortoiseSVN → Уладить... Появится диалог со списком всех конфликтующих файлов в этой папке, и вы сможете выбрать, какие из них пометить как улаженные.

4.6.2. Конфликты деревьев

Конфликт деревьев возникает, когда разработчик переместил/переименовал/удалил файл или папку, которые другой разработчик также переместил/переименовал/удалил или же только изменил. Есть множество различных ситуаций, которые могут привести к конфликту деревьев, и все они требуют разлишных шагов для улаживания конфликта.

Когда файл удаляется локально в Subversion, файл также удаляется и в локальной файловой системе, поэтому, даже если он участвует в конфликте деревьев, на нём не может быть показана пометка конфликта и вы не можете щёлкнуть на нём правой кнопкой и уладить конфликт. Воспользуйтесь вместо этого диалогом Проверка на наличие изменений для доступа к опции Редактировать конфликты.

TortoiseSVN может помочь обнаружить подходящее место для слияния изменений, но, возможно, потребуются дополнительные усилия по улаживанию конфликтов. Помните, после обновления рабочая БАЗА всегда содержит ту ревизию каждого элемента, которая была у него в хранилище во время обновления. И когда вы убираете изменение после обновления, оно вернётся к состоянию, каким оно было в хранилище, а не к тому, каким оно было, когда вы начали делать свои локальные изменения.

4.6.2.1. Локальное удаление, поступающее при обновлении редактирование

1. Разаработчик A изменяет `Foo.c` и фиксирует это в хранилище.

2. Одновременно разаработчик Б переименовывает `Foo.c` в `Bar.c` в своей рабочей копии, или просто удаляет `Foo.c` или его родительскую папку.

Обновление рабочей копии разработчика Б приводит к конфликту деревьев:

- `Foo.c` удалён из рабочей копии, но помечен как участвующий в конфликте деревьев.
- Если конфликт произошёл из-за переименования, а не из-за удаления, то `Bar.c` помечен как добавленный, но он не содержит изменений, выполненных разработчиком А.

Разработчик Б теперь должен решить, оставлять ли изменения разработчика А. В случае переименования файла, он может перенести изменения из `Foo.c` в переименованный файл `Bar.c` путём слияния. Для простых удалений файлов или папок он может оставить элементы с изменениями разработчика А и отказаться от удаления. Или, пометив конфликт как улаженный и ничего больше не делая, отказаться в итоге от изменений разработчика А.

Диалог редактирования конфликта предлагает слить изменения, если он может найти исходный файл переименованного `Bar.c`. В зависимости от того, откуда было вызвано обновление, может не получиться обнаружить файл-источник.

4.6.2.2. Локальное редактирование, поступающее при обновлении удаление

1. Разработчик А переименовывает `Foo.c` в `Bar.c` и фиксирует это в хранилище.
2. Разработчик Б изменяет `Foo.c` в своей рабочей копии.

Или в случае переименования папки...

1. Разработчик А переименовывает родительскую папку `FooFolder` в `BarFolder` и фиксирует это в хранилище.
2. Разработчик Б изменяет `Foo.c` в своей рабочей копии.

Обновление рабочей копии разработчика Б приводит к конфликту деревьев. Для простого конфликта файлов:

- `Bar.c` добавлен в рабочую копию как обычный файл.
- `Foo.c` помечен как добавленый (с историей) и как участвующий в конфликте деревьев.

Для конфликта папок:

- `BarFolder` добавлена в рабочую копию как обычная папка.
- `FooFolder` помечена как добавленная (с историей) и как участвующая в конфликте деревьев.

  `Foo.c` помечен как добавленный.

Теперь разработчик Б должен решить, принять ли проведённую разработчиком А реорганизацию и слить свои изменения с соответствующим файлом в новой структуре, или просто отменить эти изменения и оставить локальный файл.

Чтобы выполнить слияние своих локальных изменений с такой перетасовкой, разработчик Б сначала должен выяснить, какое имя получил конфликтующий файл `Foo.c` при переименовании/перемещении в хранилище. Это можно сделать при помощи диалога журнала. Затем изменения должны быть слиты вручную, поскольку на данный момент нет способа автоматизировать или даже упростить этот процесс. Как только изменения перенесены, конфликтующий файл больше не нужен и может быть удалён. В этом случае используйте кнопку Удалить в диалоге редактирования конфликтов для наведения порядка и для пометки конфликта как улаженного.

Если разработчик Б решает, что изменения разработчика А были неправильными, то он должен выбрать кнопку Оставить в диалоге редактирования конфликтов. Это помечает конфликтующий файл/папку как улаженные, но изменения разработчика А необходимо будет убрать вручную. Опять же диалог журнала поможет разыскать то, что было перемещено.

4.6.2.3. Локальное удаление, поступающее при обновлении удаление

1. Разработчик А переименовывает `Foo.c` в `Bar.c` и фиксирует это в хранилище.

2. Разработчик Б переименовывает `Foo.c` в `Bix.c`

Обновление рабочей копии разработчика Б приводит к конфликту деревьев:

- `Bar.c` помечен как добавленный с историей.
- `Bar.c` добавлен в рабочую копию со статусом 'нормальный'.
- `Foo.c` помечен как изменённый и участвующий в конфликте деревьев.

Для улаживания этого конфликта разработчик Б должен выяснить, какое имя получил конфликтующий файл `Foo.c` при переименовании/перемещении в хранилище. Это можно сделать при помощи диалога журнала.

Затем разработчик Б должен решить, какое из новых имён файла `Foo.c`оставить - то, которое дал разработчик А или то, в которое он переименовал его сам.

После того, как разработчик Б вручную уладил этот конфликт, конфликт деревьев помечается как улаженный при помощи кнопки в диалоге редактирования конфликтов.

4.6.2.4. Локально отсутствующее, поступающее при обновлении редактирование

1. Разаработчик А, работая в стволе, изменяет `Foo.c` и фиксирует это в хранилище.

2. Разработчик Б, работая в ответвлении, переименовывает `Foo.c` в `Bar.c` и фиксирует это в хранилище.

Слияние изменений разработчика А в стволе с ответвлением в рабочей копии разработчика Б приводит к конфликту деревьев:

- `Bar.c` уже в рабочей копии со статусом 'нормальный'.
- `Foo.c` помечен как отсутствующий и участвующий в конфликте деревьев.

Для улаживания этого конфликта разработчик Б должен пометить файл как улаженный в диалоге редактирования конфликтов, который уберёт его из списка конфликтов. После этого он должен решить, скопировать отсутствующий файл `Foo.c` из хранилища в рабочую копию, или слить изменения разработчика А в файле `Foo.c` в переименованный `Bar.c`, или же проигнорировать изменения, пометив конфликт как улаженный и больше ничего не делая.

Обратите внимание: если вы скопируете отсутсвующий файл из хранилища и после этого пометите конфликт как улаженный, то ваша копия будет снова удалена. Сначала вы должны уладить конфликт.

4.6.2.5. Локальное редактирование, поступающее при слиянии удаление

1. Разработчик А, работая в стволе, переименовывает `Foo.c` в `Bar.c` и фиксирует это в хранилище.

2. Разработчик Б, работая в ответвлении, изменяет `Foo.c` и фиксирует это в хранилище.

Есть эквивалентный случай для перемещения папок, и он всё ещё не обнаруживается в Subversion 1.6 ...

1. Разработчик А, работая в стволе, переименовывает родительскую папку `FooFolder` в `BarFolder` и фиксирует это в хранилище.

2. Разработчик Б, работая в ответвлении, изменяет `Foo.c` в своей рабочей копии.

Слияние изменений разработчика А в стволе с ответвлением в рабочей копии разработчика Б приводит к конфликту деревьев:

- `Bar.c` помечен как добавленный.
- `Foo.c` помечен как изменённый и участвующий в конфликте деревьев.

Теперь разработчик Б должен решить, принять ли проведённую разработчиком А реорганизацию и слить свои изменения с соответствующим файлом в новой структуре, или просто отменить эти изменения и оставить локальный файл.

Чтобы выполнить слияние своих локальных изменений с такой перетасовкой, разработчик Б сначала должен выяснить, какое имя получил конфликтующий файл `Foo.c` при переименовании/перемещении в хранилище. Это можно сделать при помощи диалога журнала для источника слияния. Редактор конфликтов показывает только журнал для рабочей копии, так как ему не известно, какой путь был использован при слиянии, и поэтому вы должны найти его самостоятельно. Затем изменения должны быть слиты вручную, поскольку на данный момент нет способа автоматизировать или даже упростить этот процесс. Как только изменения перенесены, конфликтующий файл больше не нужен и может быть удалён. В этом случае используйте кнопку Удалить в диалоге редактирования конфликтов для наведения порядка и для пометки конфликта как улаженного.

Если разработчик Б решает, что изменения разработчика А были неправильными, то он должен выбрать кнопку Оставить в диалоге редактирования конфликтов. Это помечает конфликтующий файл/папку как улаженные, но изменения разработчика А необходимо будет убрать вручную. Опять же диалог журнала для источника слияния поможет разыскать то, что было перемещено.

4.6.2.6. Локальное удаление, поступающее при слиянии удаление

1. Разработчик А, работая в стволе, переименовывает `Foo.c` в `Bar.c` и фиксирует это в хранилище.
2. Разработчик Б, работая в ответвлении, переименовывает `Foo.c` в `Bix.c` и фиксирует это в хранилище.

Слияние изменений разработчика А в стволе с ответвлением в рабочей копии разработчика Б приводит к конфликту деревьев:

- `Bix.c` помечен как имеющий нормальный (неизменённый) статус.
- `Bar.c` помечен как добавленный с историей.
- `Foo.c` помечен как отсутствующий и участвующий в конфликте деревьев.

Для улаживания этого конфликта разработчик Б должен выяснить, какое имя получил конфликтующий файл `Foo.c` при переименовании/перемещении в хранилище. Это можно сделать при помощи диалога журнала для источника слияния. Редактор конфликтов показывает только журнал для рабочей копии, так как ему не известно, какой путь был использован при слиянии, и поэтому вы должны найти его самостоятельно.

Затем разработчик Б должен решить, какое из новых имён файла `Foo.c`оставить - то, которое дал разработчик А или то, в которое он переименовал его сам.

После того, как разработчик Б вручную уладил этот конфликт, конфликт деревьев помечается как улаженный при помощи кнопки в диалоге редактирования конфликтов.

4.7. Получение информации о статусе

При работе с рабочей копией, вам часто надо понять, какие файлы вы изменили/добавили/удалили или переименовали, или же какие из файлов были изменены и зафиксированы другими.

4.7.1. Пометки на значках

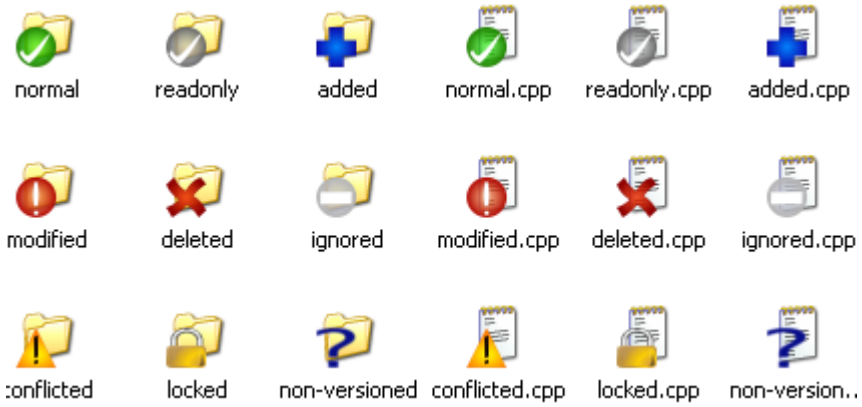


Рисунок 4.12. Проводник с пометками на значках

Теперь, после извлечения рабочей копии из хранилища Subversion, вы можете заметить, что значки в Проводнике Windows немного изменились, и это одна из причин такой популярности TortoiseSVN. TortoiseSVN добавляет так называемую пометку на значке для каждого файла, которая накладывается на исходный значок файла. Пометки различаются и зависят от статуса файла в Subversion.

В свежеизвлечённой рабочей копии все пометки выглядят как зеленая галочка. Это означает, что статус Subversion - *нормальный*.

Как только вы начнете редактировать файл, статус поменяется на *изменено* и пометка станет выглядеть как красный восклицательный знак. Таким образом, вы можете легко увидеть, какие файлы были изменены с момента последнего обновления вашей рабочей копии и нуждаются в фиксации.

Если в процессе обновления возник *конфликт*, тогда пометка меняется на желтый восклицательный знак.

Если вы установили для файла свойство `svn:needs-lock`, Subversion помечает этот файл как доступный только для чтения, пока вы не получите блокировку для этого файла. Эта пометка на файлах означает, что вы должны заблокировать файл перед тем, как начнёте его редактировать.

Если вы владеете блокировкой на файл, и его статус в Subversion *нормальный*, эта пометка напомнит вам, что вы должны разблокировать файл, если вы его не используете, чтобы и другие могли зафиксировать свои изменения в этом файле.

Эта пометка показывает, что некоторые файлы или папки внутри текущей папки запланированы для *удаления* из-под управления версиями, или же что файл, находящийся под управлением версиями, в папке отсутствует.

Символ плюс сообщает о том, что файл или папка запланированы для *добавления* под управление версиями.

Минус говорит о том, что файл или папка *игнорируется* системой управления версиями. Это необязательная пометка.

Этот значок предназначен для файлов, которые не находятся под управлением версиями, но в то же время не являются игнорируемыми. Это необязательная пометка.

Фактически вы можете обнаружить, что не все из этих пометок используются в вашей системе. Это происходит из-за того, что число пометок, доступных в Windows, сильно ограничено, и если вы используете также и старую версию TortoiseCVS, тогда доступных позиций для размещения пометок будет недостаточно. TortoiseSVN попытается быть «Добропорядочным Гражданином (TM)» и ограничивает использование собственных пометок, оставляя эту возможность и другим программам.

Теперь, с появлением всё большего числа клиентов Tortoise (TortoiseCVS, TortoiseHG, ...), ограничение на количество значков становится настоящей проблемой. Для её обхода проект TortoiseSVN представил общий разделяемый набор значков, загружаемый как DLL, который может быть использован всеми клиентами Tortoise. Проверьте в своём клиенте, была ли интегрирована в него поставщиком эта возможность :-)

Для описания соответствия пометок на значках статусам Subversion и других технических подробностей, прочтите Раздел F.1, «Пометки на значках».

4.7.2. Колонки TortoiseSVN в Проводнике Windows

Информация, доступная при помощи пометок (и не только она), может быть отображена как дополнительные столбцы в режиме 'Таблица' Проводника Windows.

Просто сделайте правый щелчок на одном из заголовков колонок, и из контекстного меню выберите Дополнительно... Появиться диалог, в котором можно будет указать, какие колонки и в каком порядке будут отображаться в режиме просмотра «Таблица». Прокрутите вниз, пока не появятся записи, начинающиеся с SVN. Пометьте столбцы, которые вы хотите видеть, и закройте диалог, нажав OK. Столбцы будут добавлены справа от отображаемых в данный момент. Вы можете изменить порядок следования столбцов, перетащив на нужное место, и изменить их размер по необходимости.

Важно

Дополнительные колонки в Проводнике Windows не доступны в Vista после того, как Microsoft решила больше не предоставлять таких колонок для *всех* файлов, а предоставлять только для файлов отдельных типов.

Подсказка

Если вы желаете, чтобы текущая компоновка отображалась во всех ваших рабочих копиях, вы можете сделать её видом по умолчанию.

4.7.3. Локальный и удалённый статус

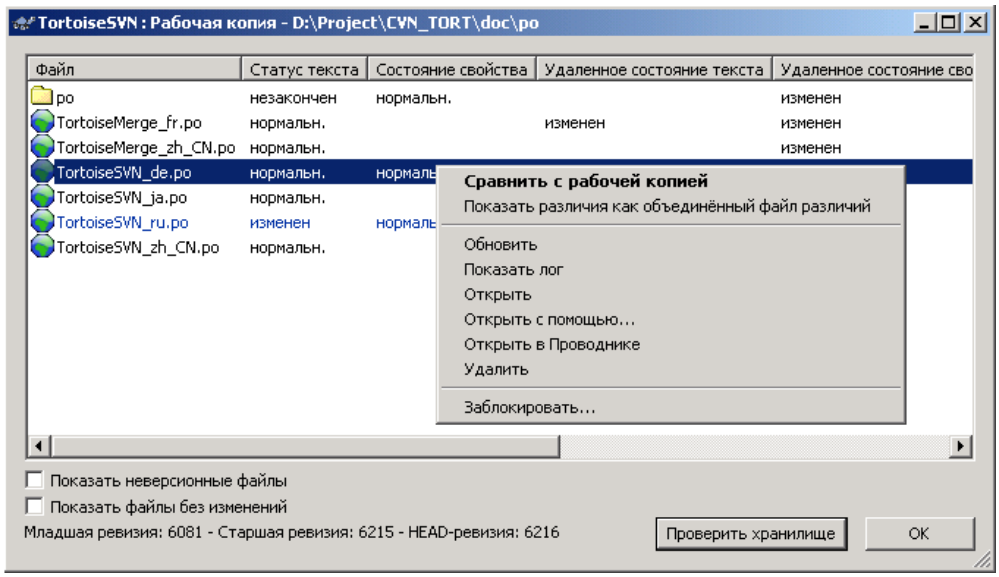


Рисунок 4.13. Проверка на наличие изменений

Часто очень полезно знать, какие файлы вы изменили и какие файлы были изменены и зафиксированы другими. Для этого может пригодиться команда TortoiseSVN → Проверить на наличие изменений.... Появившийся диалог покажет вам каждый файл, который вы каким-либо образом изменили в вашей рабочей копии, а также неверсированные файлы, которые у вас могут быть.

Если вы нажмёте на Проверить хранилище, тогда вы сможете также узнать про изменения в хранилище. Таким образом, перед обновлением вы можете проверить возможность возникновения конфликтов. Вы также можете обновить только выбранные файлы из хранилища без обновления всей папки. По умолчанию, кнопка Проверить хранилище получает удалённый статус только с глубиной извлечения рабочей копии. Если вы желаете увидеть все файлы в хранилище, даже те, которые вы не извлекали, то вам надо удерживать клавишу **Shift** при щелчке на кнопке Проверить хранилище.

Диалог использует различные цвета для обозначения статуса.

Голубой
: Локально изменённые элементы.

Пурпурный
: Добавленные элементы. Элементы, которые были добавлены с историей, имеют знак + в столбце Статус текста, и подсказка показывает, откуда был скопирован элемент.

Темно-красный
: Удалённые или отсутствующие элементы.

Зеленый
: Элементы, изменённые локально и в хранилище. Изменения будут объединены при обновлении. Это *может* привести к конфликту при обновлении.

Ярко-красный
: Элементы, изменённые локально и удалённые в хранилище, или измененные в хранилище и удалённые локально. Эта ситуация *вызовет* конфликт при обновлении.

Чёрный
: Неизменённые и неверсированные элементы.

Это используемая по умолчанию цветовая схема, но вы можете настроить эти цвета в диалоге настроек. Для дополнительной информации смотрите Раздел 4.30.1.4, «Настройки цветов в TortoiseSVN».

Элементы, переключенные на другие пути в хранилище, отмечаются маркером `(s)`. Возможно, вы переключили что-то, пока работали в ответвлении, и забыли переключить обратно в основной ствол. Этот маркер - предупредительный сигнал!

Из контекстного меню диалога вы можете отобразить изменения как различия между файлами. Просмотреть локальные изменения, которые *вы* сделали, можно при помощи Контекстное меню → Сравнить с рабочей копией. Просмотреть изменения в хранилище, сделанные другими, можно используя Контекстное меню → Показать различия как объединенные различия.

Вы также можете отменить изменения в отдельных файлах. Если вы случайно удалили файл, он будет отображаться как *Отсутствующий* и вы можете использовать *Убрать изменения* для его восстановления.

Неверсированные и игнорируемые файлы могут быть отправлены отсюда в корзину при помощи команды Контекстное меню → Удалить. Если вы желаете удалить файлы навсегда (минуя корзину) держите нажатой клавишу **Shift** при щелчке на Удалить.

Если вы желаете детально изучить файл, вы можете перетащить файл отсюда в другое приложение, такое как текстовый редактор или IDE.

Столбцы можно настроить. Если щёлкнуть правой кнопкой на заголовке любого столбца, появится контекстное меню, позволяющее выбрать отображаемые столбцы. Вы также можете изменить ширину столбца при помощи указателя перемещения, который появляется при прохождении указателя мыши через границу столбца. Эти настройки сохраняются, так что вы увидите заголовки в том же виде и в следующий раз.

Если вы работаете над несколькими несвязанными задачами одновременно, вы можете сгруппировать файлы в группы изменений. Прочтите Раздел 4.4.2, «Группы изменений» для более подробной информации.

В нижней части окна показывается сводка о диапазоне ревизий хранилища, встречающихся в вашей рабочей копии. Это ревизии *фиксаций* , а не ревизии *обновлений*; они отображают диапазон ревизий, в которых эти файлы были последний раз зафиксированы, а не ревизий, до которых они были обновлены. Обратите внимание: показанный диапазон относится только к показанным элементам, а не ко всей рабочей копии. Если вам нужны эти же данные для всей рабочей копии, необходимо отметить флажок Показать файлы без изменений.

Подсказка

Если вам нужен плоский вид вашей рабочей копии, т.е. отображение всех файлов и папок со всех уровней иерархии, тогда диалог Проверка на наличие изменений - простейший путь этого добиться. Просто пометьте флажок Показать файлы без изменений для отображения всех файлов в вашей рабочей копии.

Исправление внешних переименований

Иногда файлы переименовываются вне Subversion, и тогда в списке файлов каждый из них присутствует как два: один отсутствующий, другой - неверсированный. Чтобы избежать потери истории файла, необходимо известить Subversion о том, что они связаны. Просто выберите оба файла: и со старым именем (отсутствующий), и с новым именем (неверсированный) и выполните Контекстное меню → Поправить переименование, чтобы обозначить эту пару файлов как переименование.

4.7.4. Просмотр различий

Часто возникает необходимость просмотреть содержимое ваших файлов, чтобы понять, что же было изменено. Вы можете достичь этого путём выбора пункта Различия в контекстном меню TortoiseSVN для нужного файла с изменениями. Это запустит внешнюю программу просмотра различий, которая сравнит текущий файл с нетронутой копией (`BASE`, `Базовой` ревизией), которая была сохранена после последнего извлечения или обновления.

Подсказка

Даже для файлов не из рабочей копии, или когда у вас файл представлен в нескольких версиях, вы всё равно можете посмотреть различия:

Отметьте два файла, которые вы хотите сравнить, в Проводнике (например, используя **Ctrl** и мышку) и выберите из контекстного меню TortoiseSVN команду Различия. Файл, отмеченный последним (тот, который в фокусе, т.е. в прямоугольнике из точек), будет считаться более поздним.

4.8. Группы изменений

В идеальном мире вы всегда работаете не более чем над одной вещью за раз, и ваша рабочая копия содержит только один логический набор изменений. Ну хорошо, теперь обратно к реальности: частенько случается, что вам приходится работать над несколькими несвязанными задачами одновременно, и при взгляде на диалог фиксации вы видите, что все изменения перемешаны друг с другом. *Группы изменений* помогают вам сгруппировать файлы, облегчая понимание того, над чем вы работаете. Конечно, это выполнимо, если изменения не пересекаются. Если две различных задачи затрагивают один и тот же файл, изменения разделить невозможно.

Важно

Группировка изменений в TortoiseSVN доступна только для Windows XP и более поздних версий, поскольку она зависит от возможностей оболочки, которые отсутствуют в Windows 2000. К сожалению, Windows 2000 на данный момент уже заметно устарела, так что не жалуйтесь, пожалуйста.

Вы можете встретить группы изменений в нескольких местах, но наиболее важные из них - это диалог фиксации и диалог проверки на наличие изменений. Давайте начнём в диалоге проверки на наличие изменений, после того, как вы поработали над несколькими возможностями и над многими файлами. Когда вы впервые открываете диалог, все изменённые файлы выводятся одним списком. Предположим, теперь вы желаете навести порядок и сгруппировать эти файлы в соответствии с реализуемыми ими возможностями.

Выберите один или более файлов и воспользуйтесь Контекстное меню → Переместить в группу изменений для помещения выделенного в группу изменений. Изначально ни одной группы изменений не существует, поэтому, когда вы делаете это в первый раз, создаётся новая группа. Назовите её так, чтобы было понятно, для чего она будет использоваться, и нажмите ОК. Теперь диалог изменится, чтобы показать группы элементов.

После того, как вы создали группу изменений, вы можете перетаскивать в неё элементы, как из других групп, так и из Проводника Windows. Перетаскивание из Проводника может пригодиться, поскольку позволяет добавлять элементы в группу до того, как файл будет изменён. Это можно сделать и из диалога проверки на наличие изменений, но для этого придётся включить отображение всех неизменённых файлов.

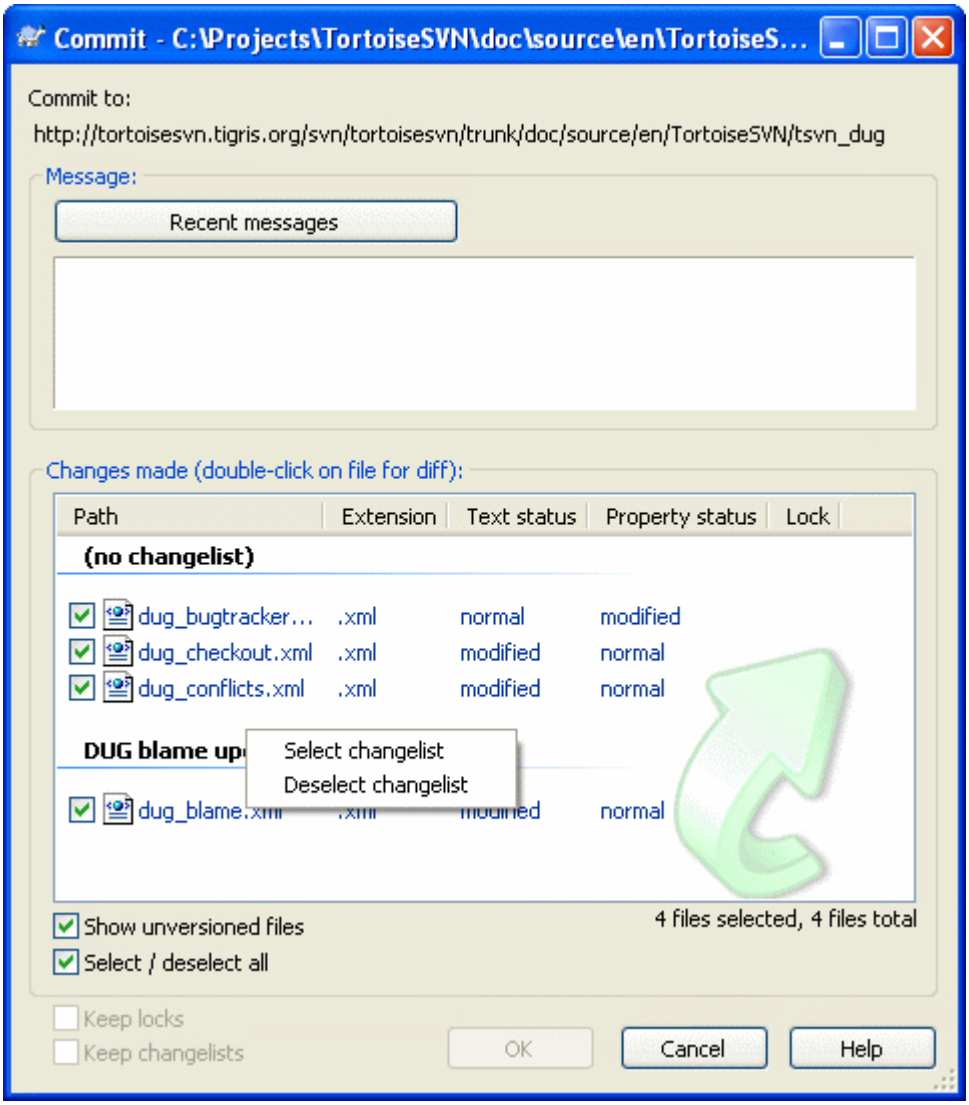


Рисунок 4.14. Диалог фиксации с группами изменений.

В диалоге фиксации вы можете вижеть те же файлы, сгруппированные по группам изменений. Заголовки групп, помимо наглядного отображения разбиения на группы, можно также использовать для выбора файлов для фиксации.

При правом щелчке на заголовке группы в Windows XP появляется контекстное меню, при помощи которого можно (раз)отметить все входящие в группу элементы. В Vista, однако, в контекстном меню необходимости нет: щёлкните на заголовке группы для выбора всех элементов, после чего отметьте один из выбранных элементов - и будут отмечены все.

Одно из имён группы изменений TortoiseSVN резервирует для собственного использования, а именно `ignore-on-commit`. Оно используется для пометки версированных файлов, которые почти никогда не должны фиксироваться, даже если в них есть локальные изменения. Эту возможность описывает Раздел 4.4.3, «Исключение элементов из списка для фиксации».

Обычно ожидается, что после фиксации файлов, входящих в группу изменений, их принадлежность к этой группе больше не нужна. Поэтому по умолчанию файлы автоматически исключаются из групп изменений при фиксации. Если вы желаете оставить файл в группе изменений, воспользуйтесь флажком Сохранить группы изменений, расположенным внизу диалога фиксации.

Подсказка

Группировка изменений - возможность, присущая исключительно локальному клиенту. Создание и ликвидация групп изменений не затрагивает ни хранилище, ни рабочих копий других пользователей. Это просто удобный способ организации ваших файлов.

4.9. Диалоговое окно журнала ревизий

Для каждого сделанного и зафиксированного изменения вы должны ввести сообщение журнала, описывающее это изменение. Таким образом, вы позже сможете выяснить, что было изменено и почему, и у вас будет подробный журнал всего процесса разработки.

Диалог журнала ревизий извлекает и отображает все эти сообщения журнала. Окно разделено на три части:

- В верхней панели находится список ревизий, в которых фиксировались изменения в файле/папке. В этом списке также показываются дата и время, зафиксировавший ревизию пользователь и начало сообщения журнала.

  Отображаемые голубым строки означают, что что-то было скопировано в эту линию разработки (возможно, из ответвления).

- Средняя панель отображает полное сообщение журнала для выбранной ревизии.

- Нижняя панель отображает список всех файлов и папок, которые были изменены в этой же ревизии.

Но возможности диалога намного шире: предусмотрены команды контекстного меню, при помощи которых можно получить дополнительную информацию об истории проекта.

4.9.1. Вызов диалога журнала ревизий

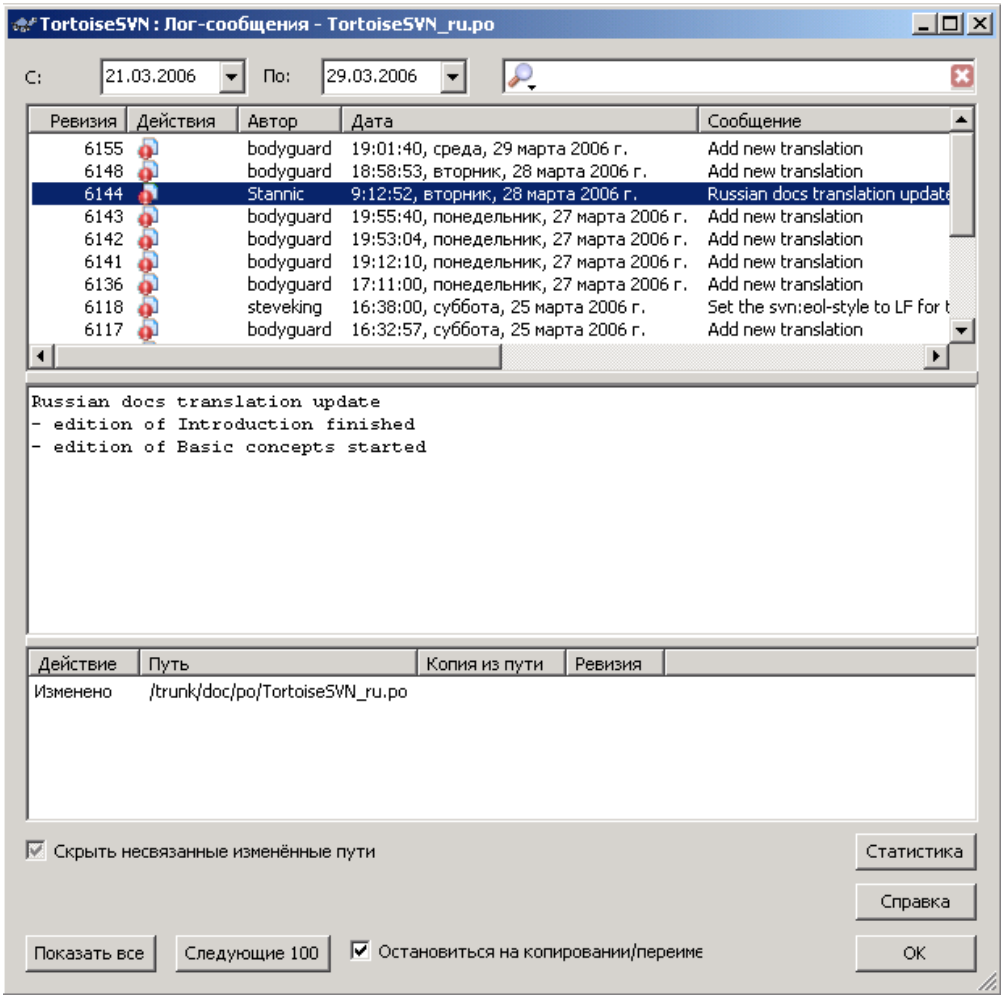


Рисунок 4.15. Диалоговое окно журнала ревизий

Есть несколько мест, из которых можно вызвать диалоговое окно журнала:

- Из контекстного меню TortoiseSVN
- Со страницы свойств
- Из окна выполнения после окончания обновления. В этом случае диалог журнала отображает только те ревизии, которые были изменены с момента последнего обновления

Если хранилище недоступно, вы увидите диалог Перейти в автономный режим?, который описывает Раздел 4.9.10, «Автономный режим».

4.9.2. Действия в журнале ревизий

В верхней панели есть столбец Действия, содержащий значки с обозначением того, что было сделано в этой ревизии. Всего есть четыре различных значка, каждый из которых показывается в своей позиции.

Если в ревизии были изменены файл или папка, в первой колонке отображается значок *изменён*.

Если в ревизии были добавлены файлы или папки, то во второй колонке отображается значок *добавлен*.

Если в ревизии были удалены файлы или папки, то в третьей колонке отображается значок *удалён*.

Если в ревизии файлы или папки были перемещены, то в четвёртой колонке отображается значок *перемещён*.

4.9.3. Получение дополнительной информации

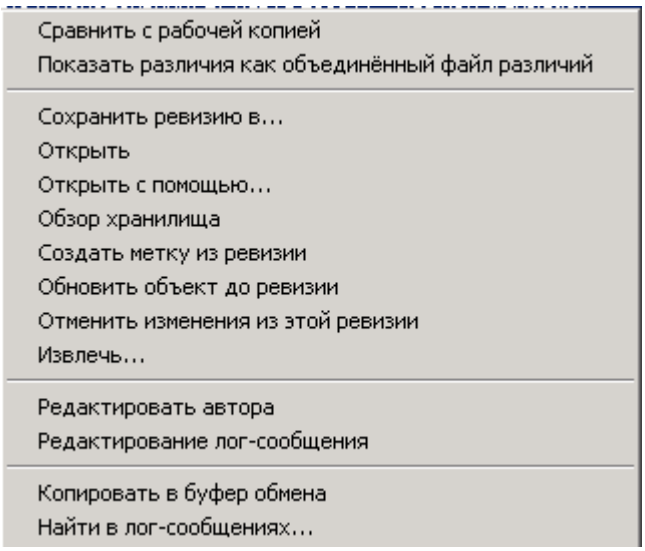


Рисунок 4.16. Контекстное меню верхней панели диалогового окна журнала ревизий

В верхней панели диалога журнала есть контекстное меню, позволяющее получить намного больше дополнительной информации. Некоторые из этих пунктов меню появляются только когда журнал показывается для файла, а некоторые - только когда журнал показывается для папки.

Сравнить с рабочей копией
: Сравнивает выбранную ревизию с вашей рабочей копией. По умолчанию в качестве программы сравнения используется TortoiseMerge, поставляемая с TortoiseSVN. Если диалог журнала вызван для папки, он отобразит список изменённых файлов, и позволит просмотреть изменения, сделанные в каждом отдельном файле.

Сравнить с рабочей базой с просмотром авторства
: Получает авторство для выбранной ревизии и для файла из рабочей базы, после чего сравнивает результаты об авторстве с применением визуального средства просмотра различий. Прочтите Раздел 4.23.2, «Авторство различий» для дополнительной информации (только для файлов).

Показать изменения как объединённые различия
: Позволяет просмотреть сделанные в выбранной ревизии изменения в виде объединённого файла различий (Unified-Diff) (формат заплаток GNU). Отображаются только различия (с несколькими строками контекста). Этот вид сложнее для изучения, чем визуальное сравнение файлов, но позволяет показать все изменения в файле в компактном формате.

Сравнить с предыдущей ревизией
: Сравнивает выбранную ревизию с предыдущей. Это работает подобно сравнению с вашей рабочей копией. Для папок этот пункт покажет сначала диалог изменённых файлов, позволяющий выбрать файлы для сравнения.

Сравнить с предыдущей ревизией с просмотром авторства
: Показывает диалог изменённых файлов, позволяющий выбирать файлы. Получает авторство для выбранной и для предыдущей ревизий, после чего сравнивает результаты с применением визуального средства просмотра различий (только для папок).

Сохранить ревизию в...
: Сохраняет выбранную ревизию в файл, чтобы у вас была и более старая ревизия этого файла (только для файлов).

Открыть / Открыть с помощью...
: Открывает выбранный файл либо в программе просмотра по умолчанию для этого типа файлов, либо в другой выбранной вами программе (только для файлов).

Авторство...
: Получает информацию об авторстве для файла вплоть до выбранной ревизии (только для файлов).

Обзор хранилища
: Открывает обозреватель хранилища для исследования выделенных файлов или папок в хранилище, какими они были в выбранной ревизии.

Создать ответвление/метку из ревизии
: Создаёт ответвление или метку из выбранной ревизии. Это полезно, например, если вы забыли создать метку и уже зафиксировали некоторые изменения, не предназначенные для этого выпуска.

Обновить элемент до ревизии
: Обновляет вашу рабочую копию до выбранной ревизии. Это полезно, если вам нужна рабочая копия, соответствующая какому-либо времени в прошлом, или если произошли последующие фиксации в хранилище и вы желаете обновлять вашу рабочую копию по одному шагу за раз. Лучше обновить всю папку целиком в вашей рабочей копии, а не только один файл, иначе ваша рабочая копия может стать несогласованной.

  Если вы желаете навсегда убрать более раннее изменение, используйте вместо этого Вернуть к этой ревизии.

Вернуть к этой ревизии
: Возвращает к более ранней ревизии. Если вы уже выполнили несколько изменений, и затем решили, что вам действительно необходимо вернуться назад, к состоянию как в ревизии N, то это та команда, которая вам нужна. Изменения отменяются в вашей рабочей копии, так что эта операция *не влияет* на хранилище, пока вы не зафиксировали эти изменения. Обратите внимание, что она отменяет *все* изменения, сделанные после выбранной ревизии, заменяя файл/папку более ранней версией.

  Если ваша рабочая копия находится в неизменённом состоянии, после выполнения данного действия она будет показана как изменённая. Если у вас уже есть локальные изменения, эта команда произведёт *отменяющее* слияние с вашей рабочей копией.

  За сценой происходит следующее: Subversion выполняет обратное слияние всех изменений, сделанных после выбранной ревизии, отменяя результат этих предыдущих фиксаций.

  Если после выполнения этого действия вы решите *отменить отмену* и вернуть вашу рабочую копию обратно, в прежнее неизменённое состояние, вы должны использовать TortoiseSVN → Убрать изменения из Проводника Windows, которое отбросит локальные изменения, произведённые этим обратным слиянием.

  Если вы просто желаете посмотреть, как выглядели файл или папка в этой ранней ревизии, воспользуйтесь вместо этого Обновить до ревизии или Сохранить ревизию как....

Отменить изменения из этой ревизии
: Отменяет изменения, которые были сделаны в выбранной ревизии. Изменения отменяются в вашей рабочей копии, так что эта операция *не влияет* на хранилище вообще! Обратите

внимание: отменяются изменения, сделанные только в этой ревизии. Эта операция не заменяет целиком файл в вашей рабочей копии файлом более ранней ревизии. Это очень полезно для отмены более ранних изменений, после которых были произведены другие, не связанные с ними, изменения.

Если ваша рабочая копия находится в неизменённом состоянии, после выполнения данного действия она будет показана как изменённая. Если у вас уже есть локальные изменения, эта команда произведёт *отменяющее* слияние с вашей рабочей копией.

За сценой происходит следующее: Subversion выполняет обратное слияние одной этой ревизии, отменяя результат предыдущей фиксации.

Вы можете *отменить отмену* как описано выше в Вернуть к этой ревизии.

Слить ревизию с...
: Производит слияние выделенных ревизий с другой рабочей копией. При помощи диалога выбора папки можно выбрать рабочую копию для проведения слияния, но после этого не предоставляется ни запроса подтверждения, ни возможности выполнить пробное слияние. Хорошей практикой является производить слияние с неизменённой рабочей копией, чтобы вы смогли отменить изменения, если слияние не сработает! Это полезная возможность, если вы желаете слить выбранные ревизии из одного ответвления с другим.

Извлечь...
: Выполняет свежее извлечение выбранной папки из заданной ревизии. При этом открывается окно, в котором необходимо подтвердить URL-адрес и ревизию, а также выбрать место для извлечения.

Экспорт...
: Экспортирует выделенный файл/папку из выбранной ревизии. При этом открывается диалог для подтверждения URL-адреса и ревизии, а также выбора места для размещения экспорта.

Изменить автора / сообщение журнала
: Отредактировать сообщение журнала или автора, относящихся к одной из предыдущих фиксаций. Прочтите Раздел 4.9.7, «Изменение сообщения журнала и автора», чтобы узнать, как это работает.

Показать свойства ревизии
: Позволяет просмотреть и отредактировать любое свойство ревизии, а не только сообщение журнала или автора. Подробнее - Раздел 4.9.7, «Изменение сообщения журнала и автора».

Копировать в буфер обмена
: Копирует в буфер обмена подробности записей журнала для выбранных ревизий. При этом будут скопированы номер ревизии, автор, дата, сообщение журнала и список изменённых объектов для каждой ревизии.

Найти в сообщениях журнала...
: Выполняет поиск в сообщениях журнала указанного вами текста. Поиск будет производиться в введённых сообщениях, а также в сводке выполненных действий, создаваемой Subversion (отображаемой в нижней панели). Поиск не зависит от регистра.

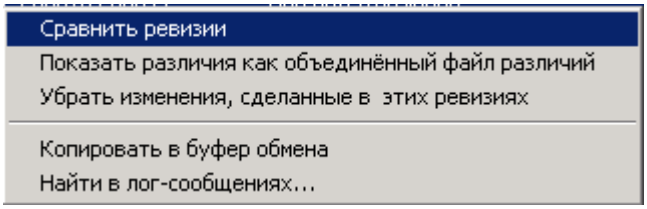


Рисунок 4.17. Контекстное меню верхней панели для двух выбранных ревизий

Если выделить сразу две ревизии (применяя для этого, как обычно, клавишу **Ctrl**), контекстное меню изменится и в нём будет меньше возможностей:

Сравнить ревизии
: Сравнивает две выбранные ревизии с использованием визуального средства просмотра различий. По умолчанию, в качестве такого средства используется TortoiseMerge, поставляемая с TortoiseSVN.

  Если вы выберите эту опцию для папок, появится диалог со списком изменённых файлов, в котором можно выбрать дальнейшие возможные действия по сравнению. Больше о диалоге сравнения ревизий можно прочитать в Раздел 4.10.3, «Сравнение папок».

Авторство ревизий
: Получает авторство для двух ревизий и сравнивает полученные результаты с применением визуального средства просмотра различий. Для получения более подробной информации прочтите Раздел 4.23.2, «Авторство различий».

Показать различия как объединённые различия
: Просмотр различий между двумя выбранными ревизиями в виде объединённого файла различий. Это работает и для файлов, и для папок.

Копировать в буфер обмена
: Копирует сообщения журнала в буфер обмена, как описано выше.

Найти в сообщениях журнала...
: Искать в сообщениях журнала, как описано выше.

Если выбрать две или больше ревизий (используя, как обычно, **Ctrl** или **Shift**), контекстное меню будет включать пункт для отмены всех изменений, которые сделаны в выбранных ревизиях. Это простейший путь откатить изменения из группы ревизий за один подход.

Также можно произвести слияние выбранных ревизий с другой рабочей копией, как было описано выше.

Если у всех выбранных ревизий один автор, вы можете изменить автора всех этих ревизий сразу.

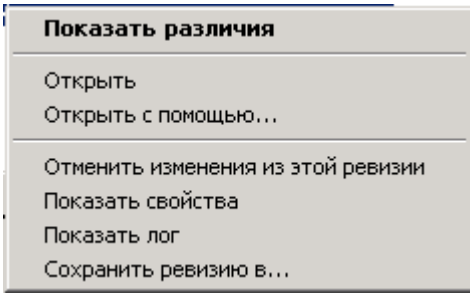


Рисунок 4.18. Контекстное меню нижней панели окна журнала

Нижняя панель окна журнала также имеет контекстное меню, которое предоставляет следующие возможности:

Показать изменения
: Показывает изменения, сделанные в выбранной ревизии для выделенного файла. Это контекстное меню доступно только для файлов, отображаемых как *изменённый*.

Авторство изменений
: Получает авторство для выбранной и предыдущей ревизий выделенного файла и сравнивает полученные результаты с применением визуального средства просмотра различий. Прочтите Раздел 4.23.2, «Авторство различий» для дополнительной информации.

Показать как объединённые различия
: Показывает изменения в формате объединённых различий. Это контекстное меню доступно только для файлов, отображаемых как *изменённый*.

Открыть / Открыть с помощью...
: Открывает выбранный файл либо в программе просмотра по умолчанию для этого типа файлов, либо в другой выбранной вами программе.

Авторство...
: Открывает диалог авторства, позволяет посмотреть авторство в выбранной ревизии.

Отменить изменения из этой ревизии
: Отменяет изменения, сделанные в выделенном файле в этой ревизии.

Показать свойства
: Позволяет посмотреть свойства Subversion для выбранных элементов.

Журнал
: Показывает журнал ревизий для одного выбранного файла.

Получить информацию о слияниях
: Показывает журнал ревизий для единственного выбранного файла, включая слитые изменения. Больше информации об этом содержит Раздел 4.9.6, «Возможности по отслеживанию слияний».

Сохранить ревизию в...
: Сохраняет выбранную ревизию в файл, чтобы вы могли получить и более старую ревизию этого файла.

Подсказка

Вы можете заметить, что иногда мы упоминаем изменения, а иногда - различия. В чём разница?

Номера ревизий в Subversion используются для обозначения двух различных вещей: обычно ревизия представляет состояние хранилища в определённый момент времени, но также может использоваться для обозначения набора изменений при помощи которого была создана эта ревизия, например «Сделано в ревизии 1234» означает, что изменения, зафиксированные в r1234, реализуют возможность X. Для того, чтобы было понятнее, какое из значений используется, мы применяем два различных термина.

Если выбрать две ревизии, N и M, в контекстном меню будет предложено показать *различия* между двумя этими ревизиями. В терминах Subversion это `diff -r M:N`.

Если выбрать единственную ревизию N, в контекстном меню будет предложено показать *изменения*, сделанные в этой ревизии. В терминах Subversion это `diff -r N-1:N` или `diff -c N`.

В нижней панели показываются файлы, изменённые во всех выбранных ревизиях, поэтому в контекстном меню всегда предлагается показать *изменения*.

4.9.4. Получение большего количества сообщений журнала

По нескольким причинам в окне журнала не всегда отображаются все когда-либо сделанные изменения:

- В большом хранилище могут быть сотни или даже тысячи изменений, и получение их всех может занять много времени. Обычно вас интересуют самые недавние изменения. По

умолчанию, число извлекаемых сообщений журнала ограничено 100, но вы можете изменить это значение во вкладке TortoiseSVN → Настройки (Раздел 4.30.1.2, «Настройки диалогов TortoiseSVN - 1»),

- Если отмечен флажок Останавливаться на копировании/переименовании, отображение журнала будет остановлено в точке, где выбранные файл или папка были скопированы из другого места в хранилище. Это может пригодиться при просмотре ответвлений (или меток), поскольку остановка происходит на корне ответвления и это позволяет быстро определить изменения, сделанные только в этом ответвлении.

  Обычно пользователи оставляют этот флажок неотмеченным. TortoiseSVN запоминает состояние этого флажка, и в дальнейшем будет действовать в соответствии с вашим выбором.

  Когда окно журнала вызывается из диалога 'Слияние', по умолчанию флажок установлен всегда. Так сделано потому, что при слиянии наиболее часто интересуют изменения в ответвлениях, и в этом случае движение дальше корня ответвления не имеет смысла.

  Обратите внимание: сейчас Subversion реализует переименование как пару копирование/удаление, так что переименование файла или папки также останавливает отображение журнала в случае, если установлен этот флажок.

Если вы желаете просмотреть остальные сообщения журнала, нажмите на Следующие 100 для извлечения следующей сотни сообщений. Вы можете повторять это столько раз, сколько вам нужно.

Рядом с этой кнопкой расположена многофункциональная кнопка, которая запоминает последнюю функцию, для которой вы её использовали. Щёлкните на стрелке для просмотра доступных вариантов.

Если вы желаете просмотреть определённый диапазон ревизий, используйте Показать ряд.... Появится диалог, предлагающий ввести начальную и конечную ревизии.

Если вы желаете просмотреть *все* сообщения, начиная с ведущей (HEAD) ревизии и заканчивая ревизией 1, используйте Показать все.

4.9.5. Текущая ревизия рабочей копии

Поскольку окно сообщений журнала показывает журнал для ведущей ревизии, а не для текущей ревизии рабочей копии, часто случается, что показываются сообщения журнала для содержимого, которое ещё не было обновлено в вашей рабочей копии. Чтобы сделать это более наглядным, сообщение фиксации, соответствующее ревизии вашей рабочей копии, отображается полужирным шрифтом.

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

4.9.6. Возможности по отслеживанию слияний

В Subversion версий 1.5 и больше слияния регистрируются при помощи свойств. Это позволяет нам получить более подробную историю слитых изменений. Например, если вы разработали новую возможность в ответвлении и потом произвели слияние этого ответвления обратно в ствол, разработка этой возможности будет показана в журнале ствола как единственная фиксация для слияния, даже если в ответвлении было произведено 1000 фиксаций во время разработки.

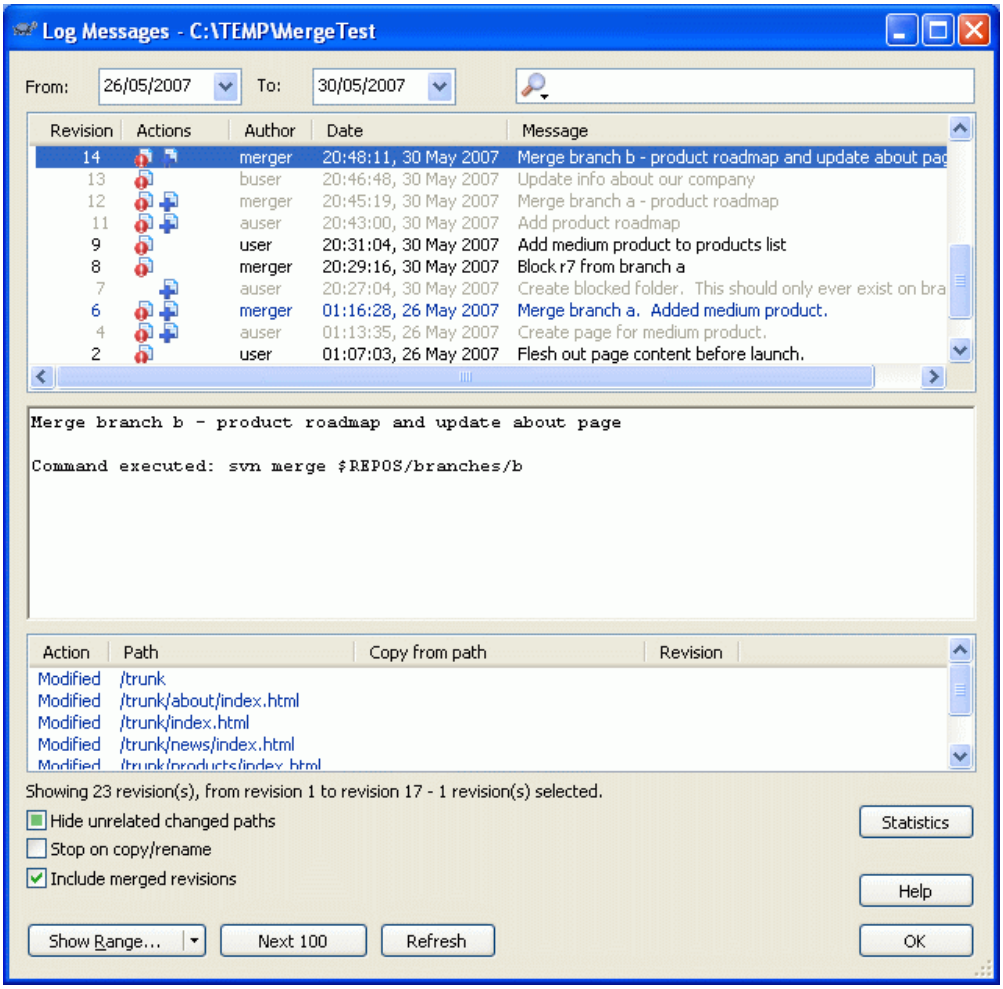


Рисунок 4.19. Диалог журнала, показывающий ревизии с отслеженными слияниями

Если вы желаете посмотреть подробно, какие ревизии были слиты как часть этой фиксации, используйте флажок Включая слитые ревизии. При этом будут заново получены сообщения журнала, но при этом также будут добавлены в нужные места сообщения журнала из ревизий, которые были слиты. Слитые ревизии показываются серым, потому что они представляют изменения, сделанные в другой части дерева.

Конечно же, слияние никогда не бывает простым! Во время разработки возможности в ответвлении, вероятно, иногда производились слияния обратно из ствола для сохранения согласованности ответвления с основной линией разработки. Поэтому история слияний ответвления также включает ещё один слой истории слияний. Эти различные слои показываются в диалоге журнала при помощи уровня отступов.

4.9.7. Изменение сообщения журнала и автора

Свойства ревизии полностью отличаются от свойств Subversion каждого элемента. Свойства ревизии - элементы описания, которые связаны с одним конкретным номером ревизии в хранилище, такие как сообщение журнала, дата фиксации и имя зафиксировавшего (автора).

Иногда вы можете пожелать изменить введённое вами сообщение журнала, может быть из-за обнаруженной орфографической ошибки, или вы желаете улучшить сообщение, или же изменить его по другим причинам. Или, возможно, вы захотите изменить автора фиксации, т.к. вы забыли настроить идентификацию, или ...

Subversion позволяет изменить свойства ревизии в любое удобное для вас время. Но, поскольку такие изменения не могут быть отменены (эти изменения не версируются), эта возможность по умолчанию отключена. Для того, чтобы её включить, вы должны установить ловушку pre-revprop-change (перед-изменением-свойства_ревизии). Пожалуйста, за подробным описанием того, как это сделать, обратитесь к главе *Hook Scripts (Скрипты ловушек)* [http://svnbook.red-bean.com/en/1.5/svn.reposadmin.create.html#svn.reposadmin.create.hooks] в Книге о Subversion. Прочтите Раздел 3.3, «Скрипты ловушек, выполняемые на стороне сервера», в котором содержатся несколько дополнительных заметок о реализации ловушек на компьютере с Windows.

После того, как вы настроили на сервере необходимые ловушки, вы можете изменять автора и сообщение (или любое другое свойство) любой ревизии, используя контекстное меню из верхней панели окна журнала. Вы также можете отредактировать сообщение журнала, воспользовавшись контекстным меню в средней панели.

Внимание

Из-за того, что свойства ревизий в Subversion не версируются, изменение таких свойств (например, свойства `svn:log` - сообщение журнала при фиксации) будет перезаписывать предыдущее значение этого свойства *навсегда*.

4.9.8. Фильтрация сообщений журнала

Чтобы просмотреть только интересующие вас сообщения, без необходимости в прокрутке списка из сотен записей, можно настроить фильтры в верхней части диалогового окна журнала. Поля ввода начальной и конечной дат позволяют вам ограничить выдачу заданным диапазоном дат. Поле поиска позволяет отобразить только те сообщения, которые содержат определённую фразу.

Щёлкните на значке поиска, чтобы выбрать область, в которой вы собираетесь производить поиск информации, а также чтобы включить режим *регулярных выражений*. Обычно бывает нужен простой поиск текста, но если вам необходимы более гибкие условия для поиска, вы можете воспользоваться регулярными выражениями. При наведении мыши на поле поиска появится подсказка с основами использования регулярных выражений. Вы также можете воспользоваться документацией и учебным курсом, доступными в Сети по адресу *http://www.regular-expressions.info/*. Строка поиска сопоставляется фильтром с записями журнала, и затем показываются только те записи, которые *соответствуют* строке поиска.

Для того, чтобы фильтр показывал все записи, которые *не соответствуют* строке поиска, начните строку с восклицательного знака ('!'). Например, строка поиска `!пользователь` покажет только те записи, которые фиксировал не `пользователь`.

Обратите внимание: эти фильтры действуют только на уже извлеченные сообщения. Они не управляют загрузкой сообщений из хранилища.

Также можно отфильтровать пути в нижней панели, используя флажок Скрыть несвязанные изменённые пути. Связанные пути - это те, которые включают путь, используемый для показа журнала. Если вы запрашиваете журнал для папки, то это означает всё в этой папке или ниже её. Для файла - это только этот файл. Флажок может находится в трёх состояниях: можно отображать все пути, сделать несвязанные пути серыми или скрыть несвязанные пути полностью.

Иногда установленный порядок работы требует, чтобы сообщения журнала соответствовали определённому формату, и это иногда означает, что текст, описывающий изменения, не виден в краткой сводке, показываемой в верхней панели. Свойство `tsvn:logsummary` может быть использовано для извлечения части сообщения журнала, которая будет показана в верхней панели. Прочтите Раздел 4.17.2, «Свойства проекта в TortoiseSVN», чтобы узнать, как применить это свойство.

Никакого форматирования журнала в обозревателе хранилища

Поскольку форматирование зависит от доступа к свойствам Subversion, вы сможете увидеть результаты только при использовании извлечённой рабочей копии. Получение свойств удалённо - медленная операция, поэтому вы не увидите работу этой воможности в обозревателе хранилища.

4.9.9. Статистическая информация

Кнопка Статистика вызывает диалоговое окно, отображающее некоторую интересную информацию о ревизиях, показываемых в настоящий момент окне журнала. В этом окне показано, сколько авторов работало, сколько фиксаций они выполнили, продвижение за неделю и много чего другого. Теперь вы можете с одного взгляда определить, кто тяжело работал, а кто прохлаждался ;-)

4.9.9.1. Страница статистики

Эта страница предоставляет разнообразные численные значения, которые вам могут понадобиться, в частности, период охвата и количество затронутых ревизий, а также некоторые минимальные/максимальные/средние значения.

4.9.9.2. Страница 'Фиксации по автору'

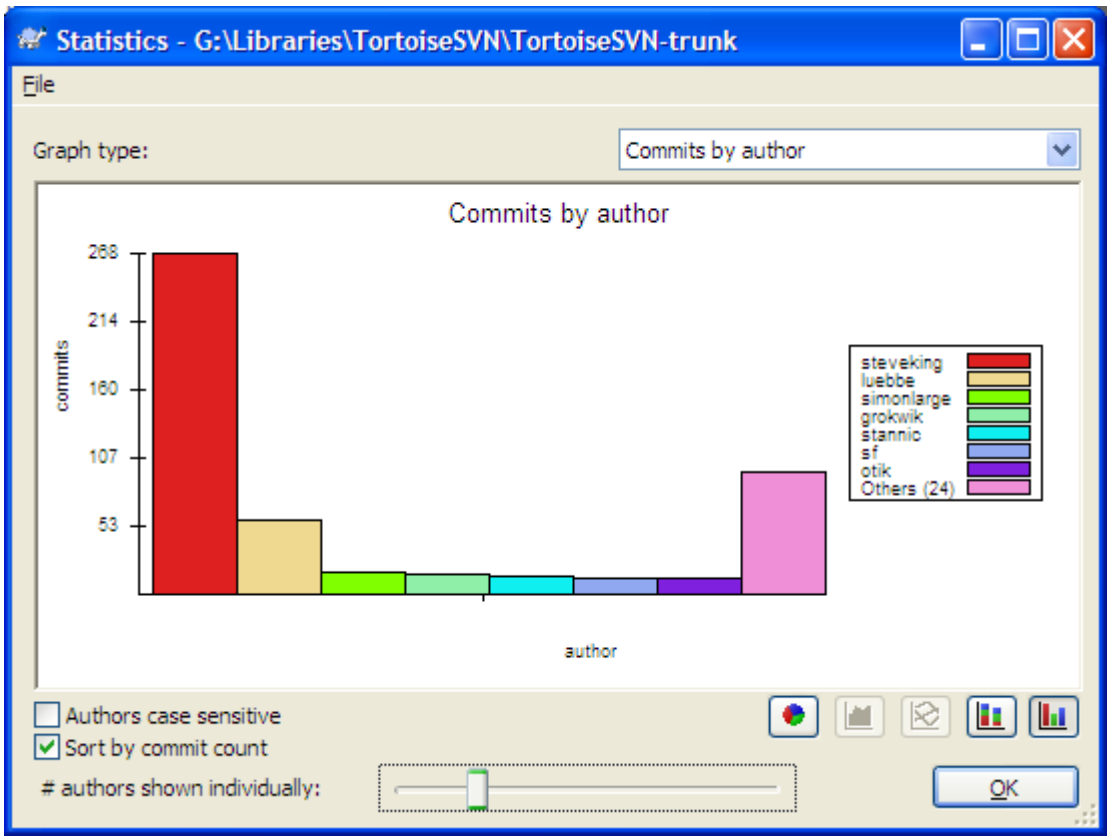


Рисунок 4.20. Гистограмма Фиксации-по-автору

Этот график показывает, кто из авторов и насколько активно работал над проектом в виде простой гистограммы, гистограммы с накоплением ("стопкой") или секторной диаграммы.

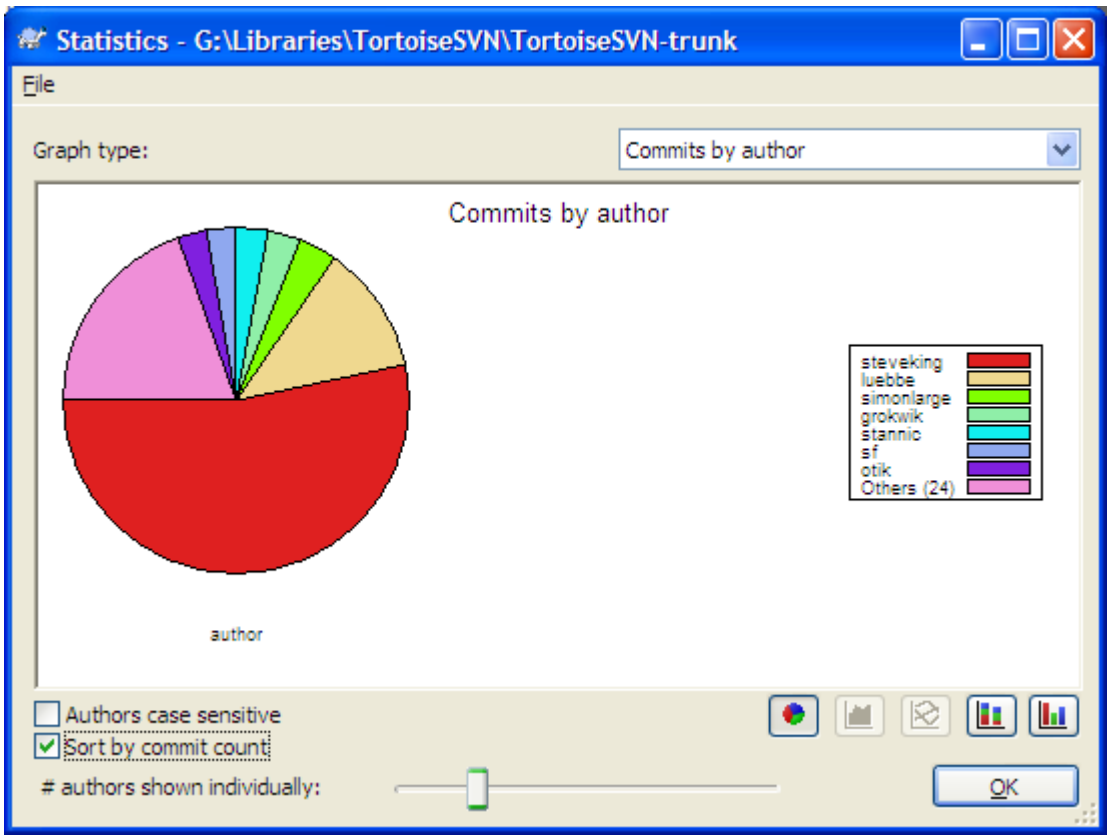


Рисунок 4.21. Секторная диаграмма Фиксации-по-автору

Когда присутствует небольшое число основных авторов и много фиксирующих от случая к случаю, значительное количество маленьких сегментов может сделать график трудночитаемым. Ползунок снизу служит для установки порогового значения (в процентах от общего числа фиксаций), ниже которого вся деятельность объединяется в категорию *Остальные*.

4.9.9.3. Страница 'Фиксации по датам'

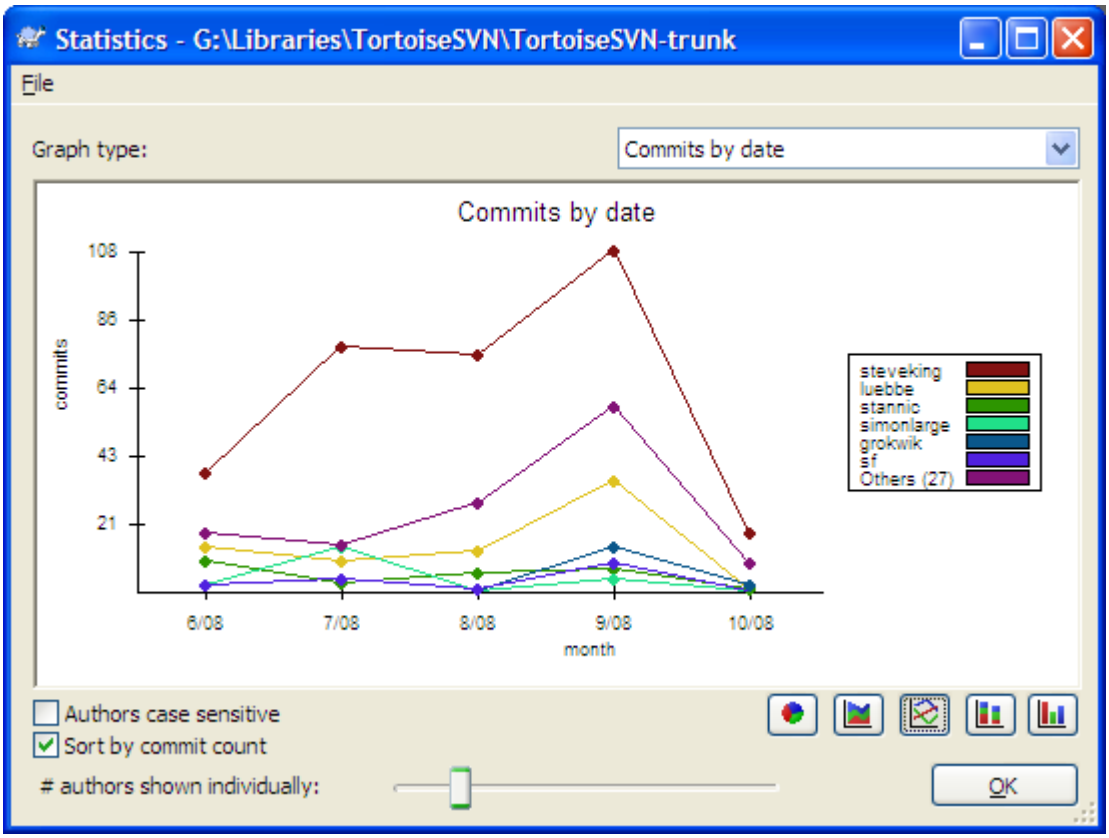


Рисунок 4.22. График Фиксации-по-датам

Эта страница предоставляет график деятельности по проекту в разрезе количества фиксаций *и* авторов. Это даёт некоторое представление о том, когда велась работа над проектом, и кто в какое время работал.

При отображении нескольких авторов на графике будет много линий. При этом можно выбрать один из двух его видов: *обычный*, где количество фиксаций каждого автора откладывается от линии оси абсцисс, и *стопкой*, где количество фиксаций каждого автора откладывается от предыдущей линии. Последняя возможность позволяет избежать пересечения линий, что, возможно, облегчает прочтение графика, но затрудняет определение вклада каждого конкретного автора.

По умолчанию, анализ производится с учётом регистра, так что пользователи `PeterEgan` и `PeteRegan` рассматриваются как разные авторы. Однако в большинстве случаев регистр в именах пользователей не важен, и иногда они вводятся по-разному, поэтому бывает желательно, чтобы пользователи `DavidMorgan` и `davidmorgan` рассматривались как один человек. Используйте флажок Авторы без учёта регистра для указания способа обращения с именами пользователей.

Обратите внимание, статистика охватывает тот же период, что и окно журнала. И если оно отображает только одну ревизию, то и статистика сообщит вам не очень много.

4.9.10. Автономный режим

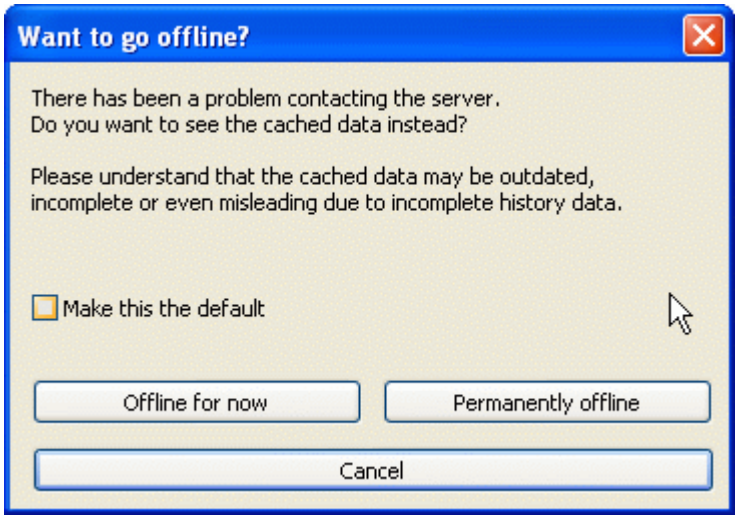


Рисунок 4.23. Диалог перехода в автономный режим

Если сервер недоступен, и включено кэширование журнала, то вы можете использовать диалог журнала и граф ревизий в автономном режиме. Этот режим использует закэшированные данные, что позволяет продолжить работу, несмотря на то, что информация может быть устаревшей или неполной.

Здесь у вас есть три возможности:

Автономно в этот раз
: Завершить текущую операцию в автономном режиме, но снова обратиться к хранилищу в следующий раз, когда потребуются данные журнала.

Всегда работать автономно
: Оставаться в автономном режиме, пока проверка хранилища не будет запрошена специально. См. Раздел 4.9.11, «Обновление вида».

Отмена
: Если вы не желаете продолжать операцию с возможно устаревшими данными, просто отмените.

Флажок Применять по умолчанию позволяет сделать так, чтобы это окно больше не появлялось и всегда применять опцию, которую вы затем выберите. Вы сможете изменить (или убрать) выбор по умолчанию и после этого, используя TortoiseSVN → Настройки.

4.9.11. Обновление вида

Если вы желаете вновь запросить сервер на предмет новых сообщений журнала, вы можете просто обновить вид при помощи **F5**. Если используется кэширование журнала (по умолчанию включено), то хранилище будет проверено на наличие более новых сообщений и будут загружены только они. Если кэш журнала работает в автономном режиме, будет произведена попытка переключиться обратно в оперативный режим.

Если вы используете кэширование журнала, и вы думаете, что содержимое сообщения или его автор были изменены, вы можете воспользоваться **Shift-F5** или **Ctrl-F5** для повторного получения отображаемых сообщений с сервера и обновления кэша журнала. Обратите внимание: это относится только к сообщениям, отображаемым в данный момент, и не делает недействительным весь кэш для этого хранилища.

4.10. Просмотр различий

Одно из самых общих требований при разработке проекта - видеть, что было изменено. Вам может понадобиться посмотреть различия между двумя ревизиями одного и того же файла, или различия между двумя различными файлами. Для просмотра различий в текстовых файлах TortoiseSVN предоставляет встроенный инструмент, называемый TortoiseMerge. В TortoiseSVN также есть утилита для просмотра различий в графических файлах под названием TortoiseIDiff. Конечно, при желании вы можете использовать вашу любимую программу для просмотра различий.

4.10.1. Различия в файлах

Локальные изменения
: Если вам надо посмотреть, какие изменения *вы* сделали в вашей рабочей копии, просто вызовите контекстное меню Проводника и выберите TortoiseSVN → Различия.

Различия с другим ответвлением/меткой
: Если вы желаете посмотреть, что изменилось в основном стволе (если вы работаете в ответвлении) или в каком-то ответвлении (если вы работаете в основном стволе), вы можете воспользоваться контекстным меню Проводника. Просто удерживайте нажатой клавишу **Shift** при щелчке на файле правой кнопкой мыши. Далее выберите TortoiseSVN → Различия с файлом по URL и в последующем диалоге укажите URL в хранилище, с которым вы желаете сравнить ваш локальный файл.

  Вы также можете выбрать в обозревателе хранилища два дерева для сравнения, возможно, две метки, или ответвление/метку и ствол. Их можно сравнить, используя Сравнить ревизии из контекстного меню. Больше прочитать об этом можно в Раздел 4.10.3, «Сравнение папок».

Различия с предыдущей ревизией
: Если вы желаете посмотреть различия между определённой ревизией и вашей рабочей копией, выберите нужную ревизию в диалоге журнала ревизий, затем выберите Сравнить с рабочей копией из контекстного меню.

  Если вы желаете посмотреть различия между последней зафиксированной ревизией и вашей рабочей копией, при условии, что рабочая копия не была изменена, просто выполните правый щелчок на файле и выберите TortoiseSVN → Сравнить с предыдущей версией. Это запустит процесс получения различий между ревизией перед последней-датой-фиксации (зарегистрированной в вашей рабочей копии) и рабочей базой. Будут показаны последние произведённые в этом файле изменения, при помощи которых файл был приведён в своё текущее состояние, наблюдаемое в вашей рабочей копии. Изменения, более поздние, чем ваша рабочая копия, не показываются.

Различия между двумя предыдущими ревизиями
: Если вы желаете посмотреть различия между двумя ранее зафиксированными ревизиями, выделите в диалоге журнала ревизий (применяя для этого, как обычно, клавишу **Ctrl**) две ревизии, которые вы хотите сравнить. Затем выберите Сравнить ревизии из контекстного меню.

  Если это сделать из журнала ревизий для папки, то появится диалог сравнения ревизий, отображающий список изменённых файлов из этой папки. Больше можно прочитать в Раздел 4.10.3, «Сравнение папок».

Все изменения, сделанные в фиксации
: Если вы желаете посмотреть все изменения, произведённые во всех файлах в определённой ревизии, собранные в одном месте, вы можете применить выдачу в виде объединённых различий (Unified-Diff, формат заплаток GNU). Будут показаны только различия с несколькими строками контекста. Этот формат сложнее для чтения, чем визуальное сравнение

файлов, но он показывает сразу все изменения. В диалоге журнала ревизий выберите интересующую вас ревизию, затем выберите Показать различия как объединённые различия из контекстного меню.

Различия между файлами
: Если вы желаете посмотреть различия между двумя разными файлами, вы можете сделать это прямо в Проводнике, выделив оба файла (как обычно, с использованием клавиши **Ctrl**) и выбрав TortoiseSVN → Различия из контекстного меню Проводника.

Различия между файлом/папкой в рабочей копии и файлом/папкой по URL
: Если вы желаете посмотреть различия между файлом из вашей рабочей копии и файлом в каком-нибудь хранилище Subversion, то это можно сделать прямо в Проводнике, выделив этот файл и вызвав контекстное меню, удерживая клавишу **Shift**, после чего выбрав TortoiseSVN → Различия с файлом по URL. Это же можно сделать и для папки в рабочей копии. TortoiseMerge показывает эти различия также, как показывает файл заплаток - в виде списка изменённых файлов, которые можно просматривать по одному за раз.

Различия с информацией об авторстве
: Если вы желаете посмотреть не только различия, но и автора, ревизию и дату сделанных изменений, вы можете объединить выдачу по различиям и авторству из диалога журнала ревизий. Прочтите Раздел 4.23.2, «Авторство различий» для дополнительной информации.

Различия между папками
: Встроенные утилиты, поставляемые с TortoiseSVN, не поддерживают показ различий между иерархиями папок. Но если у вас есть другой инструмент, обладающий такой возможностью, вы можете использовать его. В Раздел 4.10.5, «Внешние инструменты просмотра различий/слияния» мы расскажем о некоторых инструментах, которые нам довелось попробовать.

Если у вас в настройках указан сторонний инструмент сравнения, вы можете использовать клавишу **Shift** при выборе команды 'Различия' для его применения. Прочтите Раздел 4.30.5, «Настройки внешних программ», чтобы узнать, как настраивать другие инструменты сравнения.

4.10.2. Параметры сравнения завершений строк и непечатаемых знаков

За время жизни проекта случается, что вы изменяете завершения строк с `CRLF` на `LF`, или изменяете отступ какой-нибудь части. К сожалению, это приводит к тому, что большое количество строк помечаются как изменённые, даже если не было изменений смысла кода. Следующие параметры помогут справиться с такими изменениями, когда дело доходит до сравнения и применения различий. Эти настройки присутствуют в диалогах Слияния и Авторства, а также в настройках TortoiseMerge.

Игнорировать завершения строк исключает изменения, возникающие только из-за разницы типов завершений строк.

Сравнивать непечатаемые знаки включает все изменения отступов и пробельных символов внутри строк в виде добавленных/удалённых строк.

Игнорировать изменения непечатаемых знаков исключает изменения, возникающие только из-за разницы в количестве или типе пробельных символов, таких как изменение отступов или замена табуляций на пробелы. Добавление пробельного символа там, где их раньше не было или полное их удаление в каком-либо месте всё равно отображается как изменение.

Игнорировать все непечатаемые знаки исключает все изменения только пробельных символов.

Конечно, все строки с изменившимся содержимым всегда включаются в различия.

4.10.3. Сравнение папок

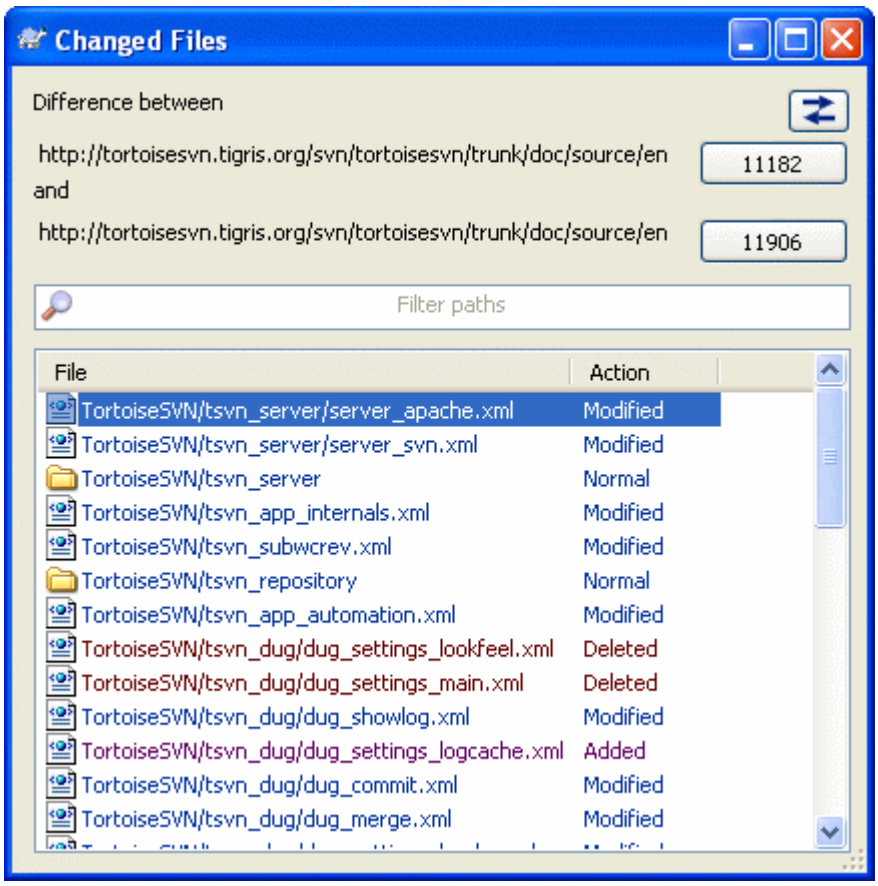


Рисунок 4.24. Диалог сравнения ревизий

Когда вы выбираете два дерева в обозревателе хранилища, или когда вы выбираете две ревизии папки в диалоге журнала, у вас есть возможность Контекстное меню → Сравнить ревизии.

Этот диалог показывает список всех изменённых файлов и позволяет производить сравнение или просматривать авторство отдельно для каждого файла, используя контекстное меню.

Вы можете экспортировать *дерево изменений*, полезное, если вам нужно отправить кому-нибудь структуру вашего проекта в виде дерева, содержащего только изменённые файлы. Эта операция работает только для выбранных файлов, поэтому вам надо выбрать интересующие файлы - часто это означает их все - и после этого Контекстное меню → Экспортировать выбранное в.... У вас будет запрошено место, куда будет сохранено дерево изменений.

Вы также можете экспортировать *список* изменённых файлов при помощи Контекстное меню → Сохранить список выбранных файлов....

Если вы желаете экспортировать список файлов *вместе* с выполненными действиями (изменено, добавлено, удалено), то это можно сделать при помощи пункта Контекстное меню → Копировать выбранное в буфер обмена.

Кнопка сверху позволяет изменить направление сравнения. Можно посмотреть изменения, необходимые, чтобы из А получить Б, или, если вам больше нравится, такие, чтобы из Б получить А.

Кнопки с номерами ревизий могут быть использованы для переключения на другой диапазон ревизий. При изменении диапазона список элементов, различающихся между ревизиями, будет обновлён автоматически.

Если список имён файлов очень длинный, можно применить поле поиска для уменьшения размеров списка: сделать так, чтобы в нём присутствовали только файлы, содержащие определённый текст в своём имени. Обратите внимание: используется простой поиск текста, поэтому если вам нужны в списке только файлы исходного кода на C, надо ввести `.c`, а не `*.c`.

4.10.4. Сравнение картинок при помощи TortoiseIDiff

Есть множество утилит для сравнения текстовых файлов, включая нашу собственную TortoiseMerge, но часто оказывалось, что нам также хотелось увидеть, что же изменилось в графических файлах. Именно поэтому мы создали TortoiseIDiff.

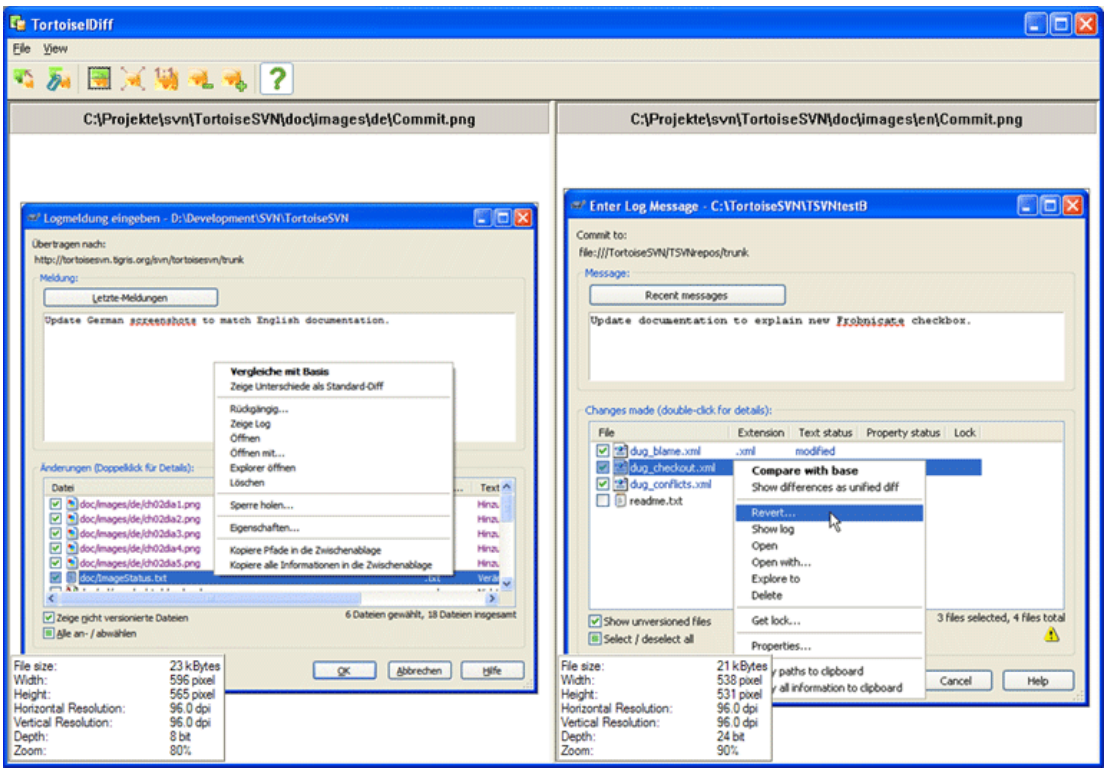


Рисунок 4.25. Программа просмотра различий в картинках

TortoiseSVN → Различия для файлов любого из широко распространённых графических форматов запускает TortoiseIDiff для показа различий в картинках. По умолчанию, картинки показываются бок о бок, но вы можете воспользоваться меню 'Вид' или инструментальной панелью для отображения картинок одна над другой, или, по желанию, вы можете наложить картинки одна на другую как при использовании проектора.

Естественно, вы можете также приблизить, удалить и передвинуть картинку. Передвинуть картинку можно также просто перетягивая её левой кнопкой мыши. Если включить флажок Связать картинки, то органы управления перемещением (полосы прокрутки, колёсико мыши) обеих картинок будут связаны.

В информационном окошке отображается дополнительная информация о графическом файле, такая как размер в пикселах, разрешение и глубина цвета. Если это окошко вам мешает, его можно скрыть, выбрав Вид → Информация о картинке. Эту же информацию можно получить во всплывающей подсказке при наведении мыши на на заголовок картинки.

Когда картинки наложены одна на другую, относительная интенсивность картинок (альфа-сопряжение) регулируется при помощи бегунка слева. Для задания нужной степени прозрачности можно щёлкнуть прямо в нужном месте бегунка, или же изменить значение при помощи

Кнопка над бегунком переключает между 0% и 100% прозрачностью, и при двойном щелчке на кнопке прозрачность будет переключаться автоматически каждую секунду, пока вы не щёлкните по кнопке ещё раз. Это может пригодиться при поиске нескольких мелких изменений.

Иногда бывает необходимо увидеть только то, чем изображения различаются, и не всегда наложение их друг на друга может помочь. Возможно, у вас есть изображения двух ревизий печатных плат и вы желаете посмотреть, какие дорожки изменились. При отключении режима альфа-сопряжения различия будут показаны при помощи операции *XOR* над значениями цветов пикселов. Неизменённые области будут чисто белыми, а изменения будут окрашены.

4.10.5. Внешние инструменты просмотра различий/слияния

Если предоставленные нами инструменты не делают того, что вам надо, попробуйте какую-нибудь из множества доступных альтернатив: программ с открытым исходным кодом или коммерческих программ. У каждого свои предпочтения, и этот список никоим образом не полон, но вот несколько программ, которые вы можете принять во внимание:

WinMerge
: *WinMerge* [http://winmerge.sourceforge.net/] - прекрасный инструмент просмотра различий, который также может работать с папками. Программа с открытым исходным кодом.

Perforce Merge
: Perforce - это коммерческая RCS, но вы можете загрузить бесплатный инструмент для просмотра различий/слияния. Дополнительную информацию можно получить на сайте *Perforce* [http://www.perforce.com/perforce/products/merge.html].

KDiff3
: KDiff3 - это бесплатный инструмент для просмотра различий, который также может работать с папками. Вы можете загрузить его *отсюда* [http://kdiff3.sf.net/].

ExamDiff
: ExamDiff Standard распространяется как freeware. Он может обрабатывать файлы, но не папки. ExamDiff Pro распространяется как shareware и добавляет несколько расширений, включая сравнение директорий и возможность редактирования. Обе разновидности, начиная с версии 3.2, могут работать с юникодом. Вы можете загрузить их с сайта *PrestoSoft* [http://www.prestosoft.com/].

Beyond Compare
: Подобно ExamDiff Pro, это прекрасный инструмент просмотра различий, умеющий работать с папками и с юникодом, и распространяемый по лицензии shareware. Загрузить его можно с сайта *Scooter Software* [http://www.scootersoftware.com/].

Araxis Merge
: Araxis Merge - это полезный коммерческий инструмент для показа различий/слияния как файлов, так и папок. Он выполняет трёхстороннее сравнение при слиянии и в нём есть ссылки синхронизации, применяемые, если вы изменили порядок функций. Его можно загрузить с сайта *Araxis* [http://www.araxis.com/merge/index.html].

SciTE
: Этот текстовый редактор включает подсветку синтаксиса для объединённых различий, делая их восприятие более лёгким. Его можно загрузить с сайта *Scintilla* [http://www.scintilla.org/SciTEDownload.html].

Notepad2
: Notepad2 разрабатывался как замена стандартного Блокнота Windows, и основывается на компоненте с открытым исходным кодом Scintilla. Он не только хорошо подходит для просмотра объединённых различий, он также намного лучше, нежели Блокнот Windows, справляется с большинством задач. Его можно бесплатно загрузить *отсюда* [http://www.flos-freeware.ch/notepad2.html].

В Раздел 4.30.5, «Настройки внешних программ» описано, как настроить TortoiseSVN для использования этих инструментов.

4.11. Добавление новых файлов и папок

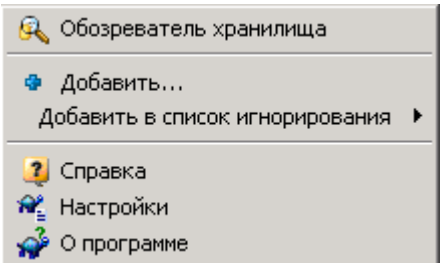


Рисунок 4.26. Контекстное меню Проводника для неверсированных файлов

Если вы создали новые файлы и/или папки во время процесса разработки, вам необходимо добавить их под управление версиями. Выберите файл(-ы) и/или папку, затем воспользуйтесь TortoiseSVN → Добавить....

После того, как вы добавите файлы/папки под управление версиями, на них появляется пометка `добавлен`, означающая, что вам необходимо сначала зафиксировать вашу рабочую копию, прежде чем эти файлы/папки станут доступны другим разработчикам. Добавление файла/папки *не затрагивает* хранилище!

Множественные добавления

Вы также можете использовать команду 'Добавить' на уже версированных папках. В этом случае в диалоге добавления будут показаны все неверсированные файлы из этой версированной папки. Это может помочь, если у вас много новых файлов и вам нужно добавить их все за один раз.

Для добавления файлов, находящихся вне вашей рабочей копии, вы можете воспользоваться обработчиком перетаскивания:

1. выберите файлы, которые вы хотите добавить
2. затем перетащите правой кнопкой мыши их на новое место внутри рабочей копии
3. отпустите правую кнопку мыши
4. выберите Контекстное меню → SVN Добавить файлы в эту рабочую копию. Файлы будут скопированы в рабочую копию и добавлены под управление версиями.

Вы также можете добавлять файлы из рабочей копии просто путём перетаскивания их левой клавишей мыши в диалог фиксации.

Если вы добавили файл или папку по ошибке, вы можете отменить это добавление до фиксации, воспользовавшись TortoiseSVN → Отменить добавление....

4.12. Копирование/перемещение/переименование файлов и папок

Часто случается, что у вас уже есть файлы, необходимые также в другом вашем проекте в том же хранилище, и вы просто хотите скопировать их туда. Конечно, вы можете просто скопировать файлы и добавить их как описано выше, но этот способ не перенесёт истории изменений. И

если вы в последующем исправите ошибку в исходных файлах, вы сможете слить исправление автоматически только если новая копия связана в Subversion с исходным файлом.

Простейший способ скопировать файлы и папки из рабочей копии - применить меню, появляющееся при переносе правой кнопкой мыши. когда вы переносите правой кнопкой файл или папку из одной рабочей копии в другую, или даже внутри этой же папки, при отпускании кнопки появляется контекстное меню.

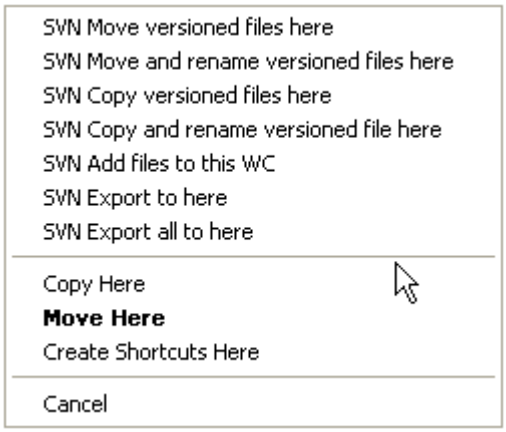


Рисунок 4.27. Меню при перетаскивании правой клавишей мыши для папки под управлением версиями

Теперь вы можете скопировать версированное содержимое в новое место, возможно, с переименованием.

Вы можете также копировать и перемещать версированные файлы в пределах рабочей копии, или между двумя рабочими копиями при помощи привычного метода вырезать-и-вставить. Воспользуйтесь стандартными операциями Windows Копировать или Вырезать для размещения одного или более версированных элементов в буфере обмена. Если в буфере обмена уже содержатся такие версированные элементы, то вы можете использовать операцию TortoiseSVN → Вставить (обратите внимание: это НЕ стандартная операция Windows Вставить) для копирования или перемещения этих элементов в новое место рабочей копии.

Вы можете копировать файлы и папки из вашей рабочей копии в другое место в хранилище используяTortoiseSVN → Ответвление/Метка. Чтобы узнать об этом больше, прочтите Раздел 4.19.1, «Создание ответвления или метки».

Вы можете найти старую версию файла или папки в диалоге журнала и скопировать её в новое место в хранилище непосредственно из диалога журнала при помощи Контекстное меню → Создать ответвление/метку из ревизии. Прочтите Раздел 4.9.3, «Получение дополнительной информации», чтобы узнать об этом больше.

Можно также использовать обозреватель хранилища для обнаружения нужных вам файлов, и скопировать их в рабочую копию непосредственно из хранилища, или скопировать их из одного места в другое внутри хранилища. Прочтите Раздел 4.24, «Обозреватель хранилища», чтобы узнать, как это сделать.

Невозможно выполнять копирование между хранилищами

В то время, как вы можете копировать и файлы, и папки *внутри* хранилища, вы *не можете* выполнять копирование или перемещение из одного хранилища в другое с сохранением истории при помощи TortoiseSVN. Даже если хранилища расположены

на одном и том же сервере. Всё что возможно сделать - скопировать содержимое в текущем состоянии и добавить его как новое содержимое во второе хранилище.

Если вы не уверены, относятся ли два адреса URL на одном и том же сервере к одному или разным хранилищам, воспользуйтесь Обозревателем хранилища, чтобы открыть эти URL и посмотреть, где находится корень хранилища. Если возможно увидеть оба местоположения в одном окне обозревателя хранилища, значит они в одном и том же хранилище.

4.13. Игнорирование файлов и папок

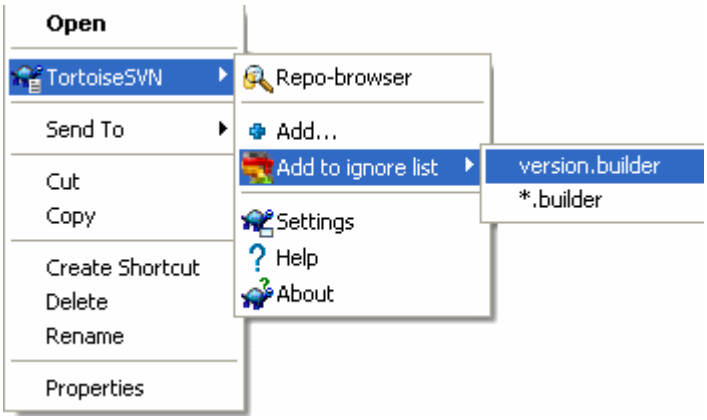


Рисунок 4.28. Контекстное меню Проводника для неверсированных файлов

В большинстве проектов у вас будут файлы и папки, которые не надо вносить под управление версиями. Это могут быть включаемые файлы, создаваемые компилятором, `*.obj`, `*.lst`, возможно, папки, в которых сохраняются создаваемые выполняемые файлы. Всякий раз, когда вы фиксируете изменения, TortoiseSVN показывает вам неверсированные файлы, заполняющие до отказа список файлов в диалоге фиксации. Конечно, вы можете вообще их не отображать, но тогда вы можете забыть добавить новый файл.

Лучший способ избежать этой проблемы - добавить воспроизводимые файлы в список игнорирования проекта. Таким образом они никогда не будут отображаться в диалоге фиксации, но настоящие неверсированные файлы будут всё-таки замечены.

Если вы щёлкните правой клавишей мыши на одиночном неверсированном файле, и выберите из контекстного меню команду TortoiseSVN → Добавить в список игнорирования, появится подменю, позволяющее вам выбрать, добавить ли в список только этот файл, или же добавить все файлы с таким же расширением. Если вы выберите несколько файлов, подменю не появится, и вы сможете добавить только эти конкретные файлы/папки.

Если вы желаете удалить один или несколько элементов из списка игнорирования, выполните правый щелчок на этих элементах и выберите TortoiseSVN → Удалить из списка игнорирования Вы также можете обратиться к свойству папки `svn:ignore` напрямую. Это позволит вам указать более общие шаблоны, используя универсализацию имён файлов, описываемую далее. Более подробную информацию об установке свойств содержит Раздел 4.17, «Установки проекта». Обратите внимание: каждый шаблон игнорирования должен быть в отдельной строке, разделение их пробелами не работает.

Глобальный список игнорирования

Другой путь игнорирования файлов - добавить их в *глобальный список игнорирования*. В этом случае самое большое отличие в том, что глобальный список

игнорирования - это клиентское свойство. Оно применяется *ко всем* проектам Subversion, но только на этом клиентском компьютере. В общем случае, лучше использовать свойство `svn:ignore` когда возможно, так как оно может быть применено к конкретным разделам проекта, и оно работает для всех извлекающих этот проект. Более подробную информацию смотрите в Раздел 4.30.1, «Общие настройки».

Игнорирование версированных файлов и папок

Версированные файлы и папки не могут игнорироваться - так устроена Subversion. Если вы версировали файл по ошибке, прочтите Раздел B.8, «Игнорировать файлы, которые уже версированы», где приведены инструкции, как сделать его «неверсированным».

4.13.1. Сопоставление шаблону в списках игнорирования

Шаблоны игнорирования в Subversion применяют универсализацию имён файлов - способ, первоначально задействованный в Unix для указания нужных файлов и использующий мета-символы для обобщения. Следующие символы имеют специальное значение:

*
: Соответствует любой строке, включая пустую строку (без символов).

?
: Соответствует любому одиночному символу.

[...]
: Соответствует любому символу, заключённому в квадратные скобки. Внутри скобок пара символов, разделённая «-» соответствует любому символу, лексически расположенному между ними. Например, `[AGm-p]` соответствует любому из `A`, `G`, `m`, `n`, `o` или `p`.

Сопоставление с шаблоном выполняется с учётом регистра, и это может вызвать проблемы в Windows. Вы можете добиться независимости от регистра трудозатратным способом: задавая все символы парами, т.е. для независимого от регистра игнорирования `*.tmp`, вы можете использовать шаблон `*.[Tt][Mm][Pp]`.

Если вам необходимо официальное описание универсализации, вы можете найти его в спецификации IEEE для командного языка оболочки *Pattern Matching Notation* [http://www.opengroup.org/onlinepubs/009695399/utilities/xcu_chap02.html#tag_02_13].

Никаких путей в глобальном списке игнорирования

Не надо включать полный путь в задаваемый шаблон. Сопоставление с шаблоном предназначено для использования с обыкновенными именами файлов и папок. Если вы желаете игнорироавать все папки `CVS`, просто добавьте `CVS` в список игнорирования. Нет необходимости указывать `CVS */CVS`, как в более ранних версиях. Если же вы желаете игнорировать все папки `tmp`, которые находятся в каталоге `prog`, но не в `doc`, вам необходимо воспользоваться свойством `svn:ignore`. Не существует надёжного способа добится этого при помощи глобальных шаблонов игнорирования.

4.14. Удаление, перемещение и переименование

В отличие от CVS, Subversion позволяет переименовывать и перемешать файлы и папки. Поэтому в подменю TortoiseSVN есть специальные пункты для удаления и переименования.

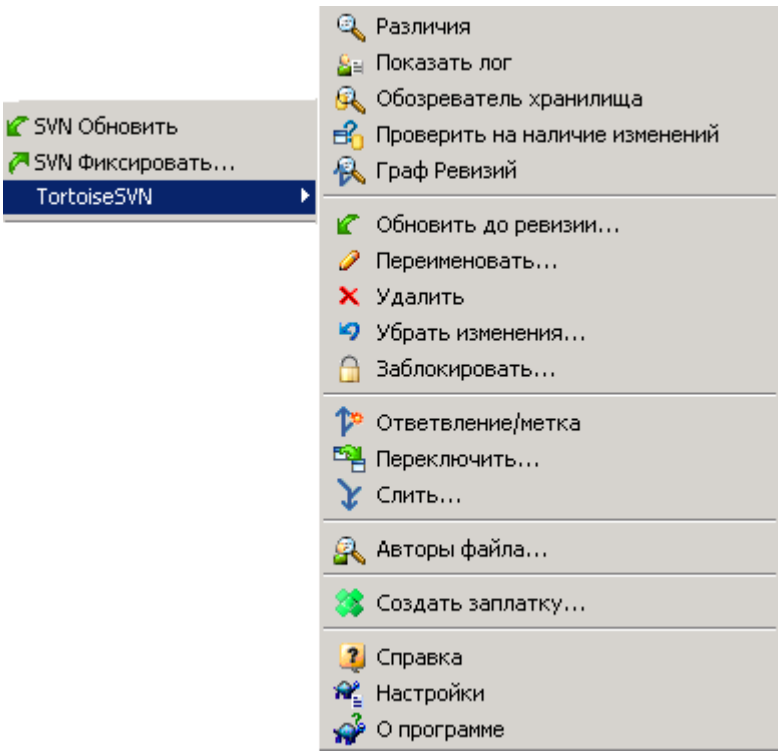


Рисунок 4.29. Контекстное меню Проводника для версированных файлов

4.14.1. Удаление файлов и папок

Для удаления файлов и папок из Subversion применяется команда TortoiseSVN → Удалить.

Когда вы решаете TortoiseSVN → Удалить файл, он сразу же убирается из вашей рабочей копии и помечается для удаления в хранилище при следующей фиксации. Родительская папка этого файла отображается с пометкой «удалённый». До тех пор, пока не произведена фиксация, вы можете вернуть файл обратно, если вызовете TortoiseSVN → Убрать изменения на родительской папке.

Когда вы решаете TortoiseSVN → Удалить папку, она остаётся в рабочей копии, но пометка на ней изменяется, обозначая, что она будет удалена. До тех пор, пока не произведена фиксация, вы можете вернуть папку обратно, если вызовете TortoiseSVN → Убрать изменения на самой этой папке. За эту разницу в работе с файлами и папками ответственна Subversion, а не TortoiseSVN.

Если вы желаете удалить какой-нибудь объект из хранилища, но в то же время оставить его локально как неверсированный файл/папку, воспользуйтесь Расширенное контекстное меню → Удалить (оставив локально). Вам необходимо удерживать клавишу **Shift** при правом щелчке на объекте в панели со списком файлов Проводника (правая панель) для того, чтобы увидеть этот пункт в расширенном контекстном меню.

Если *файл* удаляется в Проводнике, а не при помощи контекстного меню TortoiseSVN, диалог фиксации отобразит этот файл и позволит вам удалить его также из под управления версиями перед фиксацией. Однако, если вы обновите вашу рабочую копию, Subversion обнаружит отсутствующий файл и заменит его последней версией из хранилища. Если вам необходимо удалить файл, находящийся под управлением версиями, всегда используйте TortoiseSVN → Удалить, чтобы Subversion не приходилось угадывать, что вы хотите сделать на самом деле.

Если *папка* удаляется в Проводнике, а не при помощи контекстного меню TortoiseSVN, ваша рабочая копия будет повреждена и вы не сможете фиксировать изменения. Если вы обновите

вашу рабочую копию, Subversion заменит отсутствующие папки их последними версиями из хранилища, и после этого вы сможете удалить их правильно, используя TortoiseSVN → Удалить.

Возвращение назад удалённого файла или папки

Если вы удалили файл или папку и уже зафиксировали эту операцию в хранилище, тогда обычное выполнение TortoiseSVN → Убрать изменения уже не может вернуть их назад. Но файл или папка не потеряны навсегда. Если вы знаете ревизию, в которой файл или папка были удалены (если не знаете, найдите при помощи диалога журнала), откройте обозреватель хранилища и перейдите к этой ревизии. Затем выберите файл или папку, которую вы удалили, щёлкните правой кнопкой и выберите Контекстное меню → Копировать в..., в качестве цели операции копирования выберите путь к вашей рабочей копии.

4.14.2. Перемещение файлов и папок

Если вы желаете просто переименовать (без перемещения) файл или папку , используйте Контекстное меню → Переименовать... Введите новое имя переименуемого объекта и это всё.

Если вы желаете переместить файлы внутри рабочей копии, возможно, в другую подпапку, воспользуйтесь обработчиком перетаскивания правой клавишей мыши:

1. выберите файлы или папки, которые вы желаете переместить
2. затем перетащите правой кнопкой мыши их на новое место внутри рабочей копии
3. отпустите правую кнопку мыши
4. в появившемся меню выберите Контекстное меню → SVN Переместить версированные файлы сюда

Фиксируйте родительскую папку

Поскольку переименование и перемещение выполняются как удаление с последующим добавлением, вам необходимо выполнить фиксацию родительской папки перемещённого/удалённого файла, так чтобы удаляемая часть переименования/перемещения отображалась в диалоге фиксации. Если вы не зафиксируете удаляемую часть переименования/перемещения, она останется в хранилище и у тех, кто работает вместе с вами, при обновлении старые файлы удалены не будут, т.е. у них окажутся *обе* копии: и старая, и новая.

Вы *должны* зафиксировать переименование папки перед изменением любого файла внутри этой папки, иначе ваша рабочая копия может реально прийти в беспорядок.

Можно использовать также обозреватель хранилища для перемещения файлов и папок. Чтобы узнать больше о том, как это сделать, прочтите Раздел 4.24, «Обозреватель хранилища».

Не перемещайте внешнее при помощи SVN

Не надо применять команды TortoiseSVN Переместить или Переименовать к папкам, созданным с использованием `svn:externals`. Это действие приводит к удалению внешних элементов из их родительского хранилища, вероятно вызывая

замешательство у множества других людей. Если вам необходимо переместить папку с внешним, то надо использовать обычное перемещение в оболочке (например, Проводнике), а затем настроить свойство `svn:externals` исходной и целевой родительских папок.

4.14.3. Изменение регистра символов в имени файла

Изменение только регистра символов в имени файла при помощи Subversion под Windows требует применения хитрости, поскольку во время переименования на короткое время должны существовать оба имени. А так как файловая система Windows нечувствительна к регистру, это не сработает при использовании обычной команды переименования.

К счастью, существует (по крайней мере) два возможных способа переименования файла без потери его истории изменений. Важно переименовать его при помощи Subversion. Простое переименование в Проводнике испортит вашу рабочую копию!!!

Решение А) (рекомендуемое)

1. Зафиксируйте изменения в вашей рабочей копии.
2. Переименуйте файл по имени ВЕРХНИЙрегистр в верхнийРЕГИСТР непосредственно в хранилище при помощи обозревателя хранилища.
3. Обновите вашу рабочую копию.

Решение Б)

1. Переименуйте файл по имени ВЕРХНИЙрегистр в ВЕРХНИЙрегистр_ при помощи команды 'переименовать' из подменю TortoiseSVN.
2. Зафиксируйте изменения.
3. Переименуйте файл ВЕРХНИЙрегистр_ в верхнийРЕГИСТР.
4. Зафиксируйте изменения.

4.14.4. Как справиться с конфликтами из-за регистра символов в именах файлов

В случае, когда у вас в хранилище есть два файла с одинаковыми именами, различающиеся только регистром (например, `TEST.TXT` и `test.txt`), вы больше не сможете обновить или извлечь папку, содержащую эти файлы, при помощи клиента под Windows. Хотя Subversion и поддерживает имена файлов, различающиеся регистром, их не поддерживает Windows.

Иногда это случается, когда два человека фиксируют из двух различных рабочих копий файлы, имеющие одинаковые имена, но отличающиеся регистром символов. Это также может случиться при фиксации файлов из ОС, файловая система которой учитывает регистр, такой как Linux.

В этом случае вам необходимо решить, какой из них вы желаете сохранить и удалить (или переименовать) другой из хранилища.

Предотвращение двух одинаковых имён у файлов

По адресу *http://svn.collab.net/repos/svn/trunk/contrib/hook-scripts/* находится скрипт ловушки для сервера, предотвращающий фиксации, в результате которых возникнут конфликты из-за регистра символов.

4.14.5. Исправление переименования файлов

Иногда ваша дружественная IDE переименовывает для вас файлы в процессе осуществления рефакторинга, и, конечно же, не сообщает об этом Subversion. При попытке зафиксировать изменения, Subversion будет видеть файл со старым именем как отсутствующий, а с новым - как неверсированный. Конечно, вы можете пометить новое имя для того, чтобы оно было добавлено, но тогда будет потеряна история изменений, поскольку Subversion не знает о взаимосвязи двух этих файлов.

Лучший способ - сообщить Subversion о том, что это изменение - на самом деле переименование, и это можно сделать и в диалоге Фиксация, и в диалоге Проверка на наличие изменений. Просто выделите оба имени: старое имя (отсутствующее) и новое имя (неверсированное), и примените Контекстное меню → Поправить переименование для обозначения этой пары в качестве переименования.

4.14.6. Удаление неверсированных файлов

Обычно ваш список игнорирования настроен так, чтобы Subversion игнорировала все генерируемые файлы. Но что, если вы желаете очистить все эти игнорируемые элементы для порождения чистой сборки? Как правило, вы настраиваете это в вашем сборочном файле, но если вы отлаживаете сборочный файл, или изменяете систему сборки, полезно иметь способ очистки места действия.

TortoiseSVN предоставляет именно такую возможность, применяя Расширенное контекстное меню → Удалить неверсированные элементы.... Вам необходимо удерживать клавишу **Shift** при правом щелчке на папке в панели со списком Проводника (правой панели) для того, чтобы увидеть этот пункт в расширенном контекстном меню. Это выводит диалог, в котором будут перечислены все неверсированные файлы со всей вашей рабочей копии, и вы сможете отметить или разотметить элементы для удаления.

При удалении такого рода элементов используется корзина, поэтому, если вы совершили ошибку и удалили файл, который должен быть версирован, вы всё ещё можете получить его обратно.

4.15. Отмена изменений

Если вы желаете отменить все изменения, сделанные вами в файле после его последнего обновления, вам надо отметить файл, правым щелчком вызвать контекстное меню, и затем выбрать команду TortoiseSVN → Убрать изменения Появится диалог, показывающий изменённые вами файлы, которые вы можете вернуть в исходное состояние. Отметьте те, которые вы желаете вернуть и нажмите ОК.

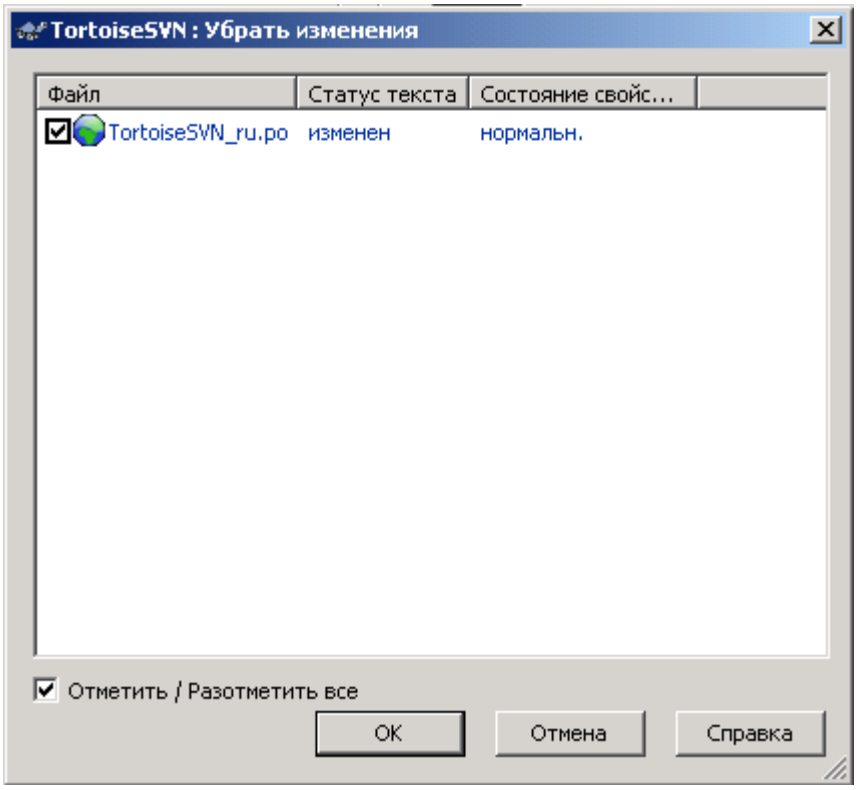


Рисунок 4.30. Диалог 'Убрать изменения'

Если вы желаете отменить удаление или переименование, вам необходимо использовать команду 'Убрать изменения' на родительской папке, поскольку удалённый элемент не существует и вам не на чем сделать правый щелчок.

Если вы желаете отменить добавление элемента, то в контекстном меню для этого есть команда TortoiseSVN → Отменить добавление.... На самом деле это та же команда 'Убрать изменения', но имя было изменено, чтобы сделать её предназначение более очевидным.

Столбцы в этом диалоге могут настраиваться таким же образом, как и столбцы в диалоге Проверка на наличие изменений. Прочтите Раздел 4.7.3, «Локальный и удалённый статус» если вам необходима дополнительная информация.

Отмена зафиксированных изменений

Команда Убрать изменения отменяет только ваши локальные изменения. Она *не отменяет* изменения, которые уже были зафиксированы. Если вы желаете отменить все изменения, которые были зафиксированы в конкретной ревизии, прочтите Раздел 4.9, «Диалоговое окно журнала ревизий» для дополнительной информации.

Убрание изменений работает медленно

При убирании изменений, вы можете обнаружить, что эта операция занимает намного больше времени, чем вы ожидали. Это происходит потому, что изменённая версия файла отправляется в корзину, чтобы вы могли получить ваши изменения обратно, если убрали их по ошибке. Однако, если ваша корзина заполнена, Windows тратит много времени на поиск места для файла. Решение простое: или очистите корзину, или отключите флажок Использовать корзину при убирании изменений в настройках TortoiseSVN.

4.16. Очистка

Если команда Subversion не может быть успешно завершена, возможно из-за проблем с сервером, ваша рабочая копия может остаться в несогласованном состоянии. В этом случае вам необходимо выполнить команду TortoiseSVN → Очистка для этой папки. Хорошей идеей будет выполнить эту команду также для папки самого верхнего уровня вашей рабочей копии.

Очистка имеет и другой полезный побочный эффект: если дата файла изменилась, а его содержимое - нет, Subversion не сможет сказать, был ли он в действительности изменён, кроме как выполнив побайтовое сравнение с нетронутой копией. Если у вас множество файлов в таком состоянии, это может привести к очень медленному получению статуса, которое, в свою очередь, вызывает замедление работы множества диалогов. Выполнение 'Очистки' на вашей рабочей копии исправит эти «нарушенные» отметки даты-времени и восстановит максимальную скорость проверки статуса.

Использовать дату-время фиксации

В некоторых ранних выпусках Subversion существовала ошибка, вызывавшая несоответствие временной метки при извлечении с установленным флажком Использовать дату-время фиксации. Используйте команду 'Очистить' для увеличения скорости работы с такими рабочими копиями.

4.17. Установки проекта

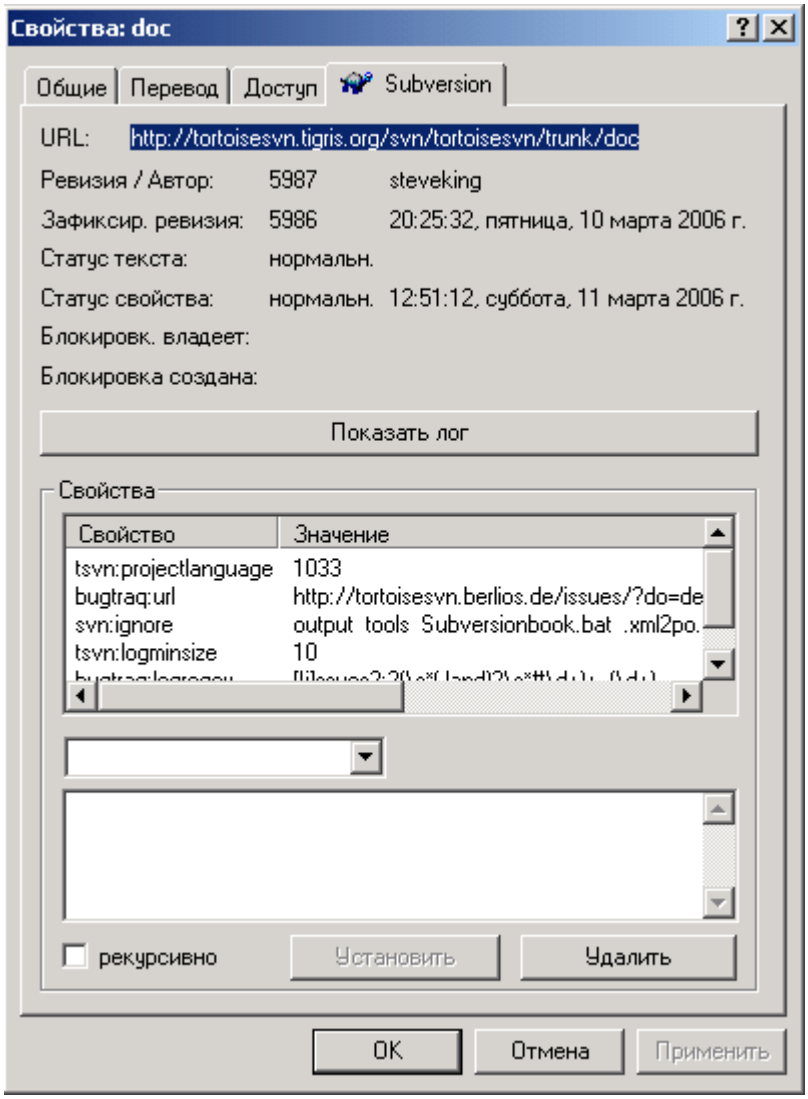


Рисунок 4.31. Страница свойств Проводника, вкладка Subversion

Иногда вам необходима более детальная информация о файле/папке, нежели пометка на значке. Вы можете получить всю информацию, предоставляемую Subversion, в диалоге свойств Проводника. Просто выберите из контекстного меню для нужного файла или папки Меню Windows → Свойства (обратите внимание: это обычный пункт 'Свойства' в контекстном меню Проводника, а не тот, который в подменю TortoiseSVN!). В диалоге свойств TortoiseSVN добавляет новую вкладку свойств для файлов/папок, находящихся под управлением Subversion, где вы можете посмотреть всю существенную информацию о выбранном файле/папке.

4.17.1. Свойства Subversion

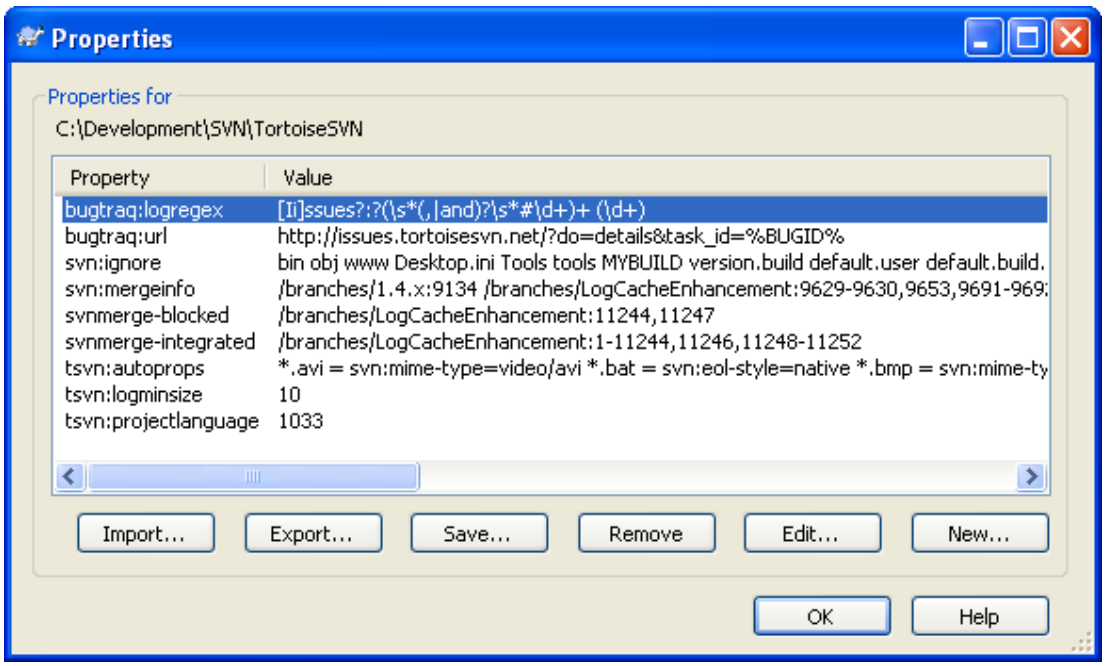


Рисунок 4.32. Страница свойств Subversion

Вы можете вызвать диалог для просмотра и установки свойств Subversion не только из диалога свойств Проводника Windows, но и также из TortoiseSVN → Свойства, и из списков состояния в различных диалогах TortoiseSVN, при помощи Контекстное меню → Свойства.

Вы можете добавить ваши собственные свойства, а также некоторые свойства, имеющие специальное значение в Subversion. Такие свойства начинаются с `svn:`. Одним из таких свойств является `svn:externals`; посмотреть, как обращаться с внешними включениями, можно в Раздел 4.18, «Внешние включения».

4.17.1.1. Ключевые слова, начинающиеся с svn:

Subversion поддерживает подстановку ключевых слов в стиле CVS, для внесения имени файла и информации о ревизии внутрь самого этого файла. Ключевые слова, поддерживаемые в данный момент:

$Date$
: Дата последней известной фиксации. Основывается на информации, полученной при обновлении рабочей копии. Хранилище *не проверяется* на наличие более поздних изменений.

$Revision$
: Ревизия последней известной фиксации.

$Author$
: Автор, выполнивший последнюю известную фиксацию.

$HeadURL$
: Полный URL этого файла в хранилище.

Id
: Сжатое сочетание предыдущих четырёх ключевых слов.

Чтобы узнать, как использовать эти ключевые слова, взгляните на *svn:keywords section (раздел о svn:keywords)* [http://svnbook.red-bean.com/en/1.5/svn.advanced.props.special.keywords.html] в

Книге о Subversion, в котором приведено полное описание этих ключевых слов, и также информация о том, как их задействовать и как использовать.

Чтобы больше узнать о свойствах в Subversion, прочтите *Special Properties (Специальные свойства)* [http://svnbook.red-bean.com/en/1.5/svn.advanced.props.html].

4.17.1.2. Добавление и редактирование свойств

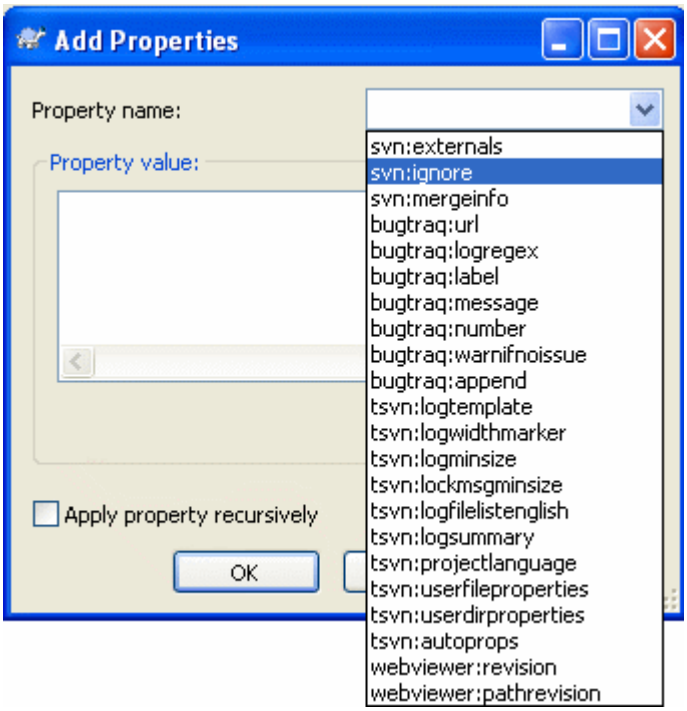


Рисунок 4.33. Добавление свойств

Для добавления нового свойства, сначала щёлкните на Добавить.... Выберите необходимое имя свойства из выпадающего списка, или вбейте имя по собственному усмотрению, а затем введите значение в нижерасположенное поле. Свойства, которые могут иметь множественные значения, такие как список игнорирования, можно ввести в нескольких строках. Нажмите OK для добавления этого свойства в список.

Если вы желаете применить свойство к нескольким элементам за одну операцию, выберите файлы/папки в Проводнике, затем вызовите Контекстное меню → Свойства.

Если вы желаете применить свойство к *каждому* файлу и папке, расположенному ниже текущей папки в иерархии, выберите флажок Применить свойство рекурсивно.

Некоторые свойства, например `svn:needs-lock`, могут быть применены только к файлам, и такие свойства не появляются в выпадающем списке для папок. Вы всё-таки можете применить такие свойства рекурсивно ко всем файлам в иерархии, но вы должны ввести имя свойства самостоятельно.

Если вы желаете отредактировать существующее свойство, выберите это свойство из списка существующих свойств, после чего нажмите Исправить....

Если вы желаете удалить существующее свойство, выберите это свойство из списка существующих свойств, после чего нажмите на Удалить.

Свойство `svn:externals` может быть использовано для включения других проектов из того же хранилища или же совсем другого хранилища. Чтобы больше узнать об этом, прочтите Раздел 4.18, «Внешние включения».

4.17.1.3. Экспорт и импорт свойств

Часто вам приходиться применять один и тот же набор свойств множество раз, например, свойство `bugtraq:logregex`. Для упрощения процесса копирования свойств из одного проекта в другой, можно воспользоваться возможностью экспорта/импорта.

Для файла или папки, у которых нужные свойства уже установлены, вызовите TortoiseSVN → Свойства, выберите свойства, которые вы желаете экспортировать, и щёлкните на Экспорт... У вас будет запрошено имя файла, в который будут сохранены имена и значения свойств.

Для папок, к которым вы желаете применить эти свойства, вызовите TortoiseSVN → Свойства и щёлкните на Импорт.... У вас будет запрошено имя файла, из которого будет производиться импорт, поэтому перейдите к месту, где вы до этого сохранили файл экспорта, и выберите его. Свойства будут добавлены к папкам нерекурсивно.

Если вы желаете добавить свойства к дереву рекурсивно, выполните вышеуказанные шаги, после чего в диалоге свойств выберите каждое свойство по очереди, щёлкните на Исправить..., отметьте флажок Применить свойство рекурсивно и щёлкните на ОК.

Формат файла импорта является двоичным и внутренним для TortoiseSVN. Его единственное предназначение - перенос свойств при помощи импорта и экспорта, поэтому не нужно редактировать эти файлы.

4.17.1.4. Двоичные свойства

TortoiseSVN может работать с двоичными свойствами при помощи файлов. Для считывания двоичного значения свойства, выполните Сохранить... это значение в файл. Для установки двоичного свойства, используйте шестнадцатеричный редактор или другой подходящий инструмент для создания файла с требуемым содержимым, а потом вы можете Загрузить... значение свойства из этого файла.

Хотя двоичные свойства не так уж часто используются, они могут оказаться полезными для некоторых применений. Например, если вы храните огромные графические файлы, или если приложение, используемое для загрузки файла, слишком велико, вы можете сохранить миниатюру в свойстве для того, чтобы можно было быстро выполнить предварительный просмотр.

4.17.1.5. Автоматическая установка свойств

Вы можете настроить Subversion и TortoiseSVN так, чтобы при добавлении файлов и папок в хранилище свойства для них устанавливались бы автоматически. Есть два способа это сделать:

Вы можете отредактировать файл настроек Subversion для включения этой функции в вашем клиенте. Во вкладке Общее в диалоге настроек TortoiseSVN есть кнопка 'Правка', открывающая этот файл. Файл настроек является простым текстовым файлом, управляющим работой некоторых функций Subversion. Вам надо изменить две вещи: сначала в разделе `miscellany` разкомментируйте строчку `enable-auto-props = yes`. Затем вам необходимо будет отредактировать раздел ниже для указания того, какие свойства к каким типам файлов надо добавлять. Этот метод является стандартной возможностью Subversion и работает в любом клиенте Subversion. Однако, он должен быть определён на каждом клиенте индивидуально - нет способа распространять эти настройки из хранилища.

Другой метод - установить свойство `tsvn:autoprops` на папках, как описано в следующем разделе. Этот способ работает только с клиентами TortoiseSVN, зато с его помощью автосвойства можно распространить во все рабочие копии при обновлении.

Какой бы метод вы не выбрали, вы должны учитывать, что автосвойства применяются только к файлам в момент их добавления в хранилище. Автосвойства не будут изменять свойства уже версированных файлов.

Если вы желаете быть абсолютно уверенными, что к новым файлам применяются правильные свойства, вы должны настроить в хранилище ловушку перед-обновлением для отклонения фиксаций, в которых не установлены обязательные свойства.

Фиксируйте свойства

В Subversion свойства являются версированными. После изменения или добавления свойства, вы должны зафиксировать ваши изменения.

Конфликты в свойствах

Если при фиксации изменений возникает конфликт из-за того, что другой пользователь изменил то же свойство, Subversion создаёт файл с расширением `.prej`. Удалите этот файл после улаживания конфликта.

4.17.2. Свойства проекта в TortoiseSVN

TortoiseSVN имеет несколько собственных специальных свойств, и они начинаются с `tsvn:`.

- `tsvn:logminsize` устанавливает минимальную длину сообщения журнала при фиксации. Если вы введёте сообщение короче, чем здесь указано, фиксация будет невозможна. Эта возможность очень полезна для напоминания вам о необходимости указания надлежащего сообщения с описанием для каждой фиксации. Если это свойство не установлено, или его значение равно нулю, допустимы пустые сообщения журнала.

  `tsvn:lockmsgminsize` устанавливает минимальную длину сообщения блокирования. Если вы введёте сообщение короче, чем здесь указано, блокирование будет невозможно. Эта возможность очень полезна для напоминания вам о необходимости указания надлежащего сообщения с описанием для каждой получаемой вами блокировки. Если это свойство не установлено, или его значение равно нулю, допустимы пустые сообщения блокирования.

- `tsvn:logwidthmarker` используется с проектами, требующими, чтобы строки в сообщении журнала не превышали некоторой максимальной длины (обычно 80 символов) до перевода строки. Установка этого свойства в ненулевое значение приводит к двум вещам в диалоге ввода сообщения журнала: помещает маркер для обозначения максимальной длины и запрещает перенос слов при отображении, так чтобы вы могли увидеть, не слишком ли длинный текст вы ввели. Обратите внимание: это свойство правильно работает, только если вы используете моноширинный шрифт (с фиксированной длиной символов) для сообщений журнала.

- `tsvn:logtemplate` используется с проектами, в которых установлены правила форматирования сообщений журнала. Свойство содержит многострочный текст, который будет вставлен в поле ввода сообщения при начале фиксации. Затем вы можете изменить его для внесения нужной информации. Обратите внимание: если вы также используете `tsvn:logminsize`, убедитесь, что установили в нём длину, превышающую шаблон, или вы утратите этот механизм защиты.

- В Subversion есть возможность задавать «автосвойства» (autoprops), которые будут применяться к свежедобавленным или импортированным файлам, исходя из их расширений. Для корректной работы этой возможности необходима установка соответствующих автосвойств у каждого клиента в его файле настроек Subversion. Автосвойства `tsvn:autoprops` можно установить на папках, и тогда они будут объединяться с локальными автосвойствами пользователя при импорте или добавлении файлов. Формат такой же, как и для автосвойств Subversion, например, `*.sh = svn:eol-style=native;svn:executable` устанавливает два свойства на файлах с расширением `.sh`

Если возникает конфликт между локальными автосвойствами и `tsvn:autoprops`, преимущество имеют настройки проекта, поскольку они предназначены специально для этого проекта.

- В диалоге фиксации есть возможность вставки в буфер обмена списка изменённых файлов вместе со статусом каждого файла (добавленный, измененный и т.п.). `tsvn:logfilelistenglish` определяет, будет ли состояние файлов указано на английском языке или локализовано. Если свойство не установлено, по умолчанию используется `true`.

- TortoiseSVN может использовать модули проверки орфографии, которые применяются также в OpenOffice и Mozilla. Если эти приложения у вас уже установлены, это свойство определяет, какой из модулей проверки использовать, т.е. на каком языке будут написаны сообщения журнала в вашем проекте. Свойство `tsvn:projectlanguage` задаёт языковой модуль, который будет использоваться для проверки орфографии при вводе сообщений журнала. Вы можете найти значения для вашего языка на этой странице: *MSDN: Language Identifiers* [http://msdn2.microsoft.com/en-us/library/ms776260.aspx].

  Вы можете ввести это значение в десятичной или шестнадцатеричной системе, для шестнадцатеричной используйте префикс `0x`. Например, американский английский может быть введён как `0x0409` или как `1033` [3].

- Свойство `tsvn:logsummary` применяется для извлечения части сообщения журнала, которая будет показана в диалоге журнала как краткая сводка этого полного сообщения.

  Значение свойства `tsvn:logsummary` должно содержать однострочное регулярное выражение, состоящее из одной группы. То, что соответствует этой группе, используется в качестве сводки.

  Например: `\[SUMMARY\]:\s+(.*)` возьмёт всё после «[SUMMARY]» из сообщения журнала и это будет использовано как сводка.

- Когда вы желаете добавить новое свойство, вы можете либо выбрать одно из выпадающего списка, либо вы можете ввести любое другое понравившееся имя для свойства. Если ваш проект использует несколько собственных свойств, и вы хотите, чтобы эти свойства появились в выпадающем списке (для избежания опечаток при вводе имени свойства), вы можете создать список ваших собственных свойств при помощи `tsvn:userfileproperties` и `tsvn:userdirproperties`. Примените эти свойства к папке. Когда вы будете редактировать свойства любого дочернего элемента, ваши собственные свойства появятся в списке среди предопределённых имён свойств.

Некоторые `tsvn:`-свойства требуют значений `true/false`. TortoiseSVN также понимает `yes` как синоним `true` и `no` как синоним `false`.

TortoiseSVN может интегрироваться с некоторыми средствами отслеживания ошибок. При этом используются свойства проекта, начинающиеся с `bugtraq:`. Прочтите Раздел 4.28, «Интеграция с системами отслеживания ошибок/проблем» для дополнительной информации.

TortoiseSVN также может интегрироваться с некоторыми работающими через веб-интерфейс средствами просмотра хранилища, используя для этого свойства, начинающиеся с `webviewer:`. Прочтите Раздел 4.29, «Интеграция со средствами просмотра хранилища, работающими через веб-интерфейс» для дополнительной информации.

Устанавливайте свойства проекта на папках

Эти специальные свойства проекта должны быть установлены на *папках*, для того чтобы эта система работала. При фиксации файла или папки свойства считываются из этой папки. Если свойства там не найдены, TortoiseSVN будет искать их выше по дереву папок, пока не достигнет неверсированной папки или корня дерева (например,

> C:\). Если вы можете быть уверены, что каждый пользователь извлекает только из одной папки (например, trunk/), а не из какой-то подпапки, тогда достаточно установить свойства для trunk/. Если вы в этом не уверены, тогда вы должны устанавливать свойства рекурсивно для каждой подпапки. Установка свойств ниже по иерархии проекта перекрывает свойства более высоких уровней (ближе к trunk/).
>
> *Только* для свойств проекта вы можете использовать флажок Рекурсивно для установки свойства для всех подпапок в иерархии, без установки его также для всех файлов.

При добавлении новых подпапок посредством TortoiseSVN, все свойства проекта, заданные в родительской папке, будут автоматически добавлены и к новой дочерней папке.

Предостережение

Хотя свойства проекта, реализованные в TortoiseSVN, исключительно полезны, они работают только с TortoiseSVN, и некоторые из них будут работать только в свежих версиях TortoiseSVN. Если люди, работающие над вашим проектом, пользуются также и другими клиентами Subversion, или, быть может, у них установлены старые версии TortoiseSVN, то, возможно, лучше использовать ловушки на стороне хранилища для принудительного применения политик проекта. Свойства проекта могут только помочь выполнить политику, но они не могут принудительно применить её.

4.18. Внешние включения

Иногда бывает полезно создать рабочую копию, состоящую из нескольких частей, извлечённых из различных источников. Например, бывает необходимо собрать различные файлы или папки из разных мест хранилища, или, возможно, вообще из различных хранилищ. Если вы желаете, чтобы у всех пользователей была одинаковая компоновка, вы можете определить свойства svn:externals, чтобы разместить указанные ресурсы в тех местах, где они нужны.

4.18.1. Внешние папки

Предположим, вы извлекаете рабочую копию /project1 в D:\dev\project1. Выберите папку D:\dev\project1, сделайте правый щелчок и выберите Меню Windows → Свойства в контекстном меню. Появится диалог 'Свойства'. Перейдите на вкладку Subversion. Здесь есть кнопка Свойства..., вызывающая диалог работы со свойствами. После появления диалога нажмите кнопку Добавить... и выберите свойство svn:externals из выпадающего списка, после чего введите URL-адрес хранилища в формате url папка или, если вы желаете указать конкретную ревизию, -rREV url папка. Вы можете добавить несколько внешних проектов, по одному в строке. Допустим, вы установили эти свойства для D:\dev\project1:

```
http://sounds.red-bean.com/repos  sounds
http://graphics.red-bean.com/repos/fast%20graphics  quick_graphs
-r21 http://svn.red-bean.com/repos/skin-maker  skins/toolkit
```

Нажмите на ОК и зафиксируйте ваши изменения. Когда вы (или кто-либо другой) обновите вашу рабочую копию, Subversion создаст подпапку D:\dev\project1\sounds и извлечёт проект sounds, другая подпапка D:\dev\project1\quick graphs будет содержать проект графики, и, наконец, вложенная подпапка D:\dev\project1\skins\toolkit будет содержать ревизию 21 проекта skin-maker.

Некоторые символы в URL должны быть правильно заэкранированы, иначе это работать не будет. Например, вы должны заменить каждый пробел на %20, как показано во втором примере выше.

Если вы желаете, чтобы локальные пути включали пробелы или другие специальные символы, вы можете заключить их в двойные кавычки, или использовать перед специальным символом символ \ (обратную косую черту) для экранирования в стиле оболочки Unix. Конечно же, это также означает, что вы должны использовать / (прямую косую черту) в качестве разделителя пути. Обратите внимание: такое поведение появилось в Subversion 1.6 и это не будет работать с более старыми клиентами.

Используйте явные номера ревизий

Вы должны серьёзно рассмотреть использование явного номера ревизии во всех ваших определениях внешних включений, как описано выше. Применение этого подхода означает, что решение, когда скачивать другую копию внешней информации, и какую конкретно копию, будете принимать вы. Помимо соображений здравого смысла, заключающихся в том, чтобы вы не будете неприятно удивлены изменениями в стороннем хранилище, управлять которым у вас нет возможности, использование явных номеров ревизий означает также, что при возвращении вашей рабочей копии к предыдущей ревизии определения внешних включений тоже вернутся к тому, какими они были в предыдущей ревизии, что, в свою очередь, означает, что внешние рабочие копии будут обновлены до того состояния, в котором *они* были тогда, когда ваше хранилище было на этой предыдущей ревизии. Для программных проектов это может означать разницу между успешной и не успешной сборкой старого снимка состояния вашей сложной базы исходного кода.

Старые определения svn:externals

Формат, показанный здесь, был введён в Subversion 1.5. Вы также могли видеть старый формат, в котором та же информация указывалась в другом порядке. Предпочтителен новый формат, поскольку он поддерживает несколько полезных возможностей, описанных ниже, но он не работает в старых клиентах. Различия приведены в *Книге о Subversion* [http://svnbook.red-bean.com/en/1.5/svn.advanced.externals.html].

Если внешний проект находится в том же хранилище, то любые изменения, которые вы в нём сделаете, будут включены в список фиксации при фиксировании основного проекта.

Если внешний проект находится в другом хранилище, вы будете уведомлены о сделанных вами изменениях во внешнем проекте при фиксировании основного проекта, но вы должны будете зафиксировать эти внешние изменения отдельно.

Если вы используете абсолютные URL в определениях `svn:externals`, и вам надо перебазировать вашу рабочую копию (т.е. URL вашего хранилища изменяется), то внешние включения не изменятся и возможно, больше работать не будут.

Во избежание таких проблем, клиенты Subversion, начиная с версии 1.5, поддерживают относительные внешние URL. Реализовано четыре метода указания относительных URL. В следующих примерах предполагается, что у нас есть два хранилища: одно по адресу `http://example.com/svn/repos-1` и другое по адресу `http://example.com/svn/repos-2`. У нас есть извлечено `http://example.com/svn/repos-1/project/trunk` в `C:\Working` и свойство `svn:externals` установлено для ствола.

Относительно родительской папки
: Такие URL всегда начинаются со строки `../` , например:

```
../../widgets/foo  common/foo-widget
```

Это будет извлекать `http://example.com/svn/repos-1/widgets/foo` в `C:\Working\common\foo-widget`.

Обратите внимание: URL указывается относительно URL папки со свойством `svn:externals`, а не папки, в которой внешнее записывается на диск.

Относительно корня хранилища
: Такие URL всегда начинаются со строки `^/` , например:

```
^/widgets/foo  common/foo-widget
```

Это будет извлекать `http://example.com/svn/repos-1/widgets/foo` в `C:\Working\common\foo-widget`.

Вы можете легко делать ссылки на другие хранилища с таким же `SVNParentPath` (общая папка, содержащая несколько хранилищ). Например:

```
^/../repos-2/hammers/claw  common/claw-hammer
```

Это будет извлекать `http://example.com/svn/repos-2/hammers/claw` в `C:\Working\common\claw-hammer`.

Относительно схемы
: URL, начинающиеся о строки `//` копируют только часть, относящуюся к схеме URL. Это может пригодится, когда к одному и тому же серверу необходимо получать доступ по разным схемам в зависимости от местоположения в сети; например, клиенты в интранет используют `http://`, тогда как внешние клиенты используют `svn+ssh://`. Например:

```
//example.com/svn/repos-1/widgets/foo  common/foo-widget
```

Это будет извлекать `http://example.com/svn/repos-1/widgets/foo` или `svn+ssh://example.com/svn/repos-1/widgets/foo` в зависимости от того, какой метод был использован для извлечения `C:\Working`.

Относительно имени сервера
: URL, начинающиеся со строки `/` копируют часть URL, содержащую схему и имя сервера, например:

```
/svn/repos-1/widgets/foo  common/foo-widget
```

Это будет извлекать `http://example.com/svn/repos-1/widgets/foo` в `C:\Working\common\foo-widget`. Но, если извлечь рабочую копию с другого сервера по адресу `svn+ssh://another.mirror.net/svn/repos-1/project1/trunk`, то внешнее включение будет извлекать `svn+ssh://another.mirror.net/svn/repos-1/widgets/foo`.

Вы также можете указать при необходимости опорную ревизию после URL, например `http://sounds.red-bean.com/repos@19`.

Если вам необходимо больше информации о том, как TortoiseSVN обходится со свойствами, прочтите раздел Раздел 4.17, «Установки проекта».

Для того, чтобы узнать о различных методах доступа к общим подпроектам, прочтите Раздел B.6, «Включить общий подпроект».

4.18.2. Внешние файлы

Начиная с Subversion 1.6, вы можете добавлять отдельные внешние файловые включения в вашу рабочую копию, используя тот же синтаксис, что и для папок. Однако, есть некоторые ограничения.

- Внешнее файловое включение должно помещать файл в уже извлечённую рабочую копию. Вообще, имеет смысл размещать файл непосредственно в папке, у которой задано `svn:externals`, но это при необходимости может быть и версированная подпапка. Для сравнения: внешние включения папок, напротив, автоматически создают все необходимые промежуточные неверсированные папки.

- URL внешнего файлового включения должен вести в то же хранилище, в которое внешнее файловое включение будет вставлено; междухранилищные внешние файловые включения не поддерживаются.

Внешнее файловое включение ведёт себя во многом как любой другой версированный файл, но оно не может быть перемещено или удалено при помощи обычных команд; вместо этого должно быть изменено свойство `svn:externals`.

Неполная поддержка внешних файловых включений в Subversion 1.6

В Subversion 1.6 невозможно убрать внешнее файловое включение из вашей рабочей копии после того, как вы его добавили, даже путём полного удаления свойства `svn:externals`. Вы должны выполнить новое извлечение рабочей копии для удаления файла.

4.19. Ответвления и метки

Одной из возможностей систем управления версиями является способность выделить изменения в отдельную линию разработки. Эта линия известна как *ответвление*. Ответвления часто используются для опробования новых возможностей без нарушения основной линии разработки ошибками компиляции и дефектами. Когда новые возможности достаточно устоятся, тогда ветка разработки *сливается* с основной ветвью (стволом).

Другой возможностью систем управления версиями является способность помечать частные ревизии (например, версию выпуска), так что вы сможете в любое время воссоздать конкретную сборку или окружение. Этот процесс известен как создание *метки*.

В Subversion нет специальных команд для работы с ответвлениями или метками, вместо этого используются так называемые «легкие копии» (cheap copies). Легкие копии похожи на жесткие ссылки в Unix: это означает, что вместо того, чтобы сделать полную копию в хранилище, создаётся внутренняя ссылка, указывающая на определённое дерево/ревизию. В результате, ответвления и метки создаются очень быстро, и почти не занимают дополнительного места в хранилище.

4.19.1. Создание ответвления или метки

Если вы импортировали ваш проект с рекомендованной структурой папок, создать ответвление или метку очень просто:

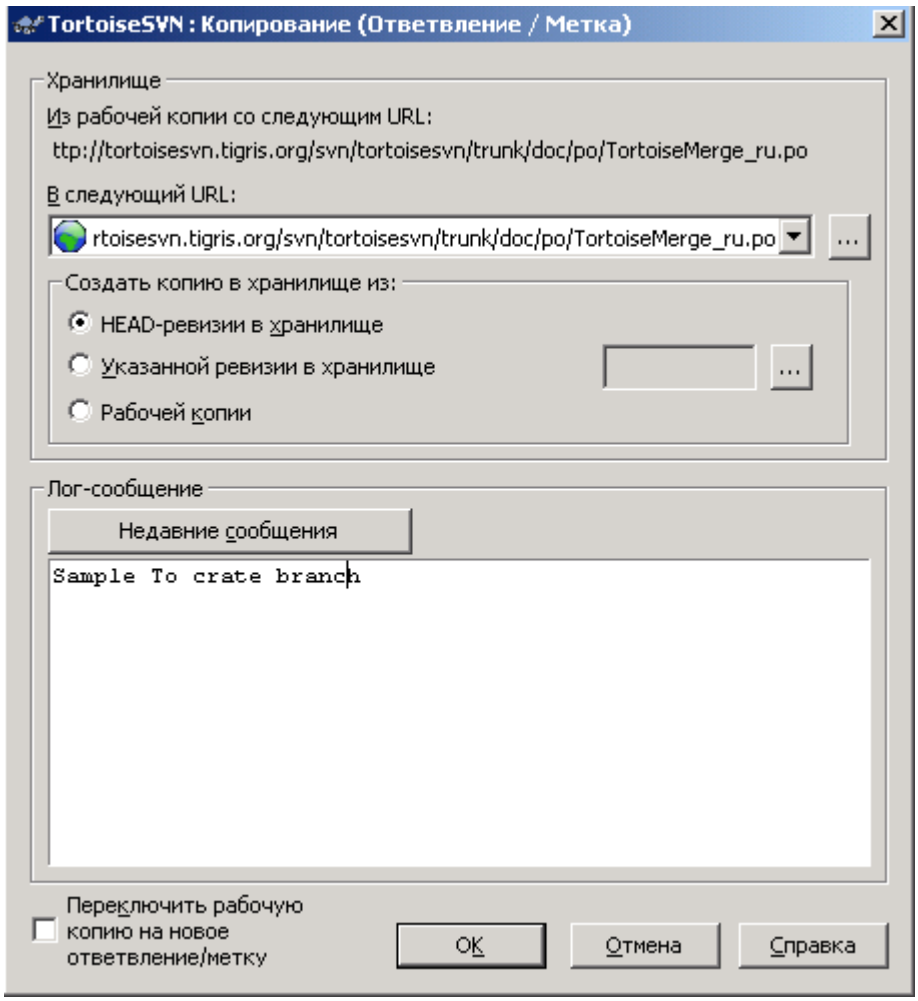


Рисунок 4.34. Диалог создания ответвления/метки

Выберите папку в рабочей копии, которую вы желаете скопировать в ответвление или пометить, затем выберите команду TortoiseSVN → Ответвление/Метка....

По умолчанию, целевым URL для нового ответвления будет URL источника, базового для вашей рабочей копии. Вы должны изменить этот URL так, чтобы он указывал новый путь вашего ответвления/метки. Так, вместо

```
http://svn.collab.net/repos/ProjectName/trunk
```

вы можете использовать что-то вроде

```
http://svn.collab.net/repos/ProjectName/tags/Release_1.10
```

Если вы не помните соглашений о наименовании, использованных в последний раз, нажмите кнопку справа - откроется обозреватель хранилища, и вы сможете посмотреть существующую структуру хранилища.

Теперь вы должны выбрать источник для копии. Здесь у вас есть три возможности:

Ведущая ревизия в хранилище (HEAD)
: В новое ответвление копируется ведущая ревизия непосредственно в хранилище. Не надо передавать никаких данных из вашей рабочей копии, и ответвление создаётся очень быстро.

Указанная ревизия в хранилище
: Новое ответвление копируется непосредственно в хранилище, но вы можете выбрать более старую ревизию. Это полезно, если вы забыли сделать метку, когда вы выпускали проект на прошлой неделе. Если вы не помните номер ревизии, нажмите кнопку справа для отображения журнала ревизий и выберите номер ревизии там. И в этом случае данные из вашей рабочей копии не предаются, ответвление создаётся очень быстро.

Рабочая копия
: Новое ответвление будет идентичной копией вашей локальной рабочей копии. Если вы обновили некоторые файлы до старых ревизий в вашей рабочей копии, или если вы сделали локальные изменения, именно это и войдёт в копию. Естественно, эта разновидность сложных меток может приводить к передаче данных из вашей рабочей копии обратно в хранилище, если их там пока нет.

Если вы желаете, чтобы ваша рабочая копия автоматически переключилась на вновь созданное ответвление, используйте флажок Переключить рабочую копию на новое ответвление/метку. Но если вы сделаете это, сначала убедитесь, что ваша рабочая копия не содержит изменений. Если содержит, эти изменения будут слиты с ответвлением при переключении.

Нажмите OK для фиксации новой копии в хранилище. Не забудьте ввести сообщение журнала. Обратите внимание: копия создаётся *внутри хранилища*.

Заметьте, что если только вы не выбрали переключение на свежесозданное ответвление, создание ответвления или метки *не затрагивает* вашу рабочую копию. Даже если вы создаёте ответвление из вашей рабочей копии, эти изменения фиксируются в новом ответвлении, а не в основном стволе, так что ваша рабочая копия всё ещё может быть помечена как изменённая по отношению к стволу.

4.19.2. Извлечь? Или переключиться?..

...вот в чём вопрос (хотя на самом деле нет). Если извлечение загружает всё из выбранного ответвления в хранилище в вашу рабочую папку, то TortoiseSVN → Переключить... передаёт только изменённые данные в вашу рабочую копию. Хорошо для загрузки сети, хорошо для вашего терпения. :-)

Для того, чтобы приступить к работе с вашей свежесгенерированным ответвлением или меткой, у вас есть несколько способов. Вы можете:

- TortoiseSVN → Извлечь её, создав новое извлечение в пустой папке. Вы можете извлечь в любое место на вашем локальном диске и вы можете создать столько рабочих копий из вашего хранилища, сколько вам нужно.
- Переключить вашу рабочую копию на вновь созданную копию в хранилище. Выберите папку самого верхнего уровня для вашего проекта и выполните TortoiseSVN → Переключить... из контекстного меню.

  В следующем диалоге введите URL ответвления, которое вы только что создали. Выберите в переключателе Ведущая ревизия (HEAD) и нажмите OK. Ваша рабочая копия будет переключена на новое ответвление/метку.

  Переключение работает точно как обновление, так что оно никогда не отбрасывает локальных изменений. Любые незафиксированные изменения, сделанные в рабочей копии, будут слиты при переключении. Если вы не желаете, чтобы это произошло, вы должны либо зафиксировать изменения перед переключением, либо откатить вашу рабочую копию до уже зафиксированной ревизии (обычно ведущей).
- Если вы желаете работать в стволе и в ответвлении, но не желаете выполнять затратное свежее извлечение, вы можете воспользоваться Проводником Windows для создания копии уже извлечённого ствола в другой папке, после чего TortoiseSVN → Переключить... эту копию на ваше новое ответвление.

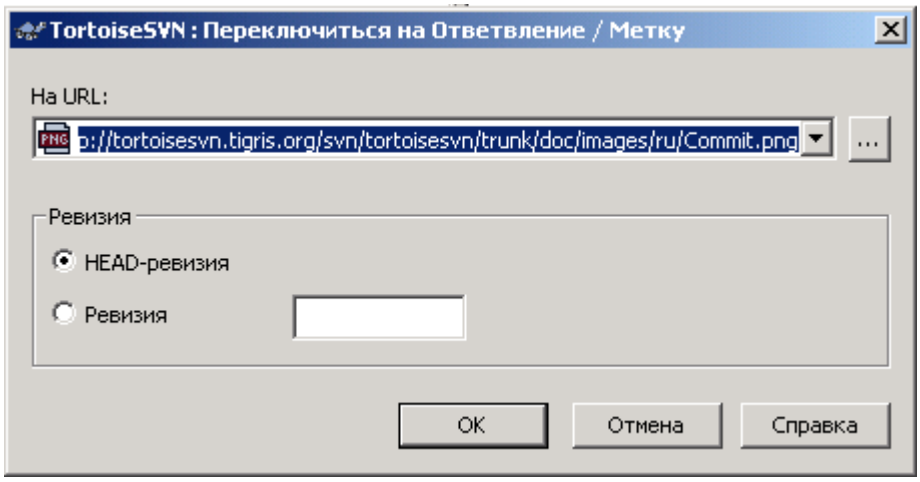


Рисунок 4.35. Диалог переключения

Несмотря на то, что сама Subversion не делает различий между метками и ответвлениями, типичные способы их использования немного отличаются.

- Метки, как правило, используются для создания статического снимка проекта на некоторой стадии. Сами они, как таковые, обычно не используются для разработки - для этого предназначены ответвления, и именно по этой причине мы изначально рекомендовали структуру хранилища в виде `/trunk /branches /tags (/ствол /ответвления / метки)`. Работа в помеченной ревизии *не очень хорошая идея*, но так как ваши локальные файлы не защищены от записи, нет ничего, что бы вас остановило, если вы это сделаете по ошибке. Однако, если вы попытаетесь выполнить фиксацию в хранилище по пути, в котором присутствует `/tags/`, TortoiseSVN вас предупредит.
- Возможно, вам понадобится произвести дальнейшие изменения в уже помеченную выпущенную версию. Правильным образом действий в такой ситуации будет сначала создать новое ответвление из этой метки, зафиксировать его, сделать ваши изменения в этом ответвлении, после чего и создать новую метку из этого нового ответвления, например `Version_1.0.1`.
- Если вы измените и зафиксируете созданную из ответвления рабочую копию, тогда все изменения попадут в новое ответвление, а *не в основной ствол*. Запоминаться будут только изменения, всё остальное останется в виде лёгкой копии.

4.20. Слияние

Там, где ответвления используются для обеспечения отдельных линий разработки, на некоторой стадии возникает необходимость произвести слияние сделанных в одном из ответвлений изменений со стволом, или наоборот, ствола с ответвлением.

Важно понимать, как слияние и ветвление работают в Subversion перед началом их использования, так как этот процесс может стать довольно сложным. Настоятельно рекомендуется прочитать главу *Branching and Merging (Ветвление и слияние)* [http://svnbook.red-bean.com/en/1.5/svn.branchmerge.html] в Книге о Subversion, которая содержит полное описание и множество примеров использования.

Обратите внимание на следующий момент: слияние *всегда* происходит в рабочей копии. Если вы желаете произвести слияние *с ответвлением*, у вас для этого ответвления должна быть извлечена рабочая копия, и мастер слияния необходимо вызывать из этой рабочей копии при помощи TortoiseSVN → Слить....

Вообще говоря, хорошей идеей является выполнять слияние с неизменённой рабочей копией. Если вы произвели какие-либо изменения в вашей рабочей копии, сначала зафиксируйте их. Если

слияние пойдёт не так, как вы ожидали, вы можете захотеть отменить изменения, и команда Убрать изменения уберёт *все* изменения, включая также и те, что вы сделали перед слиянием.

Вот три общих сценария использования слияния, которые производятся слегка различающимися способами, как описано ниже. Необходимый вам способ выбирается на первой странице мастера слияния.

Слияние с диапазоном ревизий
: Этот метод применяется в случае, когда вы сделали одну или несколько ревизий в ответвлении (или в основном стволе) и желаете перенести эти изменения и в другое ответвление.

  Получается, что в этой ситуации вы предписываете Subversion выполнить следующее: «Вычисли изменения, необходимые, чтобы [ОТ] ревизии 1 ответвления А перейти [ДО] ревизии 7 ответвления А, и примени эти изменения к моей рабочей копии (относящейся к стволу или ответвлению Б).»

Воссоединение с ответвлением
: Этот метод применяется в случае, когда вы сделали ответвление для разработки новой функции, как описано в книге по Subversion. Все изменения неделя за неделей переносились из ствола в это ответвление, и теперь, когда разработка функции завершена, вы желаете произвести её слияние обратно со стволом. Поскольку вы регулярно синхронизировали ответвление со стволом, последние версии ответвления и ствола будут полностью идентичными, за исключением ваших изменений в ответвлении.

  Это особый случай слияния деревьев, описанного ниже, и он требует только URL, откуда будет производиться слияние (обычно это ваше ответвление разработки). При этом используются возможности Subversion по отслеживанию слияний для вычисления корректных используемых диапазонов ревизий, и выполняются дополнительные проверки, чтобы убедиться, что ответвление было полностью обновлено и включает все изменения из ствола. Это сделано для того, чтобы вы случайно не убрали выполненную другими работу, зафиксированную ими в стволе после того, как вы последний раз синхронизировали изменения.

  После слияния вся разработка из ответвления полностью слита обратно с основной линией разработки. Ответвление теперь избыточно и может быть удалено.

  Как только вы выполнили воссоединительное слияние, вы больше не должны использовать это ответвление для разработки. Причина заключается в том, что при следующей попытке ресинхронизации существующего ответвления из ствола отслеживание слияний увидит это воссоединение как изменение в стволе, слияние которого с ответвлением ещё не было произведено, и попытается выполнить это слияние вида из-ответвления-в-ствол обратно в ответвление! Решением будет просто создание нового ответвления из ствола для следующей фазы вашей разработки.

Слияние двух различных деревьев
: Это более общий случай метода воссоединения. Вы предписываете Subversion выполнить следующее: «Вычисли изменения, необходимые, чтобы [ОТ] ведущей ревизии ствола перейти [ДО] ведущей ревизии ответвления, и примени эти изменения к моей рабочей копии (относящейся к стволу).» В конечном итоге ствол будет выглядеть точно так же, как и ответвление.

  Если ваш сервер/хранилище не поддерживают отслеживание слияний, то это единственный способ произвести слияние ответвления обратно в ствол. Другой типичный случай использования возникает при использовании ответвлений для кода сторонних производителей (vendor branches), когда вам необходимо произвести слияние изменений, возникающих после появления новой версии стороннего кода, с вашим кодом в стволе. Для дополнительной информации прочтите раздел об *ответвлениях для кода сторонних производителей* [http://svnbook.red-bean.com/en/1.5/svn.advanced.vendorbr.html] в Книге о Subversion.

4.20.1. Слияние с диапазоном ревизий

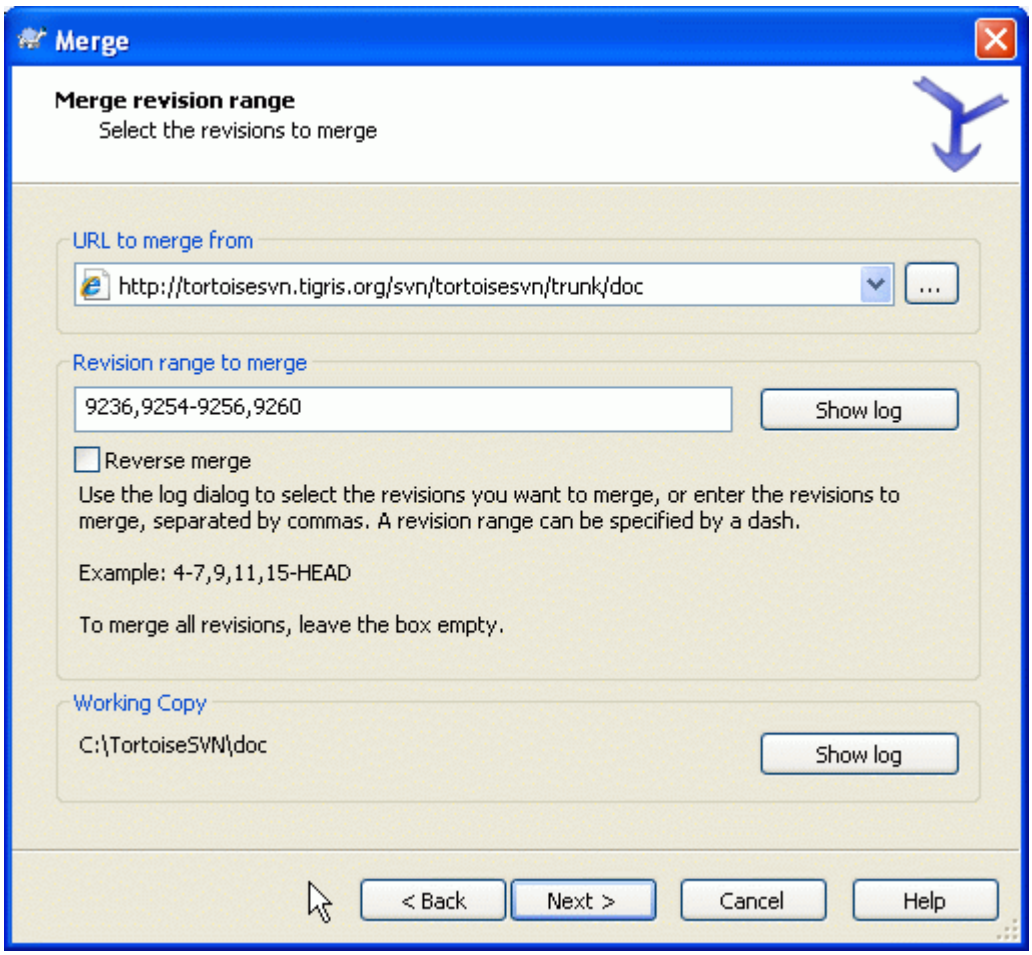


Рисунок 4.36. Мастер слияния - выбор диапазона ревизий

В поле От: введите полный URL папки ответвления или метки с изменениями, которые вы желаете перенести в вашу рабочую копию. Вы можете также нажать на ... для того, чтобы просмотреть хранилище и обнаружить нужное ответвление. Если вы уже производили слияние из этого ответвления, то просто воспользуйтесь выпадающим списком, в котором показывается история использованных ранее URL.

В поле Диапазон ревизий для слияния введите список ревизий, с которыми вы желаете произвести слияние. Это может быть одиночная ревизия, список разделённых запятыми конкретных ревизий, или диапазон ревизий, разделённых тире, или любая комбинация вышеперечисленного.

Важно

Существует важное отличие в способе указания диапазона ревизий между TortoiseSVN и клиентом командной строки. Наиболее лёгкий метод представить это - вообразить ограду со столбиками и пролётами.

В клиенте командной строки вы указываете изменения для слияния при помощи двух ревизий - «столбиков ограды», которые задают точки *до* и *после*.

В TortoiseSVN вы указываете набор изменений для слияния при помощи «пролётов ограды». Причина этого становится понятной при использовании диалога журнала для указания ревизий для слияния, где каждая ревизия выглядит как набор изменений.

> Если вы производите слияние ревизий большими кусками, показанный в Книге о Subversion метод предполагает, что вы сливаете ревизии 100-200 в первый заход и 200-300 в следующий. В TortoiseSVN вы сливаете 100-200 сначала и 201-300 потом.
>
> Это различие создаёт известный накал в списках рассылки. Мы признаём, что отличие от клиента коммандной строки есть, но мы полагаем, что большинству пользователей оконного интерфейса будет легче понять реализованный нами способ.

Простейший способ выбрать нужный вам диапазон ревизий - нажать на Журнал, поскольку будет выведен список последних изменений с соответствующими сообщениями журнала. Если вы желаете произвести слияние с изменениями из одной ревизии, просто выберите её. Если вы желаете произвести слияние с изменениями из нескольких ревизий, то выберите этот диапазон (применяя для этого, как обычно, клавишу **Shift**). Нажмите на OK и для вас будет сформирован список с номерами ревизий для слияния.

Если вы желаете произвести *обратное* слияние изменений, т.е. убрать из рабочей копии изменения, которые уже были зафиксированы, выберите ревизии, которые надо убрать, и убедитесь, что флажок Обратное слияние отмечен.

Если вы уже производили слияние некоторых изменений из этого ответвления, будем надеется, вы сделали заметку в сообщении журнала о последней ревизии, участвовавшей в слиянии, при фиксации изменений. В этом случае вы можете воспользоваться кнопкой Журнал для рабочей копии, чтобы обнаружить это сообщение. Используйте конечную точку в последнего слияния как начальную точку в новом. Например, если вы в последний раз сливали ревизии с 37 по 39, тогда начальной точкой для текущего слияния должна стать ревизия 40.

Если вы используете возможности Subversion по отслеживанию слияний, вам не нужно запоминать, какие ревизии уже были слиты - это для вас регистрирует Subversion. Если оставить диапазон ревизий пустым, то в слиянии будут участвовать все пока ещё не слитые ревизии. Прочтите Раздел 4.20.6, «Отслеживание слияний» для того, чтобы узнать об этом больше.

У других людей остаётся возможность фиксировать изменения, поэтому будьте внимательны при использовании ведущей ревизии. Она может оказаться совсем не той ревизией, которую вы считаете ведущей, если кто-нибудь ещё произведёт фиксацию после вашего последнего обновления.

Нажмите Далее и перейдите к Раздел 4.20.4, «Параметры слияния»

4.20.2. Воссоединение с ответвлением

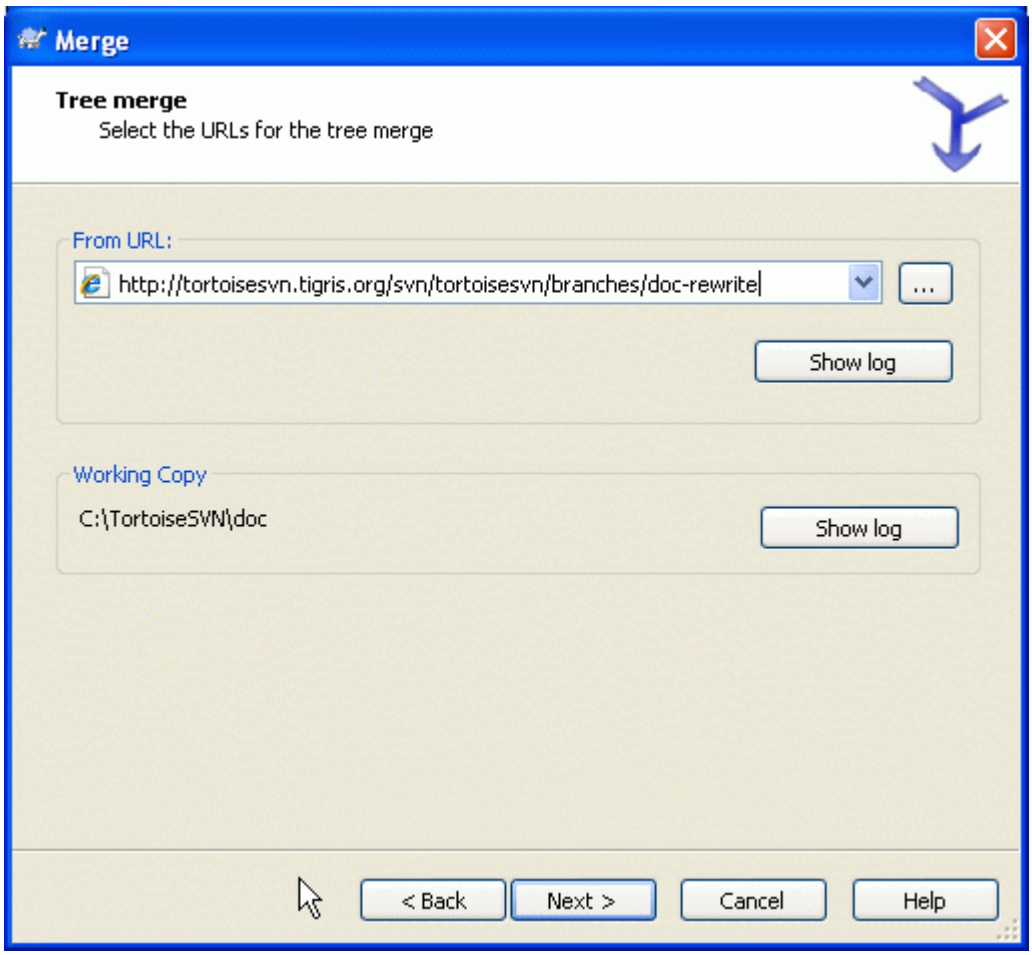


Рисунок 4.37. Мастер слияния - воссоединительное слияние

Для того, чтобы произвести слияние ответвления новой функции обратно со стволом, вы должны запустить мастер слияния из рабочей копии, извлечённой из ствола.

В поле Исходный URL: введите полный URL папки ответвления, которую вы желаете слить обратно со стволом. Вы также можете нажать ... для просмотра хранилища.

Несколько условий воссоединительного слияния. Для начала, сервер должен поддерживать отслеживание слияний. Рабочая копия должна быть с бесконечной глубиной извлечения (без разреженных извлечений), и в ней не должно быть локальных изменений, переключенных элементов, или элементов, обновлённых до ревизий, отличных от ведущей. Все изменения, произведённые в столе за время разработки в ответвлении, должны быть слиты обратно в ответвление (или помечены как слитые). Диапазон ревизий для слияния будет вычислен автоматически.

4.20.3. Слияние двух различных деревьев

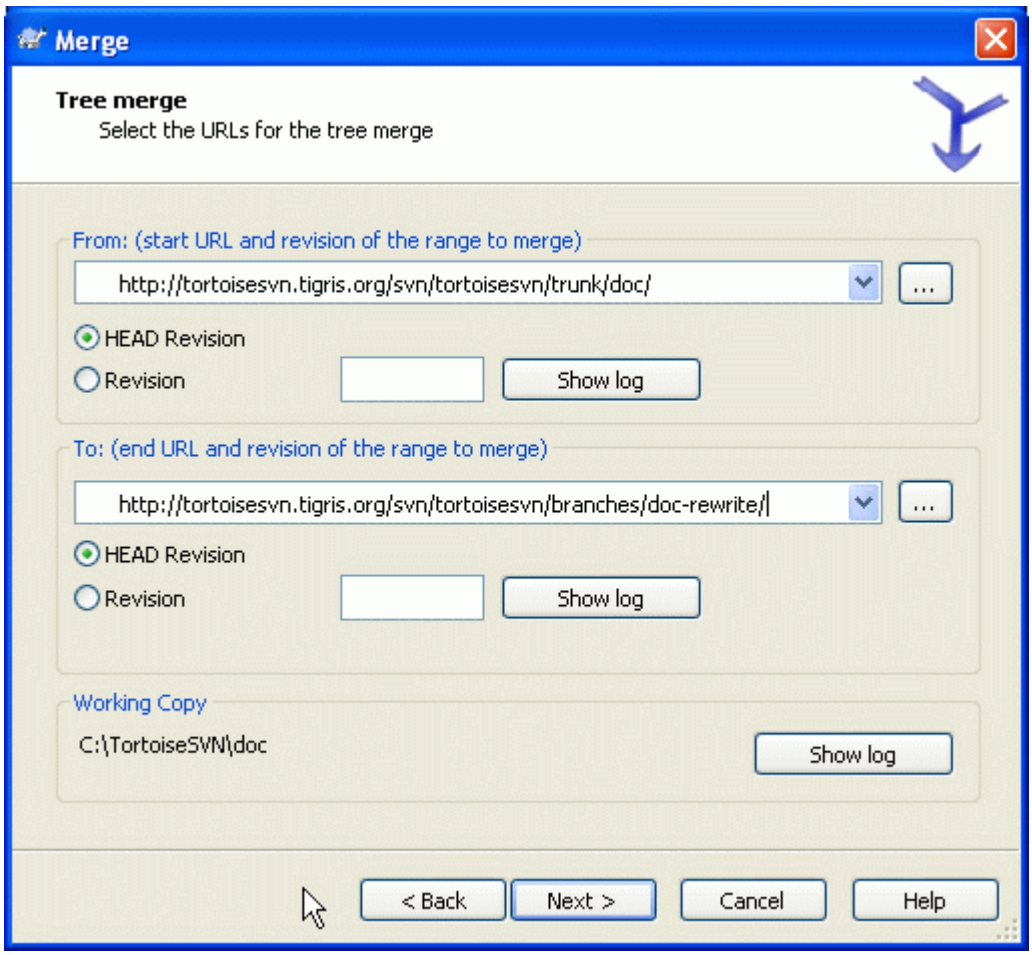


Рисунок 4.38. Мастер слияния - слияние деревьев

Если вы используете этот метод для слияния ответвления новой функции обратно со стволом, то вам необходимо запустить мастер слияния из рабочей копии, извлечённой из ствола.

В поле От: введите полный URL папки *ствола*. Это может звучать неправильно, но помните, что ствол - это начальная точка, в которую вы желаете добавить изменения из ответвления. Вы также можете нажать ... для просмотра хранилища.

В поле До: введите полный URL папки с ответвлением новой функции.

В оба поля, Начиная с ревизии и По ревизию, введите последний номер ревизии, в которой эти два дерева были синхронизированы. Если вы уверены, что больше никто не произведёт фиксацию, то можете использовать ведущую ревизию в обоих случаях. Если есть риск, что кто-либо мог сделать фиксацию после этой синхронизации, используйте конкретный номер ревизии во избежание потри более поздних фиксаций.

Для выбора ревизии вы также можете использовать кнопку Журнал.

4.20.4. Параметры слияния

На этой странице мастера вы можете перед запуском процесса слияния задать расширенные параметры. В большинстве случаев можно просто использовать настройки по умолчанию.

Вы можете указать глубину охвата, применяемую для слияния, т.е. насколько глубоко слияние должно затрагивать рабочую копию. Используемые термины для глубины охвата описаны в

Раздел 4.3.1, «Глубина извлечения». Глубина по умолчанию - Рабочая копия, использующая существующие установки глубины, что почти всегда является тем, что вам нужно.

В большинстве случаев вам необходимо, чтобы слияние учитывало историю файла, с тем чтобы сливались только изменения относительно общего предка. Иногда вам необходимо произвести слияние файлов, которые возможно и связаны, но не в вашем хранилище. Например, вы импортировали версии 1 и 2 сторонней библиотеки в две различные папки. Несмотря на то, что они логически связаны, Subversion об этом не знает, потому что видит только импортированные вами архивы. Если вы попытаетесь произвести слияние различий между двумя этими деревьями, вы увидите полное удаление с последующим полным добавлением. Для того, чтобы в Subversion использовались различия, основанные только на пути, вместо различий, основанных на истории, отметьте флажок Игнорировать предков. Больше прочитать на эту тему можно в Книге о Subversion *Noticing or Ignoring Ancestry (Учитывание или игнорирование предков)* [http://svnbook.red-bean.com/en/1.5/svn.branchmerge.advanced.html#svn.branchmerge.advanced.ancestry].

Вы можете задать способ обработки изменений завершений строк и пробельных символов. Эти параметры описаны в Раздел 4.10.2, «Параметры сравнения завершений строк и непечатаемых знаков». Поведение по умолчанию - считать все различия в завершениях строк и пробельных символах реальными изменениями, которые должны быть слиты.

Если вы используете отслеживание слияний, и вы желаете пометить ревизию как уже слитую, не выполняя слияние на самом деле, отметьте флажок Только зарегистрировать слияние. Сделать это вам может понадобится по двум причинам: случается, что слияние является слишком сложным для алгоритма слияния, поэтому вы производите нужные изменения вручную, а потом помечаете изменение (ревизию) как уже слитое, чтобы алгоритм слияния был об этом осведомлён. Или вы не желаете, чтобы определённая ревизия участвовала в слиянии. Пометив её как уже слитую, вы предотвратите слияние, если оно будет производиться клиентами, умеющими работать с информацией по отслеживанию слияний.

Теперь, когда всё подготовлено, единственное, что вам нужно сделать - нажать на кнопку Слияние. Если вы желаете предварительно посмотреть на результат, то воспользуйтесь кнопкой Пробный запуск, которая выполняет операцию слияния *без изменения* рабочей копии. После выполнения отображается список файлов, которые будут изменены при настоящем слиянии, и указываются области, где возникнут конфликты.

Диалог выполнения слияния показывает каждый этап слияния, с указанием вовлечённых диапазонов ревизий. При этом в диапазоне может быть показано на одну ревизию больше, чем вы могли бы ожидать. Например, если вы указали произвести слияние ревизии 123, в диалоге выполнения будет написано «Слияние с ревизиями с 122 по 123». Для понимания этого вам нужно помнить, что слияние тесно связано с различиями. Процесс слияния работает путём создания списка различий между двумя точками в хранилище с последующим применением этих различий к вашей рабочей копии. Диалог выполнения просто показывает начальные и конечные точки для получения различий.

4.20.5. Просмотр результатов слияния

Теперь слияние выполнено. Хорошей мыслью будет посмотреть на результаты слияния и проверить, прошло ли оно так, как ожидалось. Слияние обычно бывает довольно замысловатым; конфликты часто возникают, если ответвление далеко отошло от ствола.

Для клиентов и серверов Subversion версий меньше 1.5, информация о слиянии не сохранялась и слитые ревизии надо было отслеживать вручную: когда вы проверили изменения и собираетесь фиксировать эту ревизию, в вашем сообщении при фиксации *всегда* должны быть указаны номера ревизий, которые вы перенесли в этом слиянии. Если вы позже пожелаете произвести ещё одно слияние, вам будет необходимо знать, что вы уже сливали, поскольку вам не нужно переносить изменения более одного раза. К сожалению, информацию о слияниях Subversion не ведёт. Дополнительная информация об этом содержится в *Best Practices for Merging (Слияние. Лучший опыт)* [http://svnbook.red-bean.com/en/1.4/svn.branchmerge.copychanges.html#svn.branchmerge.copychanges.bestprac] в Книге о Subversion.

Если ваш сервер и все клиенты Subversion версии 1.5 или больше, система отслеживания слияний будет регистрировать слитые ревизии и позволит избежать ситуации, когда ревизия сливается более одного раза. Это делает вашу жизнь намного проще, поскольку вы можете просто производить каждый раз слияние всего диапазона ревизий и знать, что только новые ревизии действительно будут слиты.

Очень важен уход за ответвлениями. Если вы желаете поддерживать это ответвление в актуальном состоянии по отношению к стволу, вам необходимо производить слияния часто, чтобы ответвление и ствол не расходились далеко друг от друга. Конечно, вы должны избегать повторного слияния изменений, как объяснялось выше.

Подсказка

Если вы произвели слияние ответвления новой функции обратно в ствол, то ствол теперь содержит весь новый код новой функции, и ответвление больше не нужно и может быть удалено из хранилища по необходимости.

Важно

Subversion не может производить слияние файла с папкой и наоборот - только папки с папками и файлы с файлами. Если вы откроете диалог слияния для файла, тогда вы должны указать в нём путь к файлу. Если вы вызовите диалог для папки, тогда вы должны указать для слияния URL папки.

4.20.6. Отслеживание слияний

В Subversion 1.5 появились средства для отслеживания слияний. При слиянии изменений из одного дерева в другое, номера ревизий регистрируются и эта информация может быть использована в нескольких разных целях.

- Вы можете избежать опасности слияния одной и той же ревизии дважды (проблема повторного слияния). Как только ревизия получает пометку о том, что она уже сливалась, последующие слияния, содержащие в своём диапазоне эту ревизию, будут её пропускать.
- При слиянии ответвления обратно в ствол, в диалоге журнала могут быть показаны фиксации в ответвлении как часть журнала ствола, что позволяет лучше отслеживать изменения.
- Когда вы вызываете окно журнала из диалога слияния, уже слитые ревизии отображаются серыми.
- При показе информации об авторстве для файла можно выбрать, чтобы показывался первоначальный автор из слитых ревизий, а не тот, кто произвёл слияние.
- Можно пометить ревизии как *в слиянии не участвуют*, включив их в список слитых ревизий, но без проведения слияния на самом деле.

Информация по отслеживанию слияний регистрируется в свойстве `svn:mergeinfo` клиентом при проведении слияния. Когда слияние фиксируется, сервер регистрирует эту информацию в базе данных, и при запросе журнала, а также информации о слиянии или об авторстве сервер сможет предоставить соответствующие данные. Для того, чтобы система работала правильно, вы должны убедиться, что сервер и все клиенты обновлены до необходимых версий. Более ранние клиенты не регистрировали информацию в свойстве `svn:mergeinfo`, а более ранние серверы не предоставляли информацию, запрашиваемую новыми клиентами.

Больше узнать об отслеживании слияний можно из *документации об отслеживании слияний* [http://subversion.tigris.org/merge-tracking/index.html] Subversion.

4.20.7. Обработка конфликтов, возникающих при слиянии

Слияние не всегда проходит гладко. Иногда возникает конфликт, и, если вы производите слияние нескольких диапазонов, обычно вы желательно уладить конфликт до начала слияния со следующим диапазоном. TortoiseSVN помогает вам в этом процессе, отображая диалог *обратного вызова 'конфликты при слиянии'*.

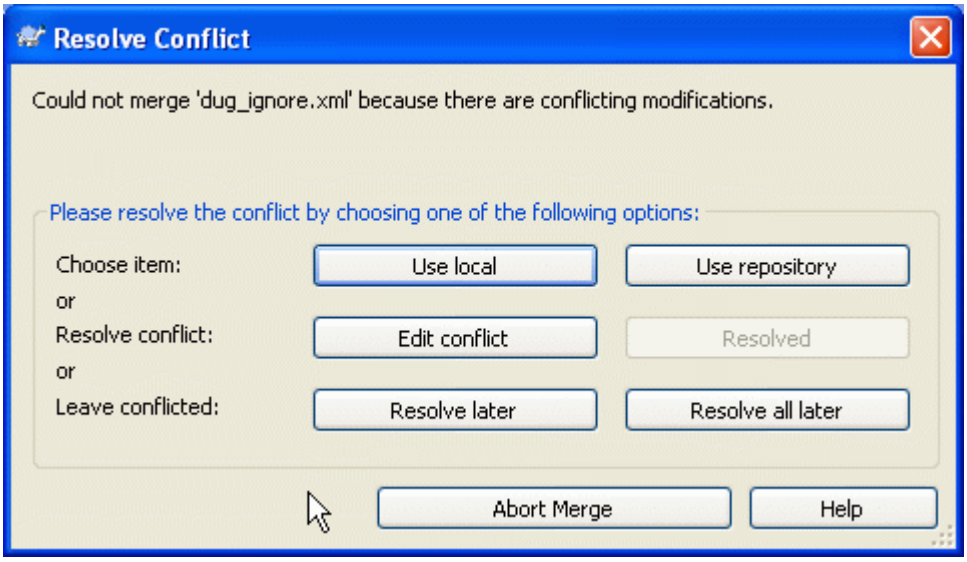


Рисунок 4.39. Диалог обратного вызова 'конфликты при слиянии'

При возникновении конфликта во время слияния, у вас есть три возможности для его обработки:

1. Вы можете решить, что ваши локальные изменения значительно более важны, и поэтому вы желаете отказаться от версии из хранилища и оставить вашу локальную версию. Или вы можете отказаться от ваших локальных изменений в пользу версии из хранилища. В любом случае, попыток выполнить слияние изменений производиться не будет - вы выбираете или одно, или другое.

2. Обычно вы желаете посмотреть на конфликты и уладить их. В этом случае, выберите Редактировать конфликт, которая запустит ваш инструмент слияния. Когда вы будете удовлетворены результатом, нажмите на Улажено.

3. Последняя возможность - отложить разрешение конфликта и продолжить слияние. Вы можете сделать это как с текущим конфликтующим файлом, так и со всеми оставшимися для слияния файлами. Однако, если в этом файле есть дальнейшие изменения, завершить слияние будет невозможно.

Если вы не желаете использовать эту интерактивную функцию обратного вызова, то для этого есть флажок Слияние без запросов в диалоге выполнения слияния. Если он отмечен при слиянии и слияние приводит к конфликту, файл помечается как конфликтующий и слияние продолжается. Вы будете должны уладить конфликты после того, как слияние полностью завершится. Если флажок очищен, то перед тем, как файл будет помечен как конфликтующий, у вас будет возможность уладить конфликт *во время* слияния. Преимущество этого в том, что если файл участвует в нескольких слияниях (несколько ревизий вносят изменения в этот файл), последующие слияния могут завершиться успешно, в зависимости от того, какие строки были затронуты. Но, конечно же, у вас не будет возможности уйти и приготовить себе чашечку кофе пока выполненяется слияние ;)

4.20.8. Слияние завершённого ответвления

Если вы желаете произвести слияние всех изменений из ответвления новой функции со стволом, то вы можете использовать пункт TortoiseSVN → Воссоединительное слияние... из расширенного контекстного меню (нажмите и удерживайте клавишу **Shift** при правом щелчке на файле).

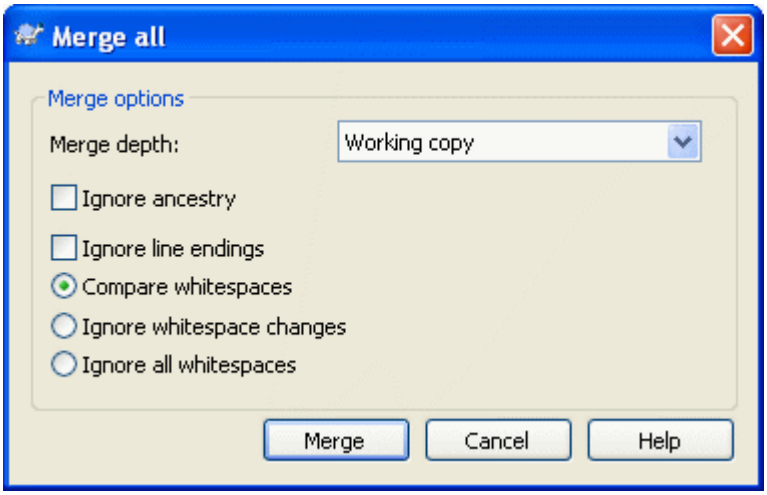


Рисунок 4.40. Диалог 'Воссоединительное слияние'

Использовать этот диалог очень просто. Всё, что вы должны сделать - это задать параметры для слияния, как описано в Раздел 4.20.4, «Параметры слияния». Остальное TortoiseSVN сделает самостоятельно, используя информацию об отслеживании слияний.

4.20.9. Сопровождение ответвления разработки новой возможности

Когда вы разрабатываете новую возможность в отдельном ответвлении, хорошей мыслью является выработка политики реинтеграции этой новой возможности по кончанию её разработки . Если в `стволе` в это же время выполняется другая работа, то вы можете обнаружить, что различия со времнем становятся всё более существенными, и их обратное слияние становится кошмаром.

Если новая возможность относительно простая и её разработка не займёт много времени, вы можете придерживаться простого подхода, который заключается в том, чтобы держать ответвление совершенно отдельным до завершения разработки, после чего слить изменения из ответвления со стволом. В мастере слияния это будет простое Слияние диапазона ревизий, в котором диапазоном ревизий будут ревизии, охватываемые ответвлением.

Если новая возможность скорее всего потребует заметного времени на разработку, и вам необходимо учитывать изменения из `ствола`, то вам будет необходимо производить синхронизацию ответвления. Это просто означает, что вы периодически сливаете изменения из ствола с ответвлением, так чтобы ответвление содержало все изменения из ствола *плюс* новую возможность. В процессе синхронизации используется Слияние диапазона ревизий. Когда новая возможность завершена, вы можете слить её обратно со стволом, используя или Воссоединение с ответвлением, или Слияние двух различных деревьев.

4.21. Блокирование

Как правило, Subversion работает лучше без блокировки, используя метод «Копирование-Изменение-Слияние», описанный ранее в Раздел 2.2.3, «Модель Копирование-Изменение-Слияние». Однако, есть несколько случаев, когда вам может потребоваться реализовать в некотором виде политику блокирования.

- Вы используете «необъединяемые» файлы - например, графические. Если двое людей изменяют один и тот же файл, слияние невозможно, поскольку изменения одного из них будут потеряны.

- Ваша компания в прошлом всегда использовала блокирующую систему контроля версий, и руководство решило, что «ничего не сравнится с блокированием».

Сначала вы должны убедиться, что ваш сервер Subversion обновлен до версии 1.2 как минимум. Более ранние версии вообще не поддерживали блокирования. Если вы используете доступ через `file://`, тогда, конечно, обновить потребуется только вашего клиента.

4.21.1. Как работает блокировка в Subversion

По умолчанию ничего не заблокировано, и любой, у кого есть доступ для фиксации, может фиксировать изменения любого файла в любое время. Другие периодически будут обновлять свои рабочие копии, и изменения в хранилище будут сливаться с локальными изменениями.

Если вы *заблокируете* файл, то только вы сможете зафиксировать этот файл. Фиксации других пользователей будут блокироваться до тех пор, пока вы не уберёте блокировку. Блокированный файл не может быть изменён в хранилище никаким способом, и это означает,что он не может быть удалён или переименован никем, кроме как владельцем блокировки.

Однако, другие пользователи могут и не знать, что вы получили блокировку. Если только они не проверяют регулярно состояние блокировок, то впервые они узнают об этом только когда их фиксация закончится неуспешно, что в большинстве случаев не очень практично. Для того, чтобы проще управлять блокировками, существует новое свойство Subversion `svn:needs-lock`. Если это свойство установлено (в любое значение) у файла, то всякий раз, когда файл извлекается или обновляется, локальная копия помечается как "только-для-чтения", *за исключением* случая, когда эта рабочая копия заблокировала этот файл. Это служит предупреждением о том, что вы не должны редактировать этот файл, пока не получите блокировку. Версированные файлы только-для-чтения отображаются со специальной пометкой в TortoiseSVN для обозначения того, что вы должны получить блокировку перед началом редактирования.

При регистрации блокировки используются данные о местоположении рабочей копии и о владельце блокировки. Если у вас есть несколько рабочих копий (на работе, дома), то вы можете владеть блокировкой только в *одной* из этих рабочих копий.

Что делать, если кто-либо из ваших коллег установил блокировку и уехал в отпуск, не сняв её? Subversion предоставляет средство для преодоления блокировки. Снятие блокировки, установленной кем-либо ещё, известно как *прерывание* блокировки, а принудительный захват блокировки, установленной кем-либо другим, называется *перехват* блокировки. Естественно, вы не должны делать это необдуманно, если вы желаете остаться друзьями со своими коллегами.

Блокировки регистрируются в хранилище, помимо этого создаётся также маркер блокировки в вашей локальной рабочей копии. Если возникает расхождение, например, если кто-либо прервал блокировку, локальный маркер блокировки становится неверным. Решающее слово всегда остаётся за хранилищем.

4.21.2. Получение блокировки

Выберите файл(ы) в рабочей копии, которые вы желаете заблокировать, после чего выполните команду TortoiseSVN → Заблокировать....

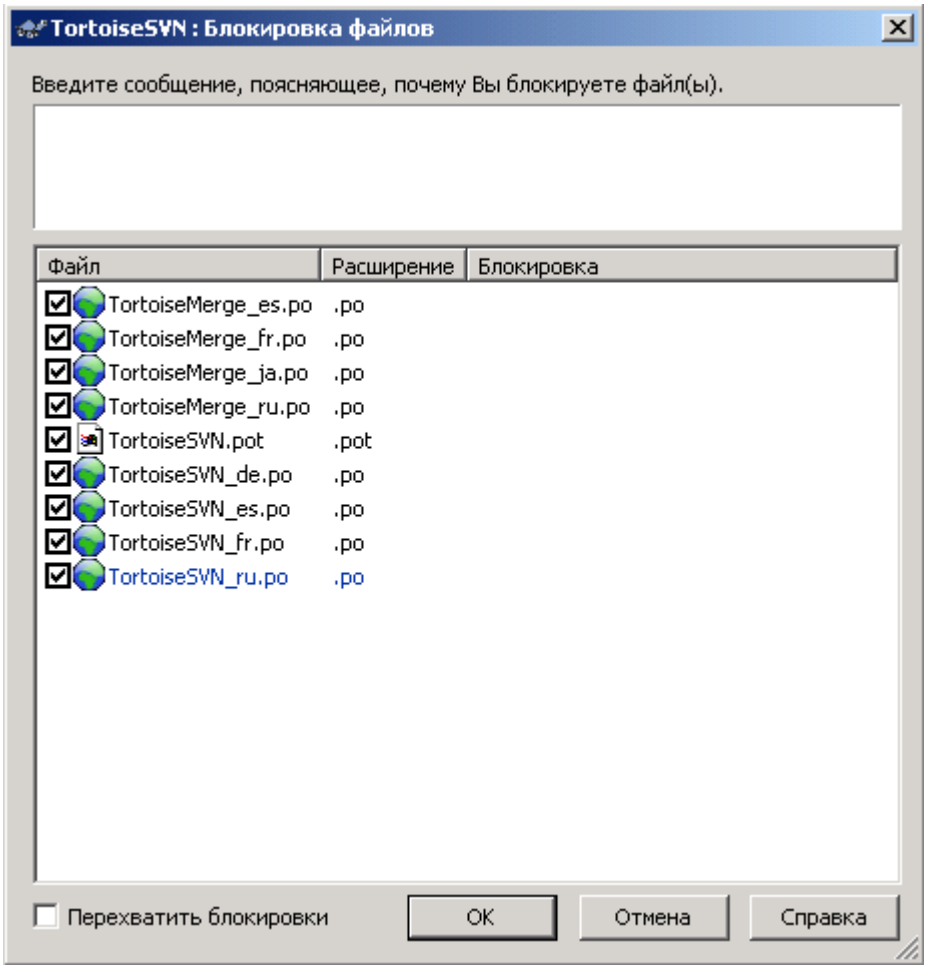


Рисунок 4.41. Диалог блокировки

Появится диалог, позволяющий ввести комментарий, чтобы другие могли увидеть, для чего вы заблокировали файлы. Комментарий необязателен, и сейчас используется только с хранилищами на базе Svnserve. Если (и *только* если) вам необходимо перехватить чужую блокировку, отметьте флажок Перехватить блокировку и нажмите OK.

Если вы выберите папку и затем выполните TortoiseSVN → Заблокировать..., диалог блокирования будет открываться для *каждого* файла в *каждой* подпапке, выбранной для блокирования. Если вы действительно хотите заблокировать всю иерархию файлов, это можно сделать и таким способом, но вы можете стать очень непопулярным среди ваших коллег, если вы заблокируете таким образом целый проект. Так что используйте с осторожностью...

4.21.3. Снятие блокировки

Для того, чтобы вы не забыли снять блокировки, которые вам больше не нужны, блокированные файлы отображаются в диалоге фиксации и они по умолчанию отмечены. Если продолжить выполнение фиксации, блокировки, которыми вы владеете для выбранных файлов, будут сняты, даже если файлы не были изменены. Если вы не желаете удалять блокировки некоторых файлов, вы можете их разотметить (если они не были изменены). Если вы желаете сохранить блокировку изменённого файла, то вам надо отметить флажок Сохранить блокировки перед фиксацией ваших изменений.

Для снятия блокировки вручную, выберите файл(ы) в вашей рабочей копии, с которых вы желаете снять блокировку, затем выполните команду TortoiseSVN → Снять блокировку. Больше ничего вводить не надо, TortoiseSVN свяжется с хранилищем и снимет блокировки. Вы можете также использовать эту команду на папке для рекурсивного снятия всех блокировок.

4.21.4. Проверка состояния блокировки

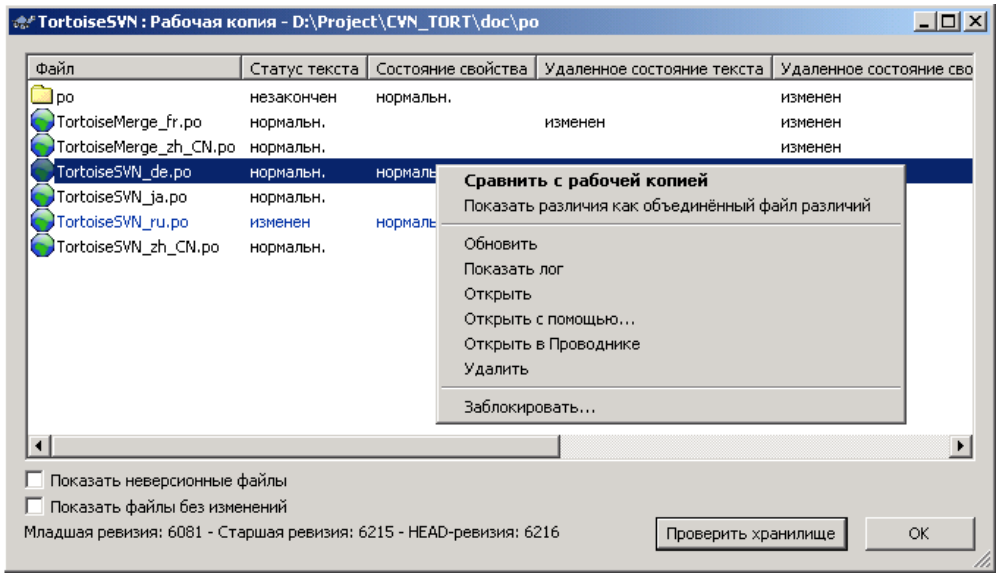


Рисунок 4.42. Диалог проверки на наличие изменений

Чтобы посмотреть, какие блокировки были установлены вами и другими разработчиками, воспользуйтесь TortoiseSVN → Проверить на наличие изменений. Установленные локально маркеры блокировки отображаются немедленно. Для проверки блокировок, установленных другими (а также для того, чтобы узнать, не были ли какие-либо из ваших блокировок прерваны или перехвачены), вам необходимо нажать на кнопку Проверить хранилище.

Здесь, используя контекстное меню, вы также можете устанавливать и снимать блокировки, помимо этого, можно также прерывать или перехватывать блокировки, установленные другими.

Избегайте прерывания и перехвата блокировок

Если вы прерываете или перехватываете чью-либо блокировку, не сообщая об этом, это потенциально может привести к потере выполненной работы. Если вы работаете с необъединяемыми типами файлов и перехватываете чью-либо блокировку, то, как только вы снимете блокировку, другие смогут зафиксировать свои изменения и перезаписать ваши. Subversion не теряет данные, но вы теряете предоставляемую блокировкой защиту при командной работе.

4.21.5. Незаблокированные файлы, доступные только-для-чтения

Как уже говорилось, наиболее эффективный путь использования блокировки - это установка у файлов свойства `svn:needs-lock`. Чтобы узнать о том, как устанавливать свойства, прочтите Раздел 4.17, «Установки проекта». Файлы, у которых установлено это свойство, при извлечении и обновлении получают признак "только-для-чтения", за исключением тех, блокировкой которых владеет ваша рабочая копия.

TortoiseSVN использует специальную пометку для обозначения этого в качестве напоминания.

Если у вас действует политика, при которой каждый файл должен быть заблокирован, то, возможно, будет легче применить возможность Subversion автоматически устанавливать свойства каждый раз, когда вы добавляете новые файлы (автосвойства). Для получения дополнительной информации прочтите Раздел 4.17.1.5, «Автоматическая установка свойств».

4.21.6. Скрипты ловушек на события блокировки

Когда вы создаёте новое хранилище при помощи Subversion версии 1.2 или более старшей, в папке хранилища `hooks` создаются четыре шаблона ловушек. Они вызываются перед и после получением блокировки, а также перед и после снятия блокировки.

Хорошей идеей является установить на сервере скрипты ловушек `после-блокировки` и `после-разблокировки`, которые будут отправлять электронное письмо с именем блокируемого файла. При наличии такого скрипта все ваши пользователи могут быть оповещены о том, что кто-либо блокирует/разблокирует файл. Вы можете найти пример скрипта ловушки `hooks/post-lock.tmpl` в папке вашего хранилища.

Вы также можете применить ловушки для запрета прерывания или перехвата блокировок, или, возможно, разрешить эти действия только администраторам. Или, может быть, вы захотите отправить письмо владельцу, когда одна из его блокировок прерывается или перехватывается.

Прочтите Раздел 3.3, «Скрипты ловушек, выполняемые на стороне сервера» для того, чтобы узнать об этом больше.

4.22. Создание и применение заплаток

В проектах с открытым исходным кодом (подобных этому) у всех есть права доступа для чтения в хранилище, и любой может внести свой вклад в проект. А как контролируются все эти вклады? Если каждый может вносить изменения, проект постоянно будет нестабильным и, возможно, постоянно неработоспособным. В этой ситуации управление изменениями осуществляется через отправку файла *заплатки* команде разработчиков, у которых есть доступ для записи. Они могут сначала отрецензировать заплатку, и затем или принять и отправить её в хранилище, или отклонить и вернуть автору обратно.

Файл заплатки - это просто файл объединённых различий, показывающий различия между вашей рабочей копией и базовой ревизией.

4.22.1. Создание файла заплатки

Сначала вы должны сделать *и проверить* ваши изменения. Затем, вместо использования на родительской папке TortoiseSVN → Фиксировать..., выберите TortoiseSVN → Создать заплатку...

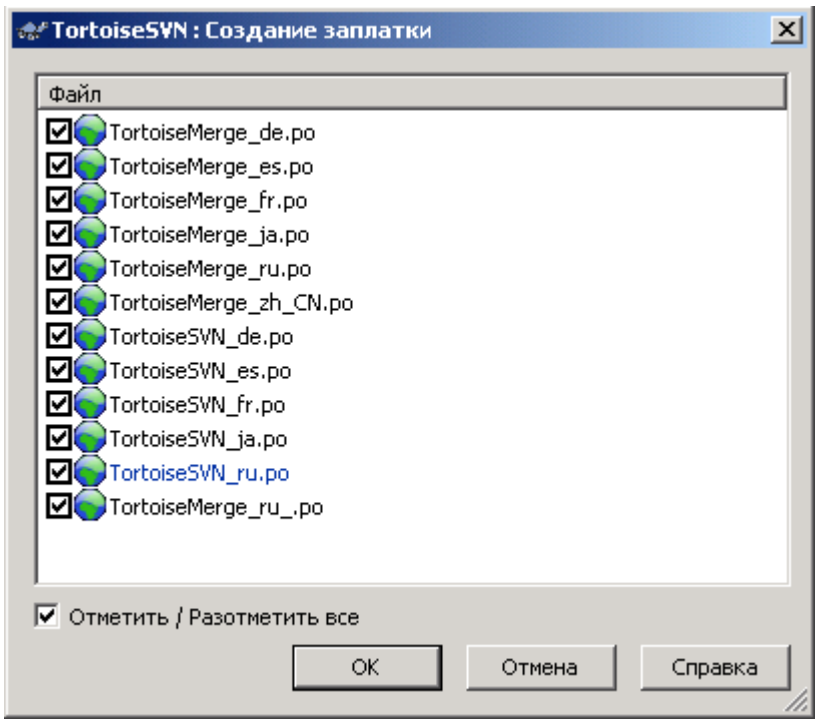


Рисунок 4.43. Диалог создания заплатки

теперь вы можете выбрать файлы, которые вы желаете включить в заплатку, точно также, как вы это делаете при полной фиксации. Будет создан один файл, содержащий сводку всех изменений, сделанных вами в выбранных файлах с момента последнего обновления из хранилища.

Столбцы в этом диалоге могут настраиваться таким же образом, как и столбцы в диалоге Проверка на наличие изменений. Прочтите Раздел 4.7.3, «Локальный и удалённый статус» если вам необходима дополнительная информация.

Вы можете создать несколько заплаток, содержащих изменения в различных наборах файлов. Конечно, если вы создадите файл заплатки, произведёте ещё какие-либо изменения в *тех же* файлах, и после этого создадите другую заплатку, этот второй файл заплатки будет включать *оба* набора изменений.

Просто сохраните файл, назвав его по собственному выбору. У файлов заплаток может быть любое понравившееся вам расширение, но по соглашению должно использоваться расширение `.patch` или `.diff`. Теперь вы готовы отправить ваш файл заплатки.

Вы можете также сохранить заплатку в буфер обмена вместо файла. Это может понадобиться для того, чтобы вставить её в сообщение электропочты для отправки на рецензирование. Или, если у вас есть две рабочие копии на одном компьютере, и вы желаете перенести изменения из одной в другую, заплатка в буфере обмена - удобный способ это сделать.

4.22.2. Применение файла заплатки

Файлы заплаток применяются к вашей рабочей копии. Применение должно производиться на том же уровне папок, который был использован для создания заплатки. Если вы этот уровень не знаете точно, просто посмотрите на первую строку файла заплатки. Например, если первый обрабатываемый файл был `doc/source/english/chapter1.xml` и первая строка в файле заплатки выглядит как `Index: english/chapter1.xml`, то вам необходимо применить заплатку к папке `doc/source/`. Однако, в том случае, если вы пытаетесь применить заплатку в надлежащей рабочей копии, и вы указали неверный уровень папки, TortoiseSVN это заметит и предложит правильный уровень.

Для того, чтобы применить файл заплатки к вашей рабочей копии, вы должны иметь как минимум доступ для чтения в хранилище. Причина этого в том, что программа слияния должна соотнести изменения с той прошлой ревизией, относительно которой они были сделаны удалённым разработчиком.

Из контекстного меню этой папки выберите TortoiseSVN → Применить заплатку... Появится диалог открытия файла, позволяющий выбрать файл заплатки для применения. По умолчанию отображаются только файлы с расширением `.patch` или `.diff`, но вы можете выбрать для показа и «Все файлы». Если вы до этого сохранили заплатку в буфере обмена, можно воспользоваться кнопкой Открыть из буфера обмена в диалоге открытия файла.

Или, если файл заплатки имеет расширение `.patch` или `.diff`, вы можете щёлкнуть на нём правой клавишей мыши и выбрать TortoiseSVN → Применить заплатку.... В этом случае у вас будет запрошено расположение рабочей копии.

Эти два метода - просто два различных способа сделать одно и то же. В первом методе вы выбираете рабочую копию и указываете файл заплатки, во втором - выбираете файл заплатки и указываете рабочую копию.

Как только вы выбрали файл заплатки и расположение рабочей копии, запускается TortoiseMerge для слияния изменений из файла заплатки с вашей рабочей копией. В небольшом окне перечислены файлы, которые были изменены. Выполните двойной щелчок на каждом файле по очереди, просмотрите изменения, и сохраните слитые файлы.

Теперь, когда заплатка удалённого разработчика была применена к вашей рабочей копии, вам надо зафиксировать результат, чтобы все остальные смогли получить эти изменения из хранилища.

4.23. Кто какую строку изменил?

Иногда вам необходимо узнать не только какие из строк изменились, но также и то, кто именно изменил определённые строки в файле. И в этом случае может пригодиться команда TortoiseSVN → Авторство..., иногда называемая также *аннотирование*.

Эта команда выводит для каждой строки в файле её автора и ревизию, в которой она была изменена.

4.23.1. Авторство для файлов

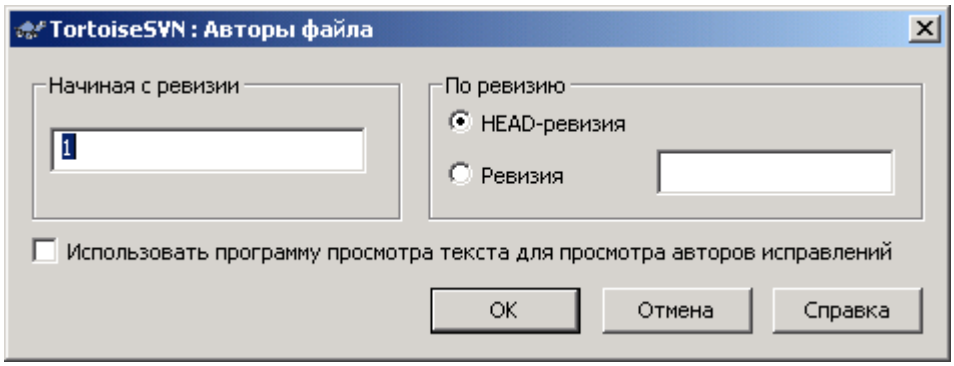


Рисунок 4.44. Диалог авторства/аннотирования

Если вас не интересуют изменения из ранних ревизий, вы можете указать ревизию, начиная с которой будет выполняться команда. Задайте значение `1`, если вы желаете получить авторство *всех* ревизий.

По умолчанию файл авторства показывается при помощи *TortoiseBlame*, который подсвечивает различные ревизии для упрощения чтения. Если вы желаете напечатать или изменить файл авторства, выберите Использовать программу просмотра текста для показа авторства

Вы можете указать, каким способом будут обрабатываться изменения завершений строк и пробельных символов. Эти возможности описывает Раздел 4.10.2, «Параметры сравнения завершений строк и непечатаемых знаков». Поведение по умолчанию - считать все различия в пробельных символах и завершениях реальными изменениями, но если вы желаете проигнорировать изменение отступа и найти первоначального автора, вы можете выбрать здесь соответствующую возможность.

После того, как вы нажмёте OK, TortoiseSVN начнёт извлекать данные для создания файла авторства. Обратите внимание: это может занять несколько минут, в зависимости от того, насколько большая часть файла была изменена и, конечно же, от производительности вашего сетевого соединения с хранилищем. После того, как процесс получения информации об авторстве завершится, результат записывается во временный файл и вы можете его просмотреть.

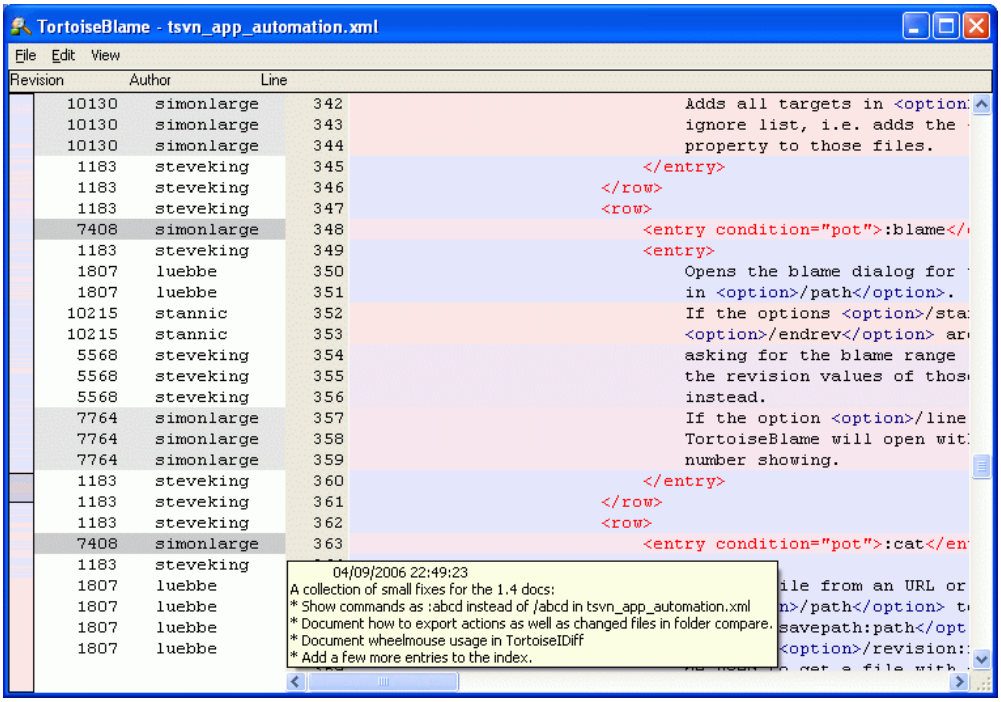


Рисунок 4.45. TortoiseBlame

TortoiseBlame, входящий в TortoiseSVN, упрощает чтение файла авторства. При наведении указателя мыши на строку в столбце информации об авторстве все строки из той же ревизии отображаются с затемнённым фоном. У строк из других ревизий, изменённых тем же автором, тоже изменяеся фон, но оттенок фона более светлый, чем у ревизии под указателем. Цветовая подсветка может не работать так хорошо, если ваш дисплей работает в режиме отображения 256 цветов.

Если вы сделаете левый щелчок на строке, будут подсвечены все строки из той же ревизии, а строки из других ревизий того же автора будут подсвечены ещё более тёмным цветом. Это подсвечивание постоянное, оно позволяет перемещать мышку без потери подсвечивания. Щёлкните на этой же ревизии ещё раз для выключения подсветки.

Комментарии к ревизии (сообщения журнала) показываются во всплывающей подсказке всякий раз при наведении указателя мыши на колонку информации об авторстве. Если вы желаете скопировать сообщение журнала этой ревизии, используйте контекстное меню, появляющееся при правом щелчке в этой же колонке.

Вы можете производить поиск в отчёте об авторстве при помощи Правка → Найти.... Это позволяет искать в номерах ревизий, авторах и в содержимом файла. Сообщения журнала в область поиска не включены - для поиска в них вы должны использовать диалог журнала.

Вы также можете перейти к строке с нужным номером при помощи Правка → Перейти к строке....

Когда указатель мыши находится над колонкой информации об авторстве, доступно контекстное меню, при помощи которого можно сравнить ревизии и изучить историю, используя номер ревизии строки под указателем в качестве опорного. Команда Контекстное меню → Авторство предыдущей ревизии создаёт отчёт об авторстве для того же файла, но использует в качестве верхнего предела предыдущую ревизию. Это даёт возможность получить отчёт об авторстве для файла, каким он был перед тем, когда в последний раз была изменена нужная вам строка. Команда Контекстное меню → Показать изменения запускает программу просмотра различий, для показа того, что было изменено в опорной ревизии. Команда Контекстное меню → Журнал служит для отображения диалога журнала ревизий, начиная с опорной ревизии.

Если вам нужен более наглядный индикатор того, где изменения более новые, а где более старые, отметьте опцию Вид → Обозначать цветом возраст строк. После этого для показа возраста строк будет использован цветовой градиент: более новые строки будут иметь красный оттенок, более старые - синий. Цвет, используемый по умолчанию, довольно светлый, но вы можете изменить его в настройках TortoiseBlame.

При использовании отслеживания слияний, там, где строки были изменены в результате слияния из другого пути, TortoiseBlame покажет ревизию и автора последнего изменения в первоначальном файле, а не ревизию, в которой произошло слияние. Признаком таких строк служат отображаемые курсивом ревизия и автор. Если вы не желаете, чтобы слитые строки выделялись таким образом, снимите отметку с флажка Включая данные о слияниях.

Если вы желаете видеть пути, вовлечённые в слияние, отметьте опцию Вид → Слитые пути.

До настроек TortoiseBlame можно добраться, используя TortoiseSVN → Настройки... на вкладке TortoiseBlame. Подробнее об этом рассказывает Раздел 4.30.9, «Настройки TortoiseBlame».

4.23.2. Авторство различий

Одним из ограничений отчёта об авторстве является то, что он показывает файл в том виде, в каком он был в конкретной ревизии, и показывает для каждой строки последнего изменившего её человека. Иногда же бывает необходимо узнать не только то, кто это сделал, но и какие изменения были произведены. В данной ситуации вам необходимо совмещение выдачи по различиям и по авторству.

Диалог журнала ревизий предоставляет несколько возможностей, которые помогают вам это сделать:

Авторство ревизий

В верхней панели выберите 2 ревизии и выполните Контекстное меню → Авторство ревизий. При этом будут извлечены данные об авторстве для этих двух ревизий, после чего будет вызвана программа просмотра различий для сравнения двух получившихся файлов, содержащих информацию об авторстве.

Авторство изменений

Отметьте одну ревизию в верхней панели, потом выберите один файл в нижней панели и выполните команду Контекстное меню → Авторство изменений. При этом будут извлечены данные об авторстве для выбранной и предыдущей ревизии, после чего будет вызвана программа просмотра различий для сравнения двух получившихся файлов с информацией об авторстве.

Сравнить с рабочей базой вместе с просмотром авторства

Вызовите журнал для одного файла и в верхней панели выберите одну ревизию, после чего выполните Контекстное меню → Сравнить с рабочей базой с просмотром авторства.

При этом будут извлечены данные об авторстве для выбранной ревизии, а также для файла в рабочей базе, после чего будет вызвана программа просмотра различий для сравнения двух получившихся файлов с информацией об авторстве.

4.24. Обозреватель хранилища

Иногда бывает необходимо поработать непосредственно с хранилищем, без наличия рабочей копии. Именно для этого и предназначен *обозреватель хранилища*. Подобно тому, как Проводник и пометки на значках позволяют просматривать рабочую копию, так и обозреватель хранилища предоставляет возможность просмотреть структуру и состояние хранилища.

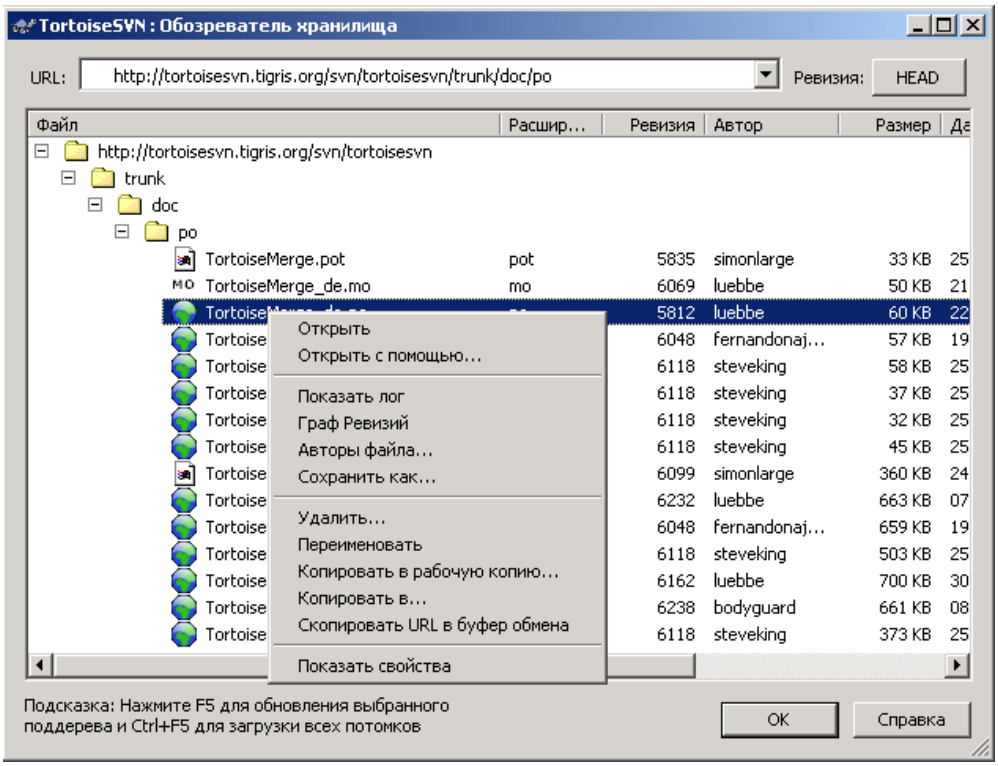


Рисунок 4.46. Обозреватель хранилища

При помощи обозревателя хранилища вы можете выполнять такие команды, как копирование, перемещение, переименование и т.д. прямо в хранилище.

Обозреватель хранилища выглядит во многом также, как и Проводник Windows, за исключением того, что он показывает содержимое хранилища для конкретной ревизии, а не файлы на вашем компьютере. В левой панели находится дерево папок, а в правой - содержимое выбранной папки. В верхней части окна обозревателя хранилища можно ввести URL хранилища и ревизию, которую вы желаете просмотреть.

Также, как и в Проводнике Windows, вы можете щёлкнуть на заголовке колонки в правой панели, если вы желаете задать порядок сортировки. И также как в Проводнике, в обоих панелях доступны контекстные меню.

При помощи контекстного меню для файла можно сделать следующее:

- Открывает выбранный файл либо в программе просмотра по умолчанию для этого типа файлов, либо в другой выбранной вами программе.
- Сохранить неверсированную копию файла на жёсткий диск.
- Показать журнал ревизий для этого файла, или показать граф всех ревизий, чтобы можно было посмотреть всю историю этого файла.

- Получить информацию об авторстве для файла, чтобы посмотреть, кто какую строку изменил и когда.
- Удалить или переименовать файл.
- Скопировать файл, либо в другую часть хранилища, либо в рабочую копию, базирующуюся в том же хранилище.
- Посмотреть/отредактировать свойства файла.

При помощи контекстного меню для папки можно сделать следующее:

- Показать журнал ревизий для этой папки, или показать граф всех ревизий, чтобы можно было посмотреть всю историю этой папки.
- Экспортировать папку в локальную неверсированную копию на жестком диске.
- Извлечь папку для создания локальной рабочей копии на жестком диске.
- Создать новую папку в хранилище.
- Добавить файлы или папки прямо в хранилище.
- Удалить или переименовать папку.
- Скопировать папку, либо в другую часть хранилища, либо в рабочую копию, базирующуюся в том же хранилище.
- Посмотреть/отредактировать свойства папки.
- Отметить папку для сравнения. Отмеченная папка показывается жирным шрифтом.
- Сравнить папку с предыдущей отмеченной папкой, либо в виде объединённых различий, либо в виде списка изменённых файлов, которые после этого можно наглядно сравнить при помощи используемой по умолчанию программы просмотра различий. Это особенно полезно для сравнения двух меток или же для ствола с ответвлением, чтобы увидеть, что изменилось.

Если выбрать две папки в правой панели, то можно посмотреть различия, либо в виде объединённых различий, либо в виде списка изменённых файлов, которые можно сравнить наглядно при помощи используемой по умолчанию программы просмотра различий.

Если выбрать несколько папок в правой панели, то можно извлечь их все за один приём в общую родительскую папку.

Если выбрать две метки, которые были скопированы из одного корня (обычно `/trunk/`), то при помощи Контекстное меню → Журнал... можно просмотреть список ревизий между двумя отмеченными точками.

Как обычно, вы можете использовать **F5** для обновления, при этом будет обновлено всё, что отображается. Если вы желаете заранее получить или обновить информацию для узлов, которые пока не отображаются, используйте **Ctrl-F5**. После этого раскрытие любого узла будет происходить немедленно, без задержки на передачу данных по сети.

Вы также можете использовать обозреватель хранилища для операций перетаскивания. Если вы перетащите папку из Проводника в обозреватель хранилища, она будет импортирована в хранилище. Обратите внимание: при перетаскивании нескольких элементов они будут импортированы отдельными фиксациями.

Если вы желаете переместить элемент в рамках хранилища, просто используйте левое перетаскивание его на новое место. Если вы желаете скопировать этот элемент, а не переместить, то используйте вместо этого **Ctrl**+левое перетаскивание. При копировании у курсора появляется символ «плюс», также как и в Проводнике.

Если вы желаете скопировать/переместить файл или папку в другое место, присвоив при этом также новое имя, вы можете применить правое перетаскивание или **Ctrl**-правое перетаскивание элемента вместо обычного левого перетаскивания. В этом случае показывается диалог переименования, где вы можете ввести новое имя для файла или папки.

Всякий раз, когда вы производите изменения в хранилище одним из этих способов, вам предлагается ввести сообщение журнала. Если вы перетащили что-то по ошибке, то это также ваш шанс отменить операцию.

Иногда, при попытке открыть какой-нибудь путь, вы можете получить сообщение об ошибке вместо деталей описания элемента. Это может случиться, если вы указали неправильный URL, или у вас нет достаточных прав на доступ к хранилищу, или из-за наличия какой-нибудь другой проблемы на сервере. Если вам необходимо скопировать это сообщение для включения в сообщение электронной почты, щёлкните на нём правой клавишей и выберите Контекстное меню → Скопировать сообщение об ошибке в буфер обмена, или же просто воспользуйтесь **Ctrl+C**.

4.25. Графы ревизий

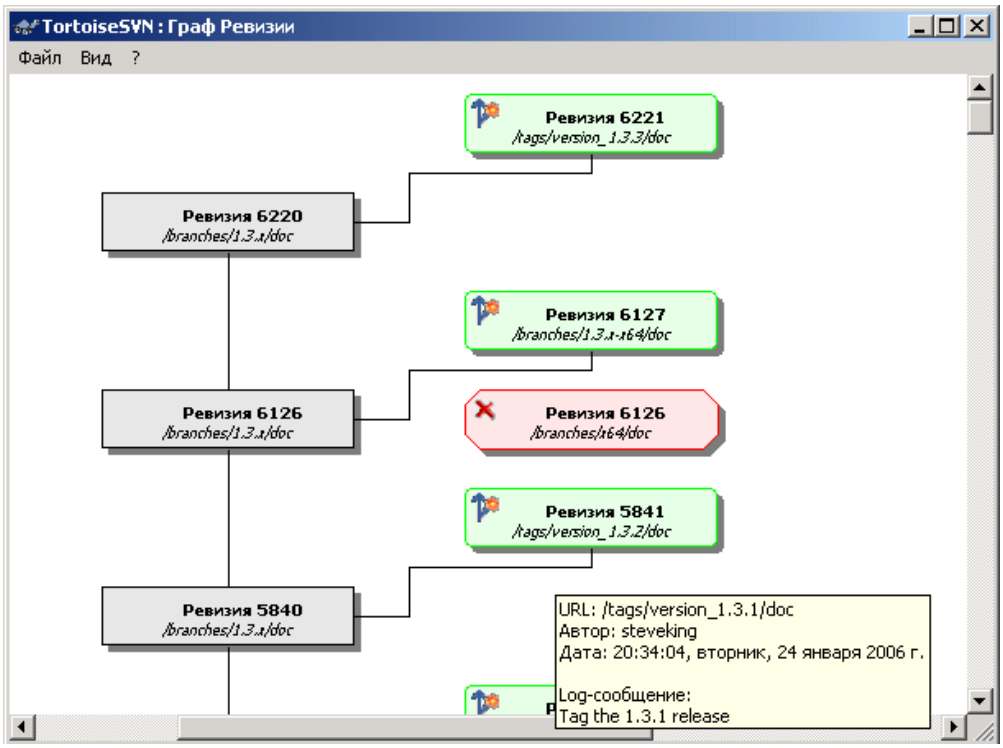


Рисунок 4.47. Граф ревизий

Иногда вам бывает необходимо узнать, из какого места ствола были созданы ответвления и метки, и идеальный способ просмотра этого типа информации - граф или структура в виде дерева. Именно в этой ситуации применяется TortoiseSVN → Граф ревизий

Эта команда анализирует историю ревизий и пытается создать дерево, отображающее точки, в которых были сделаны копии, и где эти ответвления/метки были удалены.

Важно

Для того, чтобы сформировать граф, TortoiseSVN должен извлечь все сообщения журнала из корня хранилища. Не стоит говорить, что это может занять несколько

минут, даже когда в хранилище находится всего несколько тысяч ревизий, и зависит от скорости сервера, пропускной способности сети и т.п. Если вы попробуете это на проекте вроде `Apache`, который сейчас имеет более 500,000 ревизий, вам придётся подождать некоторое время.

Хорошая новость заключается в том, что при использовании кэша сообщений журнала вам придётся подождать единожды. После этого данные журнала хранятся локально. Кэширование сообщений журнала включается в настройках TortoiseSVN.

4.25.1. Узлы графа ревизий

Каждый узел графа ревизий олицетворяет ревизию в хранилище, которая что-либо изменила в отображаемом дереве. Разнотипные узлы различаются формой и цветом. Форма не может быть изменена, а цвет можно задать, используя TortoiseSVN → Настройки

Добавленные или скопированные узлы
: Элементы, которые были добавлены, или созданы путём копирования другого файла/папки, показываются в виде прямоугольника с закругленными углами. Цвет по умолчанию зелёный. Метки и стволы рассматриваются как отдельный случай и для них используется другой оттенок, в зависимости от того, что указывают TortoiseSVN → Настройки.

Удалённые узлы
: Удалённые элементы, такие как не нужные больше ответвления, показываются в виде восьмиугольника (прямоугольник с отрезанными углами). Цвет по умолчанию красный.

Переименованные узлы
: Переименнованые элементы также показываются в виде восьмиугольника, но цвет по умолчанию синий.

Верхние ревизии ответвлений
: Граф обычно показывает только точки ответвлений, но часто бывает полезно увидеть ещё и соответствующую ведущую ревизию для каждого ответвления. Если выбрать Показать ведущие ревизии, то будет показан каждый узел ведущей ревизии (в эллипсе). Заметьте, что здесь ведущая ревизия имеет смысл последней ревизии, зафиксированной по этому пути, а не ведущей ревизии хранилища.

Ревизия рабочей копии
: Если вы вызвали граф ревизий из рабочей копии, то у вас есть возможность показать базовую ревизию на графе при помощи Показать ревизию рабочей копии, обводящее базовый узел толстой рамочкой.

Изменённая рабочая копия
: Если вы вызвали граф ревизий из рабочей копии, вы можете показать дополнительный узел, представляющий вашу изменённую рабочую копию, при помощи Показать изменения рабочей копии. Это по умолчанию красный узел в форме эллипса с толстой рамочкой.

Обычный элемент
: Все остальные элементы отображаются в виде обычного прямоугольника.

Обратите внимание: по умолчанию граф показывает только те точки, в которых элементы были добавлены, скопированы или удалены. Отображение каждой ревизии проекта породит слишком большой граф для нетривиальных случаев. Если вы действительно желаете увидеть *все* ревизии, в которых были произведены изменения, то для этого есть специальная опция, расположенная в меню Вид и на панели инструментов.

Вид по умолчанию (группировка выключена) размещает узлы так, чтобы их положение по вертикали было в строгом соответствии с порядком ревизий, и у вас было наглядное представление о том, в какой последовательности что было сделано. Там, где два узла расположены в одной колонке, порядок очевиден. Когда два узла расположены в смежных колонках, смещение довольно

мало, поскольку нет необходимости предотвращать перекрытие узлов, и в результате порядок немного менее очевиден. Такого рода оптимизации необходимы, чтобы удержать сложные графы в приемлемых размерах. Имейте ввиду, что размещение по порядку использует *край* узла с более *старой* стороны как точку отсчёта, т.е. нижний край узла, когда граф отображается с более старыми узлами снизу. Край, от которого производится отсчёт, важен, так как формы узлов не все одинаковой высоты.

4.25.2. Изменение вида

Поскольку граф ревизий часто получается довольно сложным, есть несколько возможностей, которые могут быть использованы для донастройки способа отображения графа под ваши нужды. Они доступны в меню Вид и в панели инструментов.

Сгруппировать ответвления
: При поведении по умолчанию (группировка выключена) все строки сортируются строго по ревизии. В результате долгоживущие ответвления с редкими фиксациями занимают целую колонку всего лишь для нескольких изменений и граф получается слишком широким.

  Этот режим группирует изменения по ответвлению, поэтому глобального упорядочивания ревизий не будет: последовательные ревизии в ответвлении будут отображаться в (часто) последовательных строках. Под-ответвления, однако, упорядочиваются таким образом, чтобы более поздние ответвления показывались в той же самой колонке, выше более раннего ответвления, и граф не разрастался в ширину. В результате, некоторые строки могут содержать изменения из различных ревизий.

Старые сверху
: Обычно в графе более старые ревизии показываются снизу, и дерево растёт вверх. При помощи этой опции можно указать, чтобы дерево росло наоборот, сверху вниз.

Выровнять деревья поверху
: Когда граф разбит на несколько меньших деревьев, деревья могут отображаться или в естественном порядке ревизий, или выровненными по нижнему краю окна, в зависимости от того, использовали ли вы опцию Сгруппировать по ответвлениям. А эту опцию применяйте, чтобы все деревья росли наооборот, сверху вниз.

Снизить число пересечений
: Если при отображении графа получается слишком много пересекающихся линий, применяйте эту опцию для наведения порядка. Это может привести к расположению колонок для размещения в менее логичных местах, например, по диагонали, а не в столбик, и может потребоваться больше места для отображения графа.

Различающиеся части путей
: Длинные имена путей могут занять много места и сделать блоки узлов очень большими. Используйте эту опцию, чтобы отображались только изменённые части пути (общая часть пути будет заменена точками). Например, если вы создали ответвление `/branches/1.2.x/doc/html` из `/trunk/doc/html` ответвление может быть показано в компактном виде как `/branches/1.2.x/..` поскольку последние два уровня, `doc` и `html`, не изменились.

Показать все ревизии
: Это делает именно то, что вы ожидаете и показывает каждую ревизию, в которой что-либо (в дереве, граф которого вы строите) было изменено. Для проектов с длинной историей это может порождать действительно громадный граф.

Показать ведущие ревизии
: Эта опция обеспечивает отображение в графе самой поздней ревизии каждого ответвления.

Точные источники копирования
: Поведение по умолчанию при создании ответвления/метки - показывать ответвление как созданное из последнего узла, где было произведено изменение. Строго говоря, это неточно, поскольку ответвления часто создаются из текущей ведущей ревизии, а не из какой-то

конкретной ревизии. Поэтому есть возможность показывать более правильную (но менее полезную) ревизию, которая использовалась для создания копии. Заметьте, что эта ревизия может быть моложе, чем ведущая ревизия исходного ответвления.

Свернуть метки
: Если в проекте много меток, отображение каждой метки как отдельного узла на графе занимает много места и делает неясной более интересующую структуру ответвления разработки. В то же время, вам может понадобится простой доступ к содержимому меток, чтобы вы могли сравнивать ревизии. Эта опция скрывает узлы меток и вместо этого показывает их во всплывающей подсказке для узла, из которого они были скопированы. Для обозначения того, что из узла была создана метка, справа на узле-источнике показывается значок метки.

Скрыть удалённые пути
: Скрывает пути, которых больше нет в ведущей ревизии хранилища, например, удалённые ответвления.

Скрыть неиспользуемые ответвления
: Скрывает ответвления, в которых не было зафиксировано изменений в соответсвующем файле или подпапке. Это необязательно показывает, что ответвление не использовалось, это показывает только то, что не было изменений в *этой* его части.

Показать ревизию рабочей копии
: Выделяет ревизию в графе, соответствующую ревизии обновления элемента, для которого вы строите граф. Если вы только что обновились, это будет ведущая ревизия, но если другие фиксировали изменения с момента, когда вы последний раз обновляли рабочую копию, то она может быть несколькими ревизиями ниже. Узел выделяется толстой рамочкой.

Показать изменения рабочей копии
: Если в вашей рабочей копии есть локальные изменения, эта опция нарисует их как отдельный эллиптический узел, связанный с узлом, до которого ваша рабочая копия была последний раз обновлена. Цвет рамки по умолчанию красный. Возможно, вам потребуется обновить граф при помощи **F5** для учёта последних изменений.

Фильтр
: Иногда в графе содержится больше ревизий, нежели вам необходимо просмотреть. Эта опция в открывающемся диалоговом окне позволяет ограничить диапазон показываемых ревизий, а также скрыть некоторые указанные по имени пути.

Деревья в полосах
: Когда граф содержит неколько деревьев, бывает полезно использовать в фоне чередующиеся цвета, чтобы было проще понять, что к какому дереву относится.

Показать обзорное окно
: Показывает небольшое изображение всего графа с текущим отображаемым окном в виде прямоугольника, который можно перемещать. Это позволяет передвигаться по графу намного легче. Обратите внимание: для очень больших графов обзорное окно может стать безполезным из-за чрезмерной степени увеличения и поэтому в этом случае оно показано не будет.

4.25.3. Использование графа

Для того, чтобы легче ориентироваться в большом графе, можно использовать обзорное окно. Оно показывает весь граф в небольшом окне, и текущая показываемая часть в нём выделена. Вы можете перетаскивать зону выделения для изменения отображаемой области.

При прохождении мыши над прямоугольником ревизии во всплывающей подсказке отображаются дата ревизии, автор и сообщение журнала.

Если вы выберете две ревизии (используя **Ctrl**-левый щелчок), вы можете воспользоваться контекстным меню для просмотра различий между этими ревизиями. Конечно, вы можете выбрать

просмотр различий в точках создания ответвлений, но обычно бывает желательно просмотреть различия в конечных точках ответвлений, т.е. в ведущей ревизии.

Вы можете просмотреть различия как файл объединённых различий, который показывает все различия в одном файле с некоторым минимальным контекстом. Если выбрать Контекстное меню → Сравнить ревизии, появится список измененных файлов. Выполните двойной щелчок на имени файла для извлечения обеих ревизий файла и их сравнения с использованием визуального средства просмотра различий.

После правого щелчка на ревизии вы можете выбрать Контекстное меню → Журнал для просмотра истории.

Вы можете также произвести слияние изменений из выбранных ревизий с другой рабочей копией. При помощи диалога выбора папки можно выбрать рабочую копию в котрой будет проводиться слияние, но после этого не предоставляется ни запроса подтверждения, ни возможности выполнить пробный запуск. Хорошей практикой является производить слияние с неизменённой рабочей копией, чтобы вы смогли отменить изменения, если слияние не сработает как надо! Это полезная возможность, если вы желаете слить выбранные ревизии из одного ответвления в другое.

Учимся читать граф ревизий

Начинающие пользователи могут быть удивлены тем, что граф ревизий отображает нечто, не соответствующее мысленной модели пользователя. Например, если ревизия изменяет несколько копий или ответвлений файла или папки, то будет несколько узлов для этой единственной ревизии. Хорошей привычкой будет начать с самых левых опций в панели инструментов и настраивать граф шаг за шагом, пока он не будет близок к вашей мысленной модели.

Все опции фильтров стараются терять настолько мало информации, насколько это возможно. Это может привести к тому, что некоторые узлы поменяют свой цвет, например. Всякий раз, когда результат оказывается неожиданным, отмените последнее применение фильтра и попробуйте понять, что такого особенного в данной ревизии или ответвлении. В большинстве случаев, изначально ожидаемый результат применения фильтра будет или неточным, или обманчивым.

4.25.4. Обновление вида

Если вы желаете вновь запросить сервер на предмет новой информации, вы можете просто обновить вид при помощи **F5**. Если используется кэширование журнала (по умолчанию включено), то хранилище будет проверено на наличие более новых фиксаций и будут загружены только они. Если кэш журнала работает в автономном режиме, то будет произведена попытка переключиться обратно в оперативный режим.

Если вы используете кэширование журнала, и вы думаете, что содержимое сообщения или его автор были изменены, вы должны воспользоваться диалогом журнала для обновления необходимых сообщений. Поскольку граф ревизий работает, начиная с корня хранилища, будет сделан недействительным весь кэш журнала, и повторное его заполнение может занять *очень* долгое время.

4.25.5. Подрезка деревьев

В большом дереве может быть трудно сориентироваться, и иногда бывает необходимо скрыть его часть, или разбить его на лес более маленьких деревьев. Если навести мышь на точку, где соединительная линия входит или выходит из узла, то появится одна или несколько кнопок, которые позволяют это сделать.

Щёлкните на кнопке с минусом для сворачивания присоединённого поддерева.

Щёлкните на кнопке с плюсом для разворачивания свёрнутого поддерева. Когда дерево свёрнуто, эта кнопка остаётся видимой, чтобы показать наличие скрытого поддерева.

Щёлкните на кнопке с крестом для отделения присоединённого поддерева и отображения его как отдельного дерева в графе.

Щёлкните на кнопке с кругом, чтобы присоединить отделённое дерево. Когда дерево отделено, эта кнопка остаётся видимой, чтобы показать наличие отдельного поддерева.

Щёлкните на фоне графа для вызова главного контекстного меню, предлагающего опции Развернуть все и Соединить все. Если ещё ни одно ответвление не было свёрнуто или отделено, контестное меню не показывается.

4.26. Экспорт рабочей копии Subversion

Иногда вы желаете скопировать ваше рабочее дерево без всех этих папок `.svn`, например, для создания файла архива ваших исходных кодов, или для экспорта на веб-сервер. Вместо того, чтобы выполнить копирование и после этого удалять папки `.svn` вручную, TortoiseSVN предлагает команду TortoiseSVN → Экспорт.... Экспорт из источника, заданного адресом URL, и экспорт из рабочей копии, обрабатывается немного по-разному.

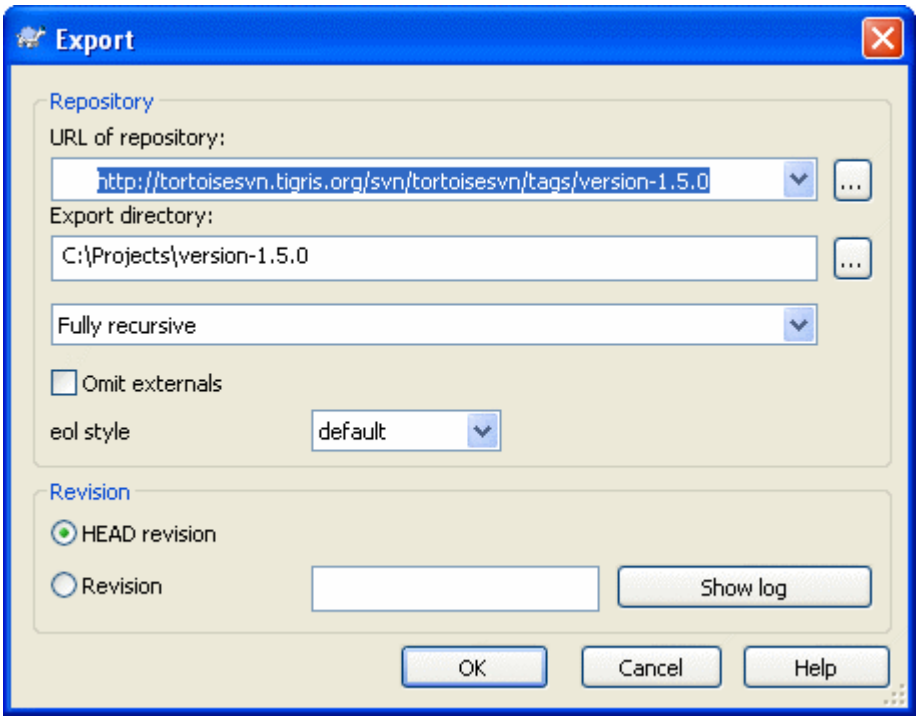


Рисунок 4.48. Диалог Экспорт-из-URL

Если вы выполняете эту команду на неверсированной папке, TorotiseSVN предполагает, что выбранная папка является целевой, и открывает диалог для ввода URL и ревизии, из которых необходимо произвести экспорт. В этом диалоге присутствуют опции, при помощи которых можно экспортировать только папку верхнего уровня, пропустить внешние ссылки, и задать тип завершения строк для файлов, у которых установлено свойство `svn:eol-style`.

Конечно же, вы также можете экспортировать прямо из хранилища. Воспользуйтесь обозревателем хранилища для перехода к соответствующему поддереву в хранилище, после чего выберите Контекстное меню → Экспорт. Вы получите описанный выше диалог Экспорт из URL.

Если выполнить эту команду на рабочей копии, вас попросят указать место для сохранения *чистой* рабочей копии без папок `.svn`. По умолчанию, экспортируются только версированные файлы, но вы можете при помощи флажка Экспортировать также и неверсированные файлы включить также все неверсированные файлы, существующие в вашей рабочей копии и не существующие в хранилище. Внешние ссылки, заданные через `svn:externals`, могут быть опущены при необходимости.

Другой способ сделать экспорт из рабочей копии - правое перетаскивание папки с рабочей копией в новое место путём выбора Контекстное меню → SVN Экспортировать сюда или Контекстное меню → SVN Экспортировать всё сюда. Второй пункт включает также и неверсированные файлы.

Если целевая папка при экспорте из рабочей копии уже содержит папку с таким же именем, как и экспортируемая, вам будет дана возможность выбора: перезаписать ли существующее содержимое, или же создать новую паку с автоматически генерируемым именем, например, `Целевая папка (1)`.

Экспортирование отдельных файлов

Диалог экспорта не позволяет экспортировать отдельные файлы, немотря на то, что Subversion это может.

Для экспорта отдельных файлов в TortoiseSVN вы должны использовать обозреватель хранилища (Раздел 4.24, «Обозреватель хранилища»). Просто перетащите файл (файлы), которые вы желали бы экспортировать, из обозревателя хранилища в Проводник, поместив их туда, куда надо, или воспользуйтесь для экспорта файлов контекстным меню обозревателя хранилища.

Экспортирование дерева изменений

Если вы желаете экспортировать копию структуры вашего проекта, содержащую только файлы, изменённые или в определённой ревизии, или между любыми двумя ревизиями, воспользуйтесь возможностью сравнения ревизий, описание которой содержит Раздел 4.10.3, «Сравнение папок».

4.26.1. Выведение рабочей копии из-под управления версиями

Иногда возникает задача преобразовать рабочую копию обратно в нормальную папку без папок `.svn`. Что вам действительно для этого нужно - это команда экспорта-на-месте, которая только убирала бы управляющие папки, нежели генерировала новое чистое дерево папок.

Решение неожиданно простое - экспорт папки в саму себя! TortoiseSVN обнаруживает этот специальный случай и спрашивает, не желаете ли вы разверсировать рабочую копию. При ответе *да* управляющие папки будут удалены, и у вас будет простое неверсированное дерево папок.

4.27. Перебазирование рабочей копии

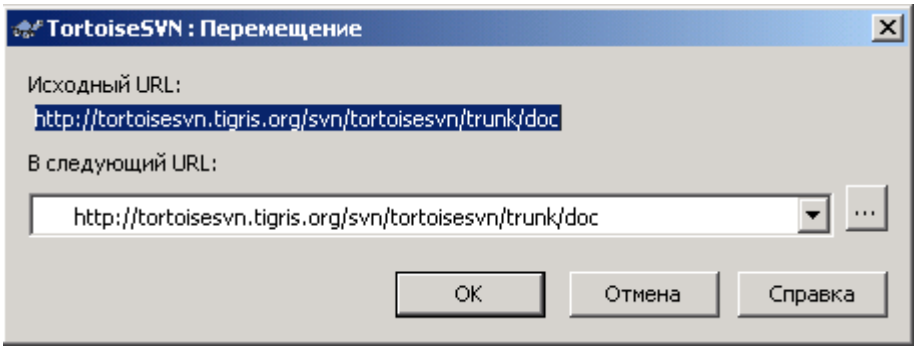


Рисунок 4.49. Диалог перебазирования

Допустим, ваше хранилище по каким-либо причинам изменило свое расположение (IP/URL). Возможно, вся ваша работа застопорилась и вы не можете фиксировать изменения, а вам не хочется заново извлекать рабочую копию из нового месторасположения и перемещать в неё все ваши изменённые данные. В этом случае команда TortoiseSVN → Перебазировать... - это то, что вам нужно. По существу, она делает очень немного: сканирует все файлы `entries` в папках `.svn` и изменяет URL содержащихся там элементов на новое значение.

Вы можете быть удивлены, обнаружив, что TortoiseSVN связывается с хранилищем в процессе выполнения этой операции. Всё что он делает - это выполняет несколько простых проверок, чтобы убедиться, что новый URL действительно ссылается на то же хранилище, что и существующая рабочая копия.

Внимание

Это очень редко используемая операция. Команда перебазирования используется *только* если изменён URL к корню хранилища. Возможные причины:

- Был изменён IP-адрес сервера.
- Был изменён протокол (например, с http:// на https://).
- Был изменён путь к корню хранилища в настройках сервера.

Другими словами, перебазирование необходимо, когда ваша рабочая копия ссылается на то же место в том же хранилище, но само хранилище было перемещено.

Перебазирование не применимо, если:

- Вы желаете перейти к другому хранилищу Subversion. В этом случае вы должны выполнить извлечение заново из нового местоположения хранилища.
- Вы желаете переключиться на другое ответвление или папку в том же хранилище. Для того, чтобы это сделать, вы должны применить TortoiseSVN → Переключить.... Прочтите Раздел 4.19.2, «Извлечь? Или переключиться?..» для получения дополнительной информации.

Если вы применили перебазирование в любом из вышеперечисленных случаев, *ваша рабочая копия будет повреждена* и вы получите множество необъяснимых сообщений об ошибках при обновлении, фиксации, и т.д. После того, как это случилось, единственным решением будет выполнение свежего извлечения.

4.28. Интеграция с системами отслеживания ошибок/проблем

При разработке программ очень часто бывает, что изменения следует связать с определенным ID ошибки или проблемы. Пользователи системы отслеживания ошибок (или системы отслеживания проблем) хотели бы связывать изменения, сделанные ими в Subversion, с конкретным ID в этой системе. Поэтому большинство систем отслеживания проблем предоставляют выполняемый перед фиксацией скрипт ловушки, который анализирует сообщение журнала для обнаружения ID ошибки, с которой связана эта фиксация. Этот способ до некоторой степени подвержен ошибкам, поскольку рассчитан на то, что пользователь напишет сообщение журнала должным образом, чтобы скрипт ловушки 'перед-фиксацией' смог разобрать его правильно.

TortoiseSVN может помочь пользователю двумя способами:

1. Когда пользователь вводит сообщение журнала, к нему может быть автоматически добавлена заранее определённая строка, содержащая номер проблемы, связанной с этой фиксацией. Это уменьшает риск того, что пользователь введёт номер проблемы таким образом, что инструменты отслеживания ошибок не смогут правильно его обработать.

   Или TortoiseSVN может подсвечивать ту часть введённого сообщения журнала, которая будет распознана системой отслеживания проблем. Благодаря этому пользователь поймёт, что сообщение журнала может быть обработано правильно.

2. Когда пользователь просматривает сообщения журнала, TortoiseSVN создаёт ссылку для каждого ID ошибки в сообщении журнала, по которой может быть запущен веб-обозреватель для просмотра описания соответствующей проблемы.

4.28.1. Добавление номеров проблем к сообщениям журнала

Вы можете интегрировать выбранную вами систему отслеживания ошибок с TortoiseSVN. Для этого вы должны определить некоторые свойства, начинающиеся с `bugtraq:`. Они должны быть установлены для папок (Раздел 4.17, «Установки проекта»).

Есть два способа интегрировать TortoiseSVN с системами отслеживания проблем: один основан на простых строках, другой - на *регулярных выражениях*. Свойства, используемые в обоих подходах:

bugtraq:url
: Задайте в этом свойстве адрес URL вашей системы отслеживания ошибок. Адрес должен быть корректно URI-закодирован и должен содержать строку `%BUGID%`. `%BUGID%` заменяется введённым вами номером проблемы. Это позволяет TortoiseSVN отобразить ссылку в диалоге журнала, чтобы при просмотре журнала ревизий вы могли перейти прямо в вашу систему отслеживания ошибок. Можно и не задавать это свойство, но тогда TortoiseSVN отображает только номер проблемы, без ссылки. Например, в проекте TortoiseSVN используется строка `http://issues.tortoisesvn.net/?do=details&id=%BUGID%`

  Вы можете также использовать относительные URL вместо абсолютных. Это может пригодиться, когда ваша система отслеживания проблем расположена в том же домене/на том же сервере, что и ваше хранилище исходного кода. В случае изменения имени домена, вам не надо будет донастраивать свойство `bugtraq:url`. Есть два способа указания относительного адреса URL:

  Если он начинается со строки `^/` предполагается, что он задан относительно корня хранилища. Например, `^/../?do=details&id=%BUGID%` будет разрешаться в `http://tortoisesvn.net/?do=details&id=%BUGID%` если ваше хранилище расположено по адресу `http://tortoisesvn.net/svn/trunk/`.

  URL, начинающийся со строки `/` предполагается заданным относительно имени сервера. Например, `/?do=details&id=%BUGID%` будет разрешаться в `http://`

`tortoisesvn.net/?do=details&id=%BUGID%` если ваше хранилище расположено где-либо на `http://tortoisesvn.net`.

bugtraq:warnifnoissue
: Установите это свойство в `true`, если желаете, чтобы TortoiseSVN предупреждал вас о незаполненном поле с номером проблемы. Допустимые значения: `true/false`. *Если свойство не задано, предполагается значение `false`.*

4.28.1.1. Номер проблемы в текстовом поле

При простом подходе TortoiseSVN показывает пользователю отдельное поле, в которое может быть введен ID ошибки. Затем это введённое пользователем значение добавляется к сообщению в конце/в начале как отдельная строка.

bugtraq:message
: Это свойство активирует систему отслеживания ошибок в режиме *поля ввода*. Если это свойство установлено, тогда TortoiseSVN будет просить ввести номер ошибки при фиксации ваших изменений. Этот номер используется для добавления строки в конец сообщения журнала, для чего сообщение должно содержать `%BUGID%`, которое заменяется на номер ошибки при фиксации. Это служит для обеспечения того, чтобы содержащаяся в журнале фиксаций ссылка на номер проблемы всегда имела совместимый формат и могла быть обработана системой отслеживания ошибок для связывания номера проблемы с конкретной фиксацией. Например, вы можете использовать `Проблема: %BUGID%`, но это зависит от используемой вами системы.

bugtraq:append
: Это свойство определяет, будет ли ID ошибки добавляться в конец сообщения журнала (значение true) или вставляться в начале сообщения (значение false). Допустимые значения: `true/false`. *Если не задано, предполагается значение `true`, чтобы не повредить существующим проектам.*

bugtraq:label
: Этот текст отображается TortoiseSVN в диалоге фиксации для обозначения поля ввода, в которое вы вводите номер проблемы. Если свойство не задано, отображается `Bug-ID / Issue-Nr:`. Помните, что окно не будет изменять размеры для размещения этой метки, поэтому размер этой метки не должен превышать 20-25 символов.

bugtraq:number
: Если установлено в `true`, в поле номера проблемы допускаются только цифры, за исключением запятой, которая может применяться в качестве разделителя при вводе нескольких номеров. Допустимые значения: `true/false`. *Если не задано, предполагается значение `true`.*

4.28.1.2. Номера проблем с использованием регулярных выражений

При подходе с `регулярными выражениями`, TortoiseSVN не показывает отдельного поля ввода, но помечает ту часть введённого пользователем сообщения журнала, которая будет распознана системой отслеживания ошибок. Это делается, пока пользователь пишет сообщение журнала. Это также означает, что ID ошибки может быть в любом месте сообщения журнала! Этот метод намного более гибкий, и именно он используется в самом проекте TortoiseSVN.

bugtraq:logregex
: Это свойство подключает систему отслеживания ошибок в режиме *регулярных выражений*. Оно содержит либо одно, либо два регулярных выражения, по одному в строке.

  Если заданы два выражения, то первое используется как предварительный фильтр для обнаружения выражений, которые содержат ID ошибок. Второе выражение после этого извлекает чистые ID ошибок из результата первого регулярного выражения. Это позволяет

использовать списки ID ошибок и обороты естественного языка при желании. Например, вы могли исправить несколько ошибок и написать строку вроде этой: «Это изменение решает проблемы #23, #24 и #25»

Если вы желаете отловить ID ошибок как в вышеприведённом выражении в сообщении журнала, вы можете применить следующие строки регулярных выражений, которые используются в проекте TortoiseSVN: `[Ii]ssues?:?(\s*(,|and)?\s*#\d+)+` и `(\d+)`

Первое выражение достаёт строку «issues #23, #24 and #25» из окружающего её сообщения журнала. Второе выражение извлекает просто десятичные числа из выдачи первого выражения, и оно вернёт «23», «24» и «25» для использования в качестве ID ошибок.

Разбирая по частям первое выражение: строка должна начинаться со слова «issue», возможно, с заглавной буквы. После этого следует необязательная «s» (более одной проблемы) и необязательное двоеточие. После чего идут одна или более групп, с нулём или более ведущих пробелов перед каждой, необязательная запятая или «and» и ещё дополнительные пробелы. В заключение, должны быть обязательный символ «#» и обязательное десятичное число.

Если задано только одно выражение, тогда группы в строке регулярного выражения должны соответствовать 'чистым' ID ошибок. Пример: `[Ii]ssue(?:s)?  #?(\d+)` Этот метод требуется некоторым системам отслеживания проблем, например trac, но в нём труднее построить регулярное выражение. Мы рекомендуем, чтобы вы использовали этот метод, если об этом явно сказано в документации вашей системы отслеживания проблем.

Если вы незнакомы с регулярными выражениями, взгляните на введение по адресу *http://en.wikipedia.org/wiki/Regular_expression*[4], а также на доступные в Сети документацию и учебный курс по адресу *http://www.regular-expressions.info/*.

Если установлены оба свойства, `bugtraq:message` и `bugtraq:logregex`, свойство `logregex` имеет преимущество.

Подсказка

Даже если у вас нет системы отслеживания проблем с выполняемой перед фиксацией ловушкой, разбирающей ваши сообщения журнала, вы всё равно можете применить эту возможность для преобразования в ссылки проблем, упомянутых в сообщениях!

И даже если эти ссылки вам не нужны, номера проблем показываются в отдельной колонке в диалоге журнала, что позволяет легче обнаруживать изменения, относящиеся к определённой проблеме.

Некоторые `tsvn:`-свойства требуют значений `true/false`. TortoiseSVN также понимает `yes` как синоним `true` и `no` как синоним `false`.

Устанавливайте свойства на папках

Эти свойства проекта должны быть установлены на *папках*, для того чтобы эта система работала. При фиксации файла или папки свойства считываются из этой папки. Если свойства там не найдены, TortoiseSVN будет искать их выше по дереву папок, пока не достигнет неверсированной папки или корня дерева (например, `C:\`). Если вы можете быть уверены, что каждый пользователь извлекает только из одной папки (например, `trunk/`), а не из какой-то подпапки, тогда достаточно установить свойства для `trunk/`. Если вы в этом не уверены, тогда вы должны устанавливать

[4]Есть статья об этом и в русском разделе Википедии: *http://ru.wikipedia.org/wiki/Регулярные_выражения* [http://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B3%D1%83%D0%BB%D1%8F%D1%80%D0%BD%D1%8B%D0%B5_%D0%B2%D1%8B%D1%80%D0%B0%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F] - прим. переводчика

свойства рекурсивно для каждой подпапки. Установка свойств ниже по иерархии проекта перекрывает свойства более высоких уровней (ближе к `trunk/`).

Только для свойств `tsvn:` вы можете использовать флажок Рекурсивно для установки свойства для всех подпапок в иерархии, без установки его также для всех файлов.

Никакой информации системы отслеживания проблем в обозревателе хранилища

Поскольку интеграция с системой отслеживания проблем зависит от доступа к свойствам Subversion, вы сможете увидеть результаты только при использовании извлечённой рабочей копии. Получение свойств удалённо - медленная операция, поэтому вы не увидите работу этой воможности в обозревателе хранилища.

Эта интеграция с системой отслеживания проблем не ограничена только TortoiseSVN; она может быть использована любым клиентом Subversion. Для получения дополнительной информации прочтите полную *Спецификацию интеграции с системами отслеживания проблем (Issue Tracker Integration Specification)* [http://tortoisesvn.tigris.org/svn/tortoisesvn/trunk/doc/issuetrackers.txt] в хранилище исходного кода TortoiseSVN (Раздел 3, «TortoiseSVN бесплатен!» рассказывает, как получить доступ к хранилищу исходного кода TortoiseSVN).

4.28.2. Получение информации из системы отслеживания проблем

В предыдущем разделе мы имели дело с добавлением информации о проблемах в сообщения журнала. Но что если вам необходимо получить информацию из системы отслеживания проблем? У диалога фиксации есть COM-интерфейс, позволяющий выполнить интеграцию с внешней программой, умеющей взаимодействовать с вашей системой отслеживания проблем. Типичный запрос - получить у системы отслеживания список назначенных вам открытых проблем, чтобы вы могли выбрать проблемы, которые были затронуты в этой фиксации.

Любой такой интерфейс, конечно же, сильно зависит от особенностей вашей системы отслеживания проблем, поэтому эту часть мы предоставить не можем, и описание создания такой программы выходит за рамки этого руководства. Определение интерфейса и примеры подключаемых модулей на языках C# и C++/ATL можно посмотреть в папке `contrib` в *хранилище TortoiseSVN* [http://tortoisesvn.tigris.org/svn/tortoisesvn/trunk/contrib/issue-tracker-plugins]. (Раздел 3, «TortoiseSVN бесплатен!» рассказывает, как получить доступ к хранилищу исходного кода TortoiseSVN). Краткое описание API также дано в Глава 6, *IBugtraqProvider interface* Другой (рабочий) пример подключаемого модуля на C# - *Gurtle* [http://code.google.com/p/gurtle/], который реализует необходимый COM-интерфейс для взаимодействия с системой отслеживания проблем *Google Code* [http://code.google.com/hosting/].

Для наглядности предположим, что ваш системный администратор предоставил вам подключаемый модуль для системы отслеживания проблем, который вы установили, и что вы настроили некоторые из ваших рабочих копий на использование этого подключаемого модуля в диалоге настроек TortoiseSVN. При открытии диалога фиксации из рабочей копии, на которую назначен подключаемый модуль, вы увидите новую кнопку в верхней части диалога.

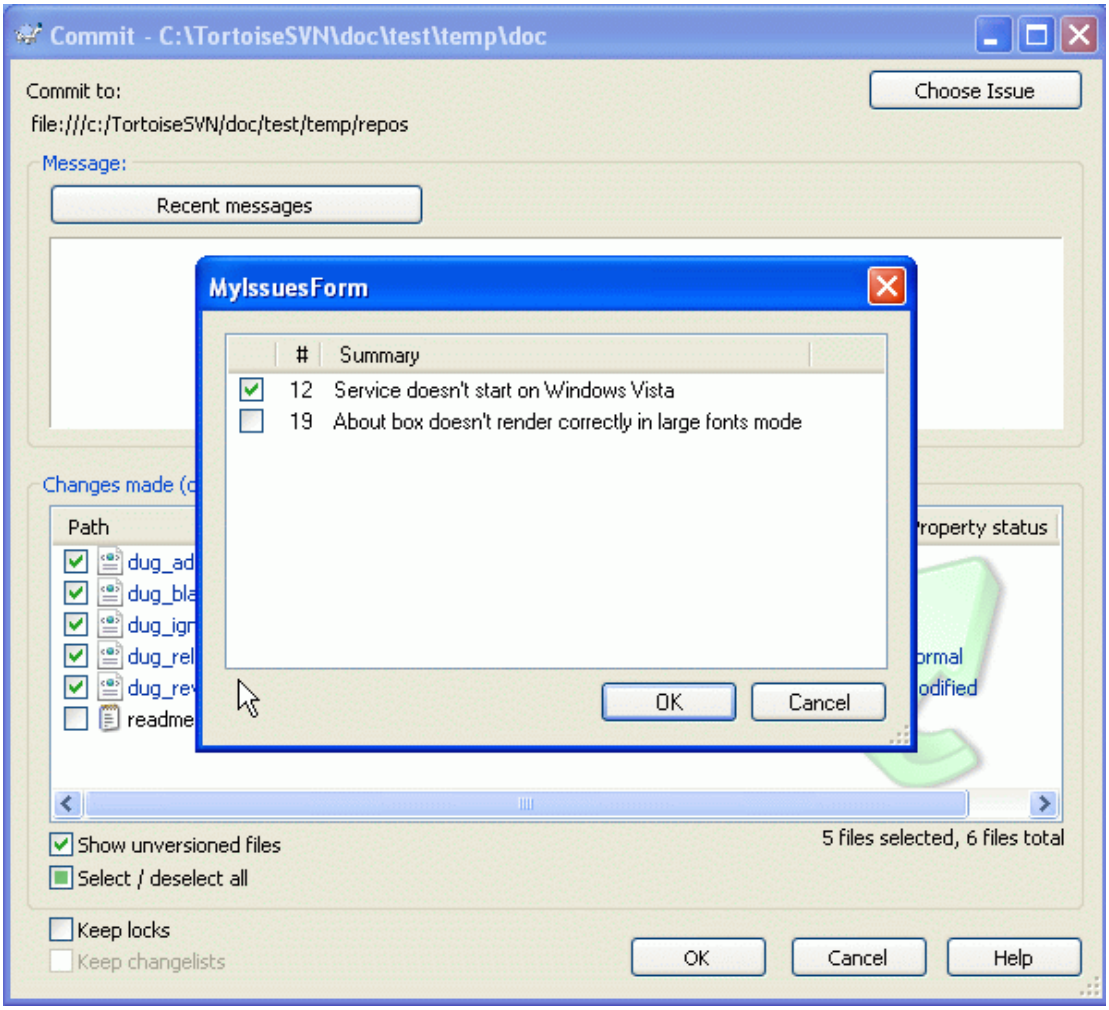


Рисунок 4.50. Пример диалога запроса системы отслеживания проблем

В этом примере вы можете выбрать одну или более открытых проблем. Затем модуль может сгенерировать специально отформатированный текст, который он добавит к вашему сообщению журнала.

4.29. Интеграция со средствами просмотра хранилища, работающими через веб-интерфейс

Есть несколько средств просмотра хранилища, работающих через веб-интерфейс, которые могут использоваться с Subversion, таких как *ViewVC* [http://www.viewvc.org/] и *WebSVN* [http://websvn.tigris.org/]. TortoiseSVN предоставляет возможность для связи с этими средствами просмотра.

Вы можете интегрировать выбранное вами средство просмотра хранилища в TortoiseSVN. Для этого вы должны задать некоторые свойства, обеспечивающие эту связь. Они должны быть установлены для папок: (Раздел 4.17, «Установки проекта»)

webviewer:revision
: Задайте в этом свойстве URL вашего средства просмотра хранилища, по которому можно посмотреть все изменения в конкретной ревизии. Оно должно быть правильно URI-закодировано и должно содержать `%REVISION%`, которое заменяется нужным номером ревизии. Это позволит TortoiseSVN показать в контекстном меню окна журнала пункт Контекстное меню → Посмотреть ревизию в веб-обозревателе

webviewer:pathrevision
: Задайте в этом свойстве URL вашего средства просмотра хранилища, по которому можно посмотреть изменения конкретного файла в конкретной ревизии. Оно должно быть правильно URI-закодировано и должно содержать `%REVISION%` и `%PATH%`. Строка `%PATH%` заменяется путём относительно корня хранилища. Это позволит TortoiseSVN показать в контекстном меню окна журнала пункт Контекстное меню → Посмотреть ревизию для пути в веб-обозревателе Например, при правом щелчке в нижней панели окна журнала на файле `/trunk/src/file`, значение `%PATH%` в URL будет заменено на `/trunk/src/file`.

Вы можете также использовать относительные URL вместо абсолютных. Это может пригодиться, когда ваша система просмотра хранилища через веб расположена в том же домене/на том же сервере, что и ваше хранилище исходного кода. В случае изменения имени домена, вам не надо будет донастраивать свойства `webviewer:revision` и `webviewer:pathrevision`. Формат применяется такой же, как и в свойстве `bugtraq:url`. Смотрите Раздел 4.28, «Интеграция с системами отслеживания ошибок/проблем».

Устанавливайте свойства на папках

Эти свойства проекта должны быть установлены на *папках*, для того чтобы эта система работала. При фиксации файла или папки свойства считываются из этой папки. Если свойства там не найдены, TortoiseSVN будет искать их выше по дереву папок, пока не достигнет неверсированной папки или корня дерева (например, `C:\`). Если вы можете быть уверены, что каждый пользователь извлекает только из одной папки (например, `trunk/`), а не из какой-то подпапки, тогда достаточно установить свойства для `trunk/`. Если вы в этом не уверены, тогда вы должны устанавливать свойства рекурсивно для каждой подпапки. Установка свойств ниже по иерархии проекта перекрывает свойства более высоких уровней (ближе к `trunk/`).

Только для свойств `tsvn:` вы можете использовать флажок Рекурсивно для установки свойства для всех подпапок в иерархии, без установки его также для всех файлов.

Никаких ссылок для просмотра хранилища в обозревателе хранилища

Поскольку интеграция со средствами просмотра хранилища зависит от доступа к свойствам Subversion, вы сможете увидеть результаты только при использовании извлечённой рабочей копии. Получение свойств удалённо - медленная операция, поэтому вы не увидите работу этой воможности в обозревателе хранилища.

4.30. Настройки TortoiseSVN

Чтобы узнать, для чего предназначены различные параметры, просто оставьте ненадолго указатель мыши на поле ввода/флажке... и появится полезная подсказка.

4.30.1. Общие настройки

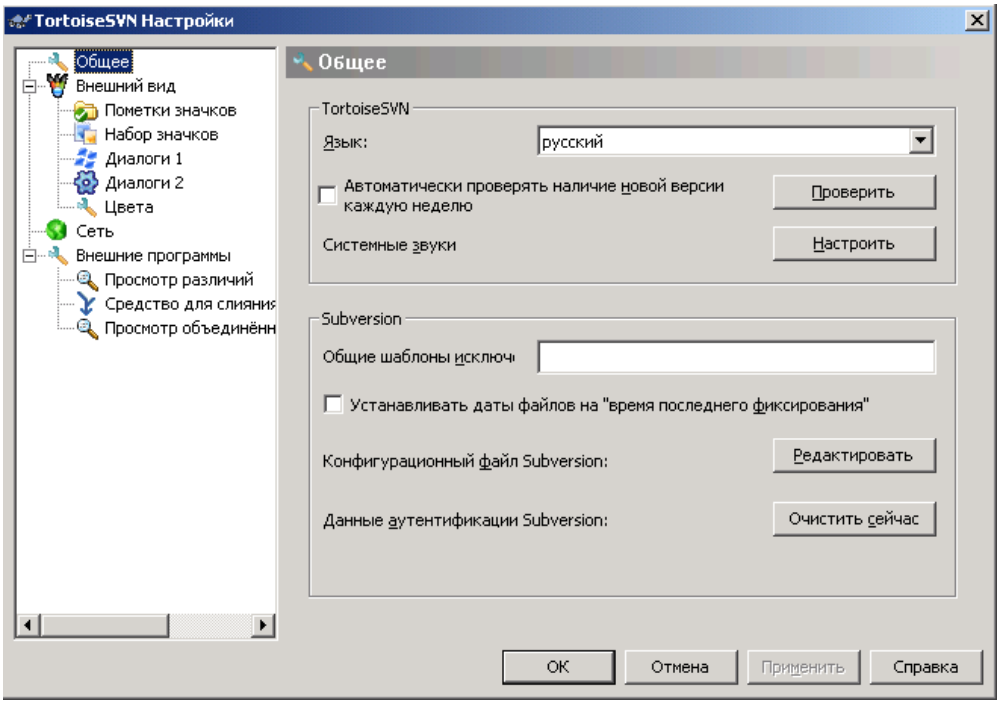


Рисунок 4.51. Страница 'Общее' в диалоге настроек

В этом диалоге можно указать предпочтитаемый вами язык интерфейса, а также некоторые специальные настройки Subversion.

Язык
: Выбор языка пользовательского интерфейса. А вы ожидали что-то другое?

Автоматически проверять наличие новой версии каждую неделю
: Если отмечено, то TortoiseSVN один раз в неделю будет связываться со своим сайтом загрузки, чтобы узнать, не появилась ли новая версия программы. Воспользуйтесь кнопкой Проверить, если вы желаете получить эту информацию прямо сейчас. Новая версия загружаться не будет, просто появляется диалог, сообщающий о том, что появилась новая версия.

Системные звуки
: TortoiseSVN по умолчанию устанавливает в систему три собственных звука.

  - Ошибка
  - Уведомление
  - Предупреждение

  Вы можете выбрать другие звуки (или вообще их отключить) при помощи панели управления Windows. Кнопка Настроить служит для её быстрого вызова.

Шаблоны игнорирования

: Общие шаблоны игнорирования используются для предотвращения отображения неверсированных файлов, например, в диалоге фиксации. Файлы, соответствующие шаблонам, игнорируются также при импорте. Игнорирование файлов или папок осуществляется путём ввода имён или расширений. Шаблоны разделяются пробелами, например `bin obj *.bak *.~?? *.jar *.[Tt]mp`. Эти шаблоны не должны включать разделители путей. Заметьте также, что нет способа провести различие файлов и папок. Прочтите Раздел 4.13.1, «Сопоставление шаблону в списках игнорирования» для дополнительной информации о синтаксисе задания шаблонов.

Обратите внимание: шаблоны игнорирования, задаваемые здесь, влияют также на других клиентов Subversion, работающих на вашем ПК, включая клиента командной строки.

Предостережение

Если вы воспользуетесь файлом настроек Subversion для установки шаблона `global-ignores`, то его значение перекроет сделанные здесь установки. Доступ к файлу настроек Subversion можно получить посредством кнопки Правка, описанной ниже.

Эти шаблоны игнорирования окажут влияние на все ваши проекты. Они не версируются, поэтому не затрагивают других пользователей. Для сравнения, вы можете также использовать версированное свойство `svn:ignore` для исключения файлов или папок из-под управления версиями. Прочтите Раздел 4.13, «Игнорирование файлов и папок» для более подробной информации.

Устанавливать даты файлов на «время последнего фиксирования»
: Эта опция предписывает TortoiseSVN устанавливать дату файлов по времени последней фиксации при выполнении извлечения или обновления. Иначе TortoiseSVN будет использовать текущую дату. Если вы разрабатываете программное обеспечение, в общем случае лучше использовать текущую дату, поскольку системы сборки обычно смотрят на метку времени для принятия решения о том, какие файлы нуждаются в компиляции. Если вы используете «время последнего фиксирования» и откатываетесь к старой ревизии файла, ваш проект вопреки ожиданиям может больше не компилироваться.

Файл настроек Subversion
: Воспользуйтесь кнопкой Правка для непосредственного редактирования файла настроек. Некоторые настройки не могут быть изменены TortoiseSVN напрямую, и вместо этого должны быть заданы здесь. Для более подробной информации о файле `config` Subversion смотрите *Область настроек времени выполнения (Runtime Configuration Area)* [http://svnbook.red-bean.com/en/1.5/svn.advanced.confarea.html]. Раздел *Автоматическая установка свойств (Automatic Property Setting)* [http://svnbook.red-bean.com/en/1.5/svn.advanced.props.html#svn.advanced.props.auto] особенно интересен, и он настраивается именно здесь. Заметьте, что Subversion может считывать информацию о настройках из нескольких мест, и вам необходимо знать, которое из них имеет приоритет. Прочтите *Конфигурация и реестр Windows (Configuration and the Windows Registry)* [http://svnbook.red-bean.com/en/1.5/svn.advanced.confarea.html#svn.advanced.confarea.windows-registry], чтобы узнать больше.

Использовать папки `_svn` вместо `.svn`
: VS.NET при использовании с веб-проектами не может обрабатывать папки `.svn`, которые Subversion использует для хранения своей внутренней информации. Это не ошибка в Subversion, это ошибка в VS.NET и в используемых им расширениях frontpage. Прочтите Раздел 4.30.11, «Рабочие папки Subversion» для дополнительной информации об этой проблеме.

  Если вы желаете изменить поведение Subversion и TortoiseSVN, вы можете использовать этот флажок для установки переменной окружения, которая этим управляет.

  Вы должны осознавать, что изменение этой опции не конвертирует автоматически существующие рабочие копии так, чтобы они использовали новую административную папку. Вы должны сделать это самостоятельно, используя скрипт (см. наше ЧаВо), или просто извлечь рабочую копию заново.

4.30.1.1. Настройки контекстного меню

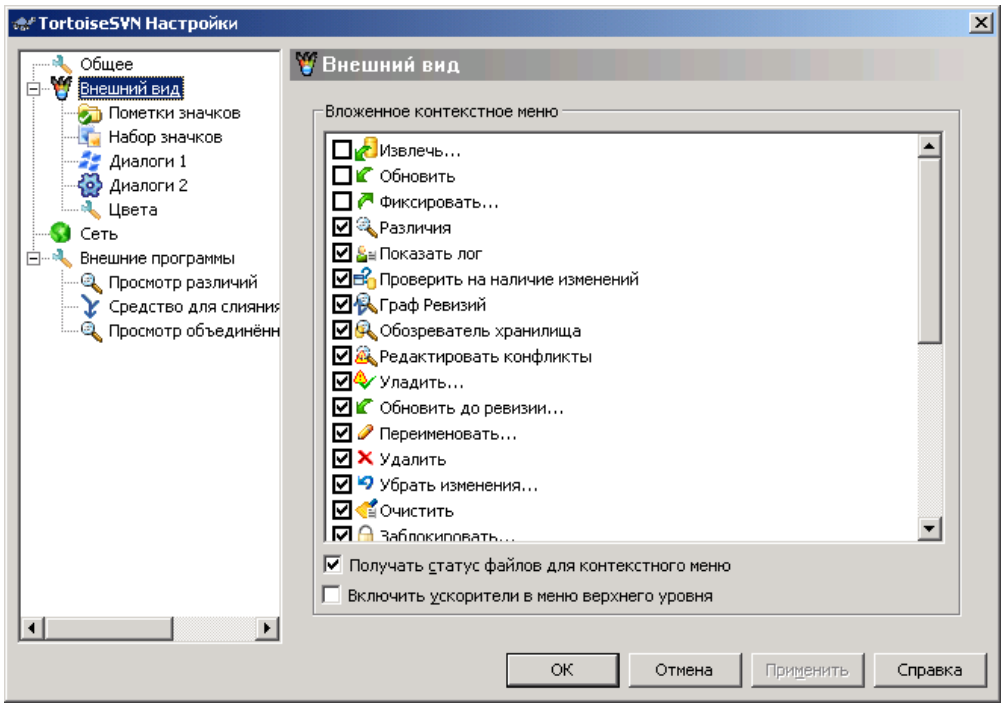


Рисунок 4.52. Страница контекстного меню в диалоге настроек

Эта страница позволяет вам указать, какие пункты контекстного меню TortoiseSVN будут отображаться в основном контекстном меню, а какие - в подменю. По умолчанию, большинство пунктов не отмечены и отображаются в подменю.

Но есть особый случай: пункт меню Заблокировать. Конечно, вы можете переместить его в меню верхнего уровня при помощи вышеуказанного списка, но поскольку большинство файлов блокировать не надо, это только создаст дополнительную помеху. Однако, для файла с установленным свойством `svn:needs-lock`, это действие необходимо производить при каждом редактировании, поэтому в этом случае очень полезно, чтобы соответствующий пункт меню был доступен сразу. Отметка на флажке означает, что когда выбран файл с установленным свойством `svn:needs-lock`, пункт Заблокировать всегда будет появляться в меню верхнего уровня.

Если вы желаете, чтобы контекстное меню TortoiseSVN по некоторым путям в вашем компьютере не показывалось вообще, вы можете указать такие пути в поле снизу.

4.30.1.2. Настройки диалогов TortoiseSVN - 1

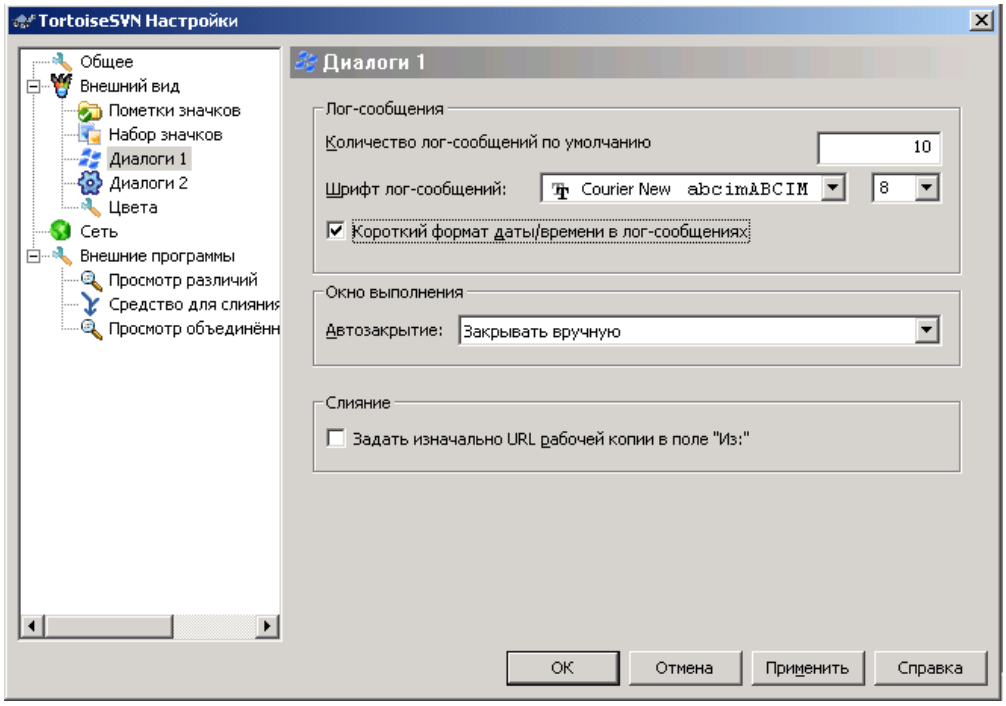


Рисунок 4.53. Страница 'Диалоги 1' в диалоге настроек

Эта страница позволяет настроить некоторые диалоги TortoiseSVN под ваши предпочтения.

Количество сообщений журнала по умолчанию
: Ограничивает число сообщений журнала, которые TortoiseSVN извлекает при первом вызове TortoiseSVN → Журнал Эта опция полезна при медленных соединениях с серверами[5]. Вы всегда можете использовать Показать все или Следующие 100 для получения остальных сообщений.

Шрифт сообщений журнала
: Выбирает начертание и размер шрифта, используемого для отображения самого сообщения журнала в средней панели окна журнала ревизий, а также при составлении сообщений в диалоге фиксации.

Короткий формат даты/времени в сообщениях журнала
: Если стандартные длинные сообщения занимают слишком много места на экране, можно использовать короткий формат.

Включить сравнение ревизии в списке с предыдущей по двойному щелчку
: Если вы заметили, что часто сравниваете ревизии в верхней панели диалога журнала, вы можете воспользоваться этой опцией для разрешения этого действия по двойному щелчку. Часто, получение различий - длительный процесс, а многие предпочитают избежать ожидания после случайного двойного щелчка, и именно поэтому эта опция по умолчанию отключена.

Окно выполнения
: TortoiseSVN может автоматически закрывать все окна, отображающие процесс выполнения, при условии успешного завершения действия. Эта настройка позволяет вам выбрать условия для закрытия этих окон. Значение по умолчанию (рекомендуемое) - Закрывать вручную, которое позволяет вам просмотреть все сообщения и проверить, что произошло. Однако, вы

[5] Да и для ограничения трафика тоже - прим. переводчика

можете принять решение о игнорировании некоторых типов сообщений, так чтобы диалог закрывался автоматически в случае отсутствия критических изменений.

Автозакрытие: не было слияний, добавлений или удалений означает, что окно выполнения будет закрыто, если были только простые обновления, но, если производилось слияние изменений из хранилища с вашими, или если какие-либо файлы были добавлены или удалены, окно останется открытым. Оно также останется открытым, если возникли конфликты или ошибки во время выполнения этого действия.

Автозакрытие при локальных операциях означает, что окно выполнения будет закрыто в тех же случаях, что и Автозакрытие: не было слияний, добавлений или удалений, но только для локальных операций, таких как добавление файлов или отмена изменений. Для удалённых действий окно останется открытым.

Автозакрытие при отсутствии конфликтов ещё больше смягчает условия и будет закрывать окно даже в случае наличия слияний, добавлений или удалений. Однако, если возникли какие-нибудь конфликты или ошибки, окно останется открытым.

Автозакрытие при отсутствии ошибок всегда закрывает окно, даже если были конфликты. Окно остаётся открытым только в случае ошибок, которые не позволили Subversion выполнить задачу. Например, обновление не удалось из-за недоступности сервера, или фиксация не удалась из-за устаревания рабочей копии.

Использовать корзину при отмене изменений
: Когда вы убираете локальные изменения, все выполненные вами изменения отбрасываются. TortoiseSVN предоставляет дополнительную страховку путём отправки изменённого файла в корзину перед тем, как вернуть нетронутую копию. Если вы предпочитаете не задействовать корзину, отключите эту опцию.

Задать изначально URL рабочей копии в поле «Из:»
: В диалоге слияния поведением по умолчанию является сохранение URL в поле Из: между слияниями. Однако, некоторые люди предпочитают выполнять слияния из нескольких различных точек в иерархии, и считают более простым начинать с URL текущей рабочей копии, который затем может быть отредактирован для указания на параллельный путь в другом ответвлении.

Путь по умолчанию
: Вы можете задать путь по умолчанию для извлечения. Если вы располагаете все ваши извлечения в одном месте, бывает полезно, когда в этом поле предварительно указан соответствующий путь, чтобы вам оставалось только дописать в конце имя новой папки.

URL по умолчанию
: Вы также можете указать и адрес URL по умолчанию для извлечения. Если вы часто извлекаете подпроекты какого-нибудь большого проекта, может оказаться полезным предварительное заполнение поля URL, чтобы достаточно было добавить только имя подпроекта в конце.

4.30.1.3. Настройки диалогов TortoiseSVN - 2

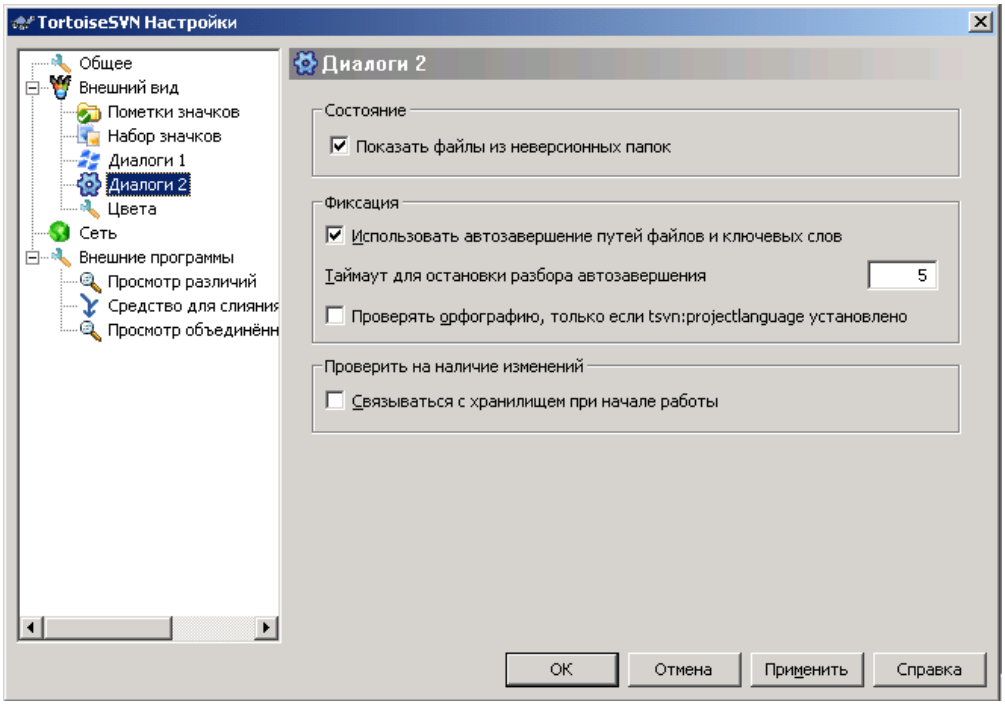


Рисунок 4.54. Страница 'Диалоги 2' в диалоге настроек

Показать файлы из неверсированных папок
: Если этот флажок отмечен (состояние по умолчанию), то при отображении статуса для неверсированной папки в диалогах Добавление, Фиксация или Проверка на наличие изменений показываются также все её дочерние файлы и папки. Если снять пометку, будет показана только неверсированная папка, что снижает избыточность списка файлов в этих диалогах. В этом случае при выборе неверсированной папки для добавления она добавляется рекурсивно.

Использовать автозавершение для путей файлов и ключевых слов
: Диалог фиксации содержит механизм анализа списка имён фиксируемых файлов. При вводе первых 3 символов какого-либо элемента из списка, появляется окно автозавершения с полным вариантом написания, и вы можете нажать клавишу ввода для завершения имени файла. Отметьте флажок для включения этой функции.

Таймаут для остановки разбора автозавершения (сек)
: Анализатор автозавершения может быть довольно медленным, если ему приходиться проверять множество больших файлов. Этот таймаут не позволяет диалогу фиксации заниматься анализом слишком долго, но если вам необходима некая важная информация, предоставляемая автозавершением, вы можете увеличить этот таймаут.

Проверять орфографию, только если `tsvn:projectlanguage` установлено
: Если вы не желаете использовать проверку орфографии при всех фиксациях, пометьте этот флажок. Проверка орфографии будет по-прежнему разрешена там, где её требуют свойства проекта.

Количество сохраняемых предыдущих сообщений журнала
: Когда вы набираете сообщение журнала в диалоге фиксации, TortoiseSVN сохраняет его для возможного повторного использования в последующем. По умолчанию, сохраняются последние 25 сообщений журнала для каждого хранилища. Вы можете изменить это число: если у вас много различных хранилищ, вы можете уменьшить его для сокращения количества записей в вашем реестре.

Обратите внимание: эта настройка применяется только к сообщениям, вводимым на этом компьютере. Это не имеет никакого отношения к кэшированию сообщений журнала.

Переоткрывать диалоги фиксации/ответвлен. после неуспешной фиксации
: Когда по какой-либо причине фиксация завершается неудачей (необходимо обновление рабочей копии, фиксацию отклоняет ловушка "перед-фиксацией", сетевая ошибка и т.д.), можно включить эту опцию, чтобы диалог фиксации оставался открытым и готовым к тому, чтобы попробовать ещё раз. Однако вы должны понимать, что это может привести к проблемам: если сбой предполагает необходимость обновления рабочей копии и это обновление приведёт к конфликтам, то вам сначала придётся их уладить.

Автоматический выбор элементов
: При обычном поведении в диалоге фиксации все изменённые (версированные) элементы отмечаются для фиксации автоматически. Если вы предпочитаете начинать из состояния, когда ничего не отмечено, и выбирать необходимые элементы для фиксации вручную, снимите пометку с этого флажка.

Связываться с хранилищем при начале работы
: Диалог 'Проверка на наличие изменений' по умолчанию проверяет рабочую копию, и соединяется с хранилищем, только если вы нажмёте Проверить хранилище. Если вы желаете всегда проверять хранилище, вы можете использовать эту установку для выполнения этого действия автоматически.

Показывать диалог блокировки перед блокированием файлов
: При выборе одного или более файлов и последующем применении TortoiseSVN → Заблокировать для блокировки этих файлов, в некоторых проектах обычно предлагается написать сообщение с объяснением, для чего вы блокируете файлы. Если вы не используете сообщения блокировки, вы можете снять отметку с этого флажка, и тогда этот диалог будет пропущен и файлы будут блокироваться сразу.

  При применении команды блокировки к папке, всегда показывается диалог блокировки, поскольку он также предоставляет возможность выбора файлов для блокировки.

  Если ваш проект использует свойство `tsvn:lockmsgminsize`, диалог блокировки будет показан независимо от этой настройки, так как сообщения блокировки являются *обязательными* в этом проекте.

4.30.1.4. Настройки цветов в TortoiseSVN

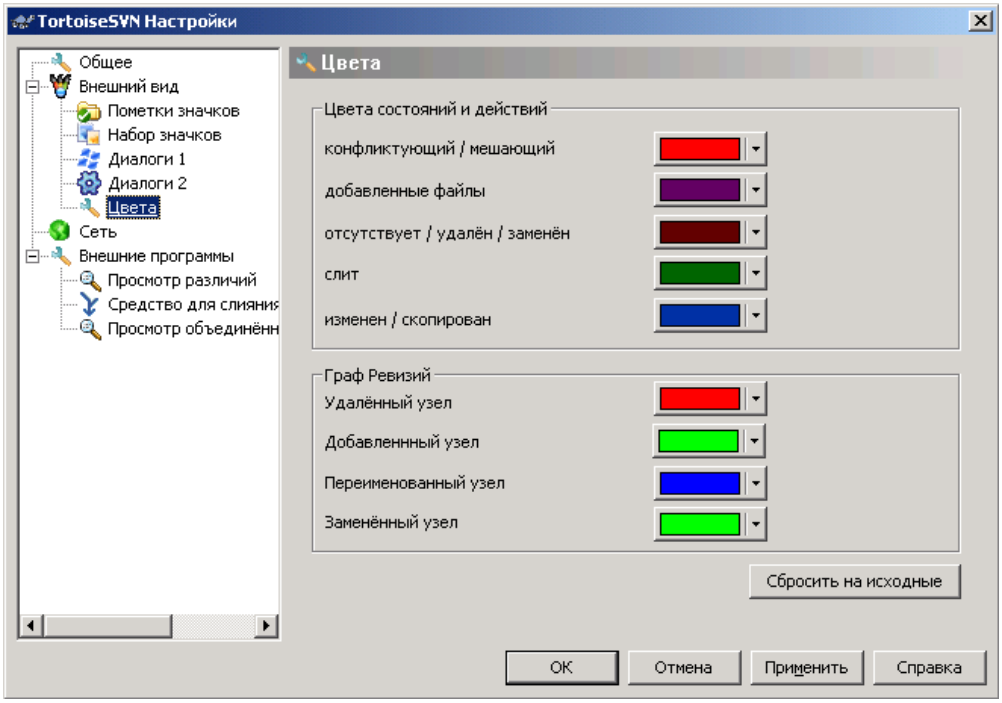


Рисунок 4.55. Страница 'Цвета' в диалоге настроек

Этот диалог позволяет настроить цвета текстов, используемых в диалогах TortoiseSVN, как вам больше нравится.

Конфликтующий / мешающий
: Конфликт, возникший во время обновления, или который может возникнуть при слиянии. Обновление, которому мешает существующий неверсированный файл/папка с тем же именем, что и версированный.

  Этот цвет также используется для сообщений об ошибках в окне выполнения.

Добавленные файлы
: Элементы, добавленные в хранилище.

Отсутствует / удалён / замещён
: Элементы, удалённые из хранилища, отсутствующие в рабочей копии, или удалённые из рабочей копии и замещённые другим файлом с тем же именем.

Слит
: Изменения из хранилища, успешно объединённые с рабочей копией без возникновения конфликтов.

Изменён / скопирован
: Добавление с историей, или скопированные в хранилище пути. Также используется в диалоге журнала для вхождений, включающих скопированные элементы.

Удалённый узел
: Элемент, который был удалён из хранилища.

Добавленный узел
: Элемент, который был добавлен в хранилище посредством операции добавления, копирования или перемещения.

Переименованный узел
: Элемент, который был переименован в хранилище.

Замещённый узел
: Элемент, который был замещён новым элементом с таким же именем после удаления исходного.

4.30.2. Настройки графа ревизий

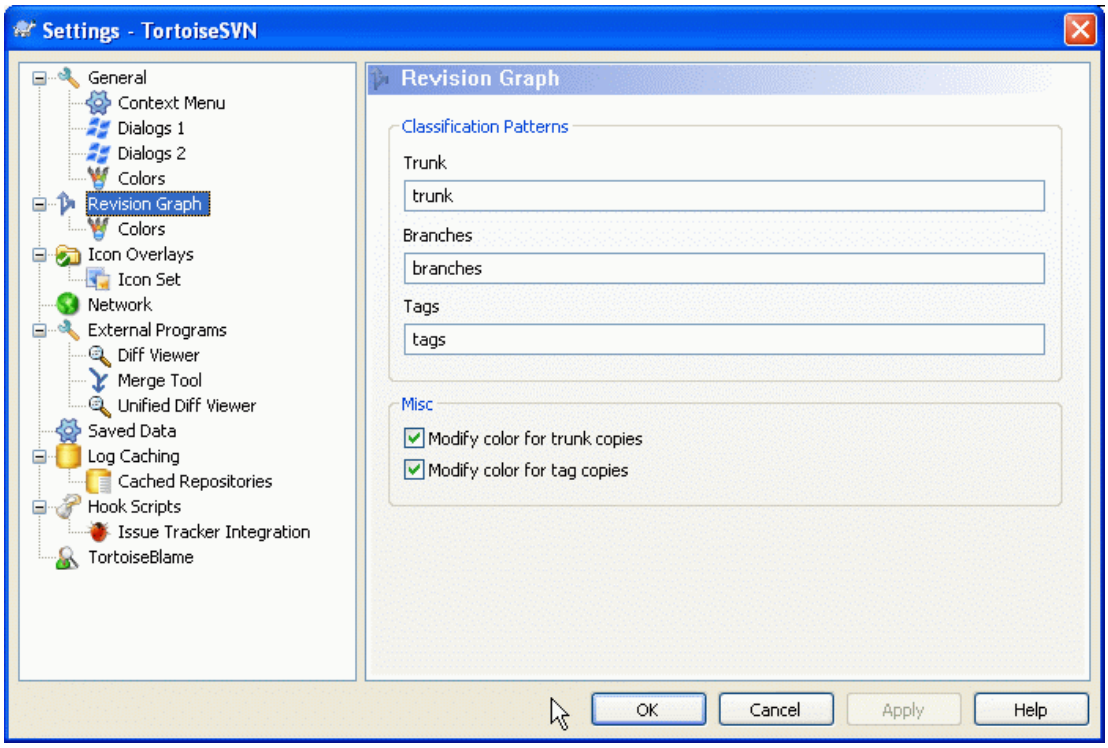


Рисунок 4.56. Страница 'Граф ревизий' в диалоге настроек

Шаблоны классификации
: Граф ревизий пытается показать более ясную картину структуры вашего хранилища, выделяя ствол, ответвления и метки. Поскольку такой встроенной классификации в Suversion нет, эта информация извлекается из путей. Настройки по умолчанию предполагают, что вы используете традиционные наименования на английском, предложенные в документации Subversion, но, конечно же, вы можете использовать и что-нибудь другое.

  Укажите шаблоны, используемые для выделения путей, в соответсвующих полях ввода. Шаблоны распознаются независимо от регистра, но вы должны ввести их в нижнем регистре. Подстановочные символы `*` и `?` работают как обычно, и вы можете использовать `;` для разделения нескольких шаблонов. Не вводите дополнительные пробельные символы, поскольку они также будут включены в шаблон при сопоставлении.

Изменение цвета
: Цвета используются в графе ревизий для обозначения типа узла, т.е. был ли узел добавлен, удалён, переименован. Для того, чтобы помочь в классификации узлов, вы можете разрешить в графе ревизий смешивать цвета, что даст информацию сразу и о типе узла, и о его классификации. Смешивание используется, если флажок отмечен. Если флажок не отмечен, цвет используется только для обозначения только типа узла. Воспользуйтесь диалогом выбора цветов для того, чтобы задействовать определённые цвета.

4.30.2.1. Цвета в графе ревизий

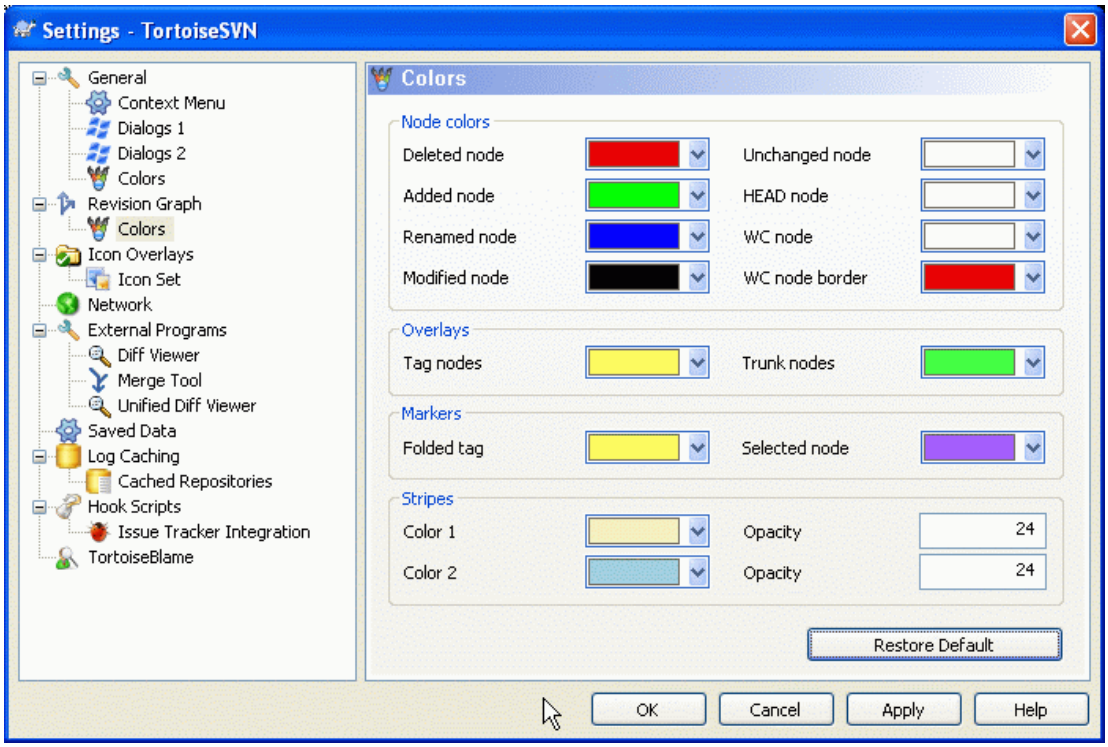


Рисунок 4.57. Страница 'Цвета' графа ревизий в диалоге настроек

Эта страница позволяет настроить используемые цвета. Учтите, что здесь указывается читый цвет. Большинство узлов разцвечиваются смесью цвета типа узла, цвета фона и, при выборе, цветом классиикации.

Удалённый узел
: Элементы, которые были удалены и не были скопированы куда-либо ещё в той же ревизии.

Добавленный узел
: Вновь добавленные элементы, или скопированные (добавленные с историей).

Переименованный узел
: Элементы, удалённые в одном месте и добавленные в другом в той же ревизии.

Изменённый узел
: Простые изменения без добавлений или удалений.

Неизменённый узел
: Может быть использован для отображения ревизии, использованной как источник копирования, даже если в этой ревизии изменений не было (у элемента, показываемом в графе).

Ведущий узел
: Текущая ведущая ревизия в хранилище (HEAD).

Узел рабочей копии
: Если вы задали отображение дополнительного узла для вашей изменённой рабочей копии, соединённого с последней зафиксированной ревизией в графе, то используется этот цвет.

Рамка узла рабочей копии
: Если вы задали отображение того факта, что рабочая копия была изменена, этот цвет используется для рамки узла рабочей копии при наличии изменений.

Узлы меток
: К узлам, классифицированным как метки, может быть примешан этот цвет.

Узлы ствола
: К узлам, классифицированным как ствол, может быть примешан этот цвет.

Обозначение свёрнутых меток
: Если вы используете сворачивание меток для экономии пространства, метки в источнике копирования обозначаются при помощи прямоугольника этого цвета.

Обозначение выбранного узла
: При выделении узла щелчком левой кнопки, выбор обозначается прямоугольником этого цвета.

Полосы
: Эти цвета используются, когда граф разделён на под-деревья и фон раскрашен в чередующиеся полосы, способствующие разделению различных деревьев.

4.30.3. Настройки пометок на значках

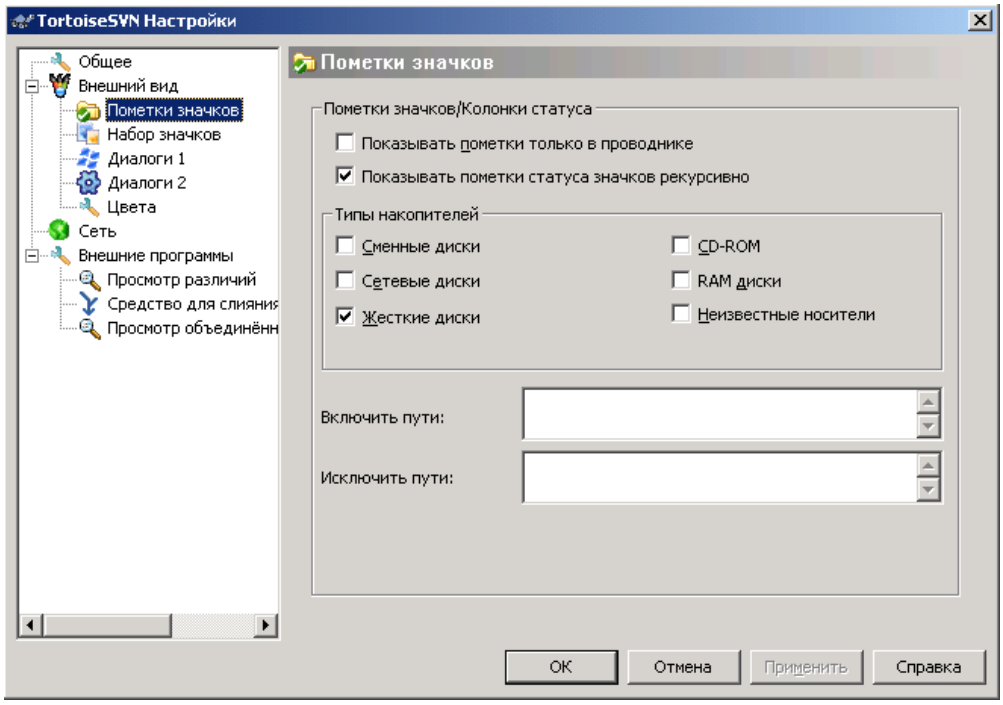


Рисунок 4.58. Страница 'Пометки на значках' в диалоге настроек

Эта страница позволяет выбрать элементы, для которых TortoiseSVN будет показывать пометки на значках.

По умолчанию, пометки на значках и контекстное меню появляются не только в Проводнике, но и во всех диалогах открытия/сохранения. Если вы желаете, чтобы они появлялись *только* в Проводнике, отметьте флажок Показывать пометки и контекстное меню только в проводнике.

Для игнорируемых и неверсированных элементов пометки на значках обычно не показываются. Если вы желаете, чтобы значки показывались и в этих случаях, просто отметьте соответствующие флажки.

Вы также сделать так, чтобы папки помечались как изменённые, если в них есть неверсированные элементы. Это может пригодиться для напоминания вам о свежесозданных, пока ещё неверсированных, файлах. Эта опция доступна только при использовании варианта кэширования статуса *по умолчанию* (см.ниже).

Поскольку получение информации о статусе рабочей копии занимает некоторое время, TortoiseSVN использует кэширование для сохранения статуса, чтобы Проводник не слишком 'задумывался' при отображении пометок. Здесь вы можете выбрать, какой тип кэша TortoiseSVN должен использовать, исходя из возможностей вашей системы и размеров рабочих копий:

По умолчанию
: Кэширует всю информацию о статусе в отдельном процессе (`TSVNCache.exe`). Этот процесс наблюдает за изменениями на всех дисках и заново получает статус при изменении файлов внутри рабочей копии. Процесс запускается с наименьшим возможным приоритетом, чтобы не мешать другим программам. Это также означает, что информация о статусе обновляется *не в реальном времени*, и может потребоваться несколько секунд, прежде чем пометки изменятся.

  Преимущества: пометки отображают статус рекурсивно, т.е. если был изменён файл, расположенный глубоко в рабочей копии, то все папки выше по иерархии, вплоть до корня рабочей копии, также будут помечены как изменённые. И поскольку процесс может посылать уведомления оболочке, также обычно изменяются и пометки в дереве левой панели Проводника.

  Недостатки: процесс работает постоянно, даже если вы не работаете над своими проектами. Он также использует примерно 10-50 Мб ОЗУ, в зависимости от количества и размера ваших рабочих копий.

В оболочке
: Кэширование выполняется непосредственно внутри DLL расширения оболочки, но только для папок, видимых в данный момент. Каждый раз при переходе к другой папке информация о статусе получается заново.

  Преимущества: необходимо совсем немного памяти (около 1Мб ОЗУ) и можно отображать статус в *реальном времени*.

  Недостатки: поскольку кэшируется только одна папка, пометки не отображают статус рекурсивно. Для больших рабочих копий может потребоваться больше времени для отображения папки в Проводнике, нежели с кэшированием по умолчанию. Также в этом случае становиться недоступным столбец mime-типа.

Без кэширования
: При этой установке TortoiseSVN вообще не получает статус в Проводнике. Из-за этого пометки на файлах не отображаются, а на папках показываются пометки 'нормальный', только если они версированные. Никакие другие пометки не отображаются, и также недоступны все дополнительные столбцы.

  Преимущества: совершенно не требует дополнительной памяти и вообще не замедляет Проводник во время просмотра.

  Недостатки: информация о статусе файлов и папок в Проводнике не отображается. Чтобы увидеть, была ли изменена ваша рабочая копия, вы должны использовать диалог «Проверка на наличие изменений».

Следующая группа позволяет выбрать, для каких классов накопителей будут отображаться пометки на значках. По умолчанию, выбраны только жёсткие диски. Вы можете даже полностью отключить показ пометок, но что это вам даст?

Сетевые диски могут быть очень медленными, поэтому по умолчанию пометки не показываются в рабочих копиях, расположенных на сетевых разделяемых ресурсах.

USB флэш-накопители, похоже, являются особым случаем, поскольку тип диска определяется самим устройством. Некоторые показываются как жёсткие диски, а некоторые - как сменные носители.

Исключить пути используется, чтобы указать TortoiseSVN пути, для которых пометки и столбцы статуса *не показываются*. Это может пригодиться, если у вас имеется несколько очень больших рабочих копий, содержащих только библиотеки, которые вы вообще не собираетесь изменять, и поэтому им пометки не нужны. Например:

`f:\development\SVN\Subversion` отключит пометки *только* в этой конкретной папке. Вы по-прежнему будете видеть пометки на всех папках и файлах внутри этой папки.

`f:\development\SVN\Subversion*` отключит пометки *на всех* файлах и папках, чьи пути начинаются с `f:\development\SVN\Subversion`. Это означает, что пометки не будут показываться для всех файлов и папок далее этого пути.

То же самое относится к Включить пути, за исключением того, что для этих путей пометки показываются, даже если пометки были отключены или для этого конкретного типа диска, или при помощи указанного выше исключения путей.

Пользователи иногда спрашивают, как взаимодействуют эти три настройки, и точный ответ будет таким:

```
если (путь в списке включаемых)
  показывать пометки
если (путь на разрешённом типе диска) И (путь не в списке исключаемых)
  показывать пометки
```

Пометки для списка включённых путей показываются *всегда*. Иначе, пометки показываются для всех отмеченных типов дисков, *кроме* исключённых путей.

TSVNCache.exe также использует эти пути для того, чтобы ограничить объём сканирования. Если вы желаете просматривать только определённые папки, запретите все типы дисков и включите только те папки, которые должны сканироваться особо.

Исключение дисков, созданных при помощи `SUBST`

Довольно часто удобно использовать созданный при помощи `SUBST` диск для доступа к вашим рабочим копиям, например, применяя команду

```
subst T: C:\TortoiseSVN\trunk\doc
```

Однако, это может привести к тому, что пометки не будут обновляться, поскольку `TSVNCache` будет получать одно оповещение при изменении файла, и обычно это исходный путь. Это означает, что пометки по `subst`-пути могут никогда не обновиться.

Простой способ обойти эту проблему - исключить отображение пометок по исходному пути, чтобы вместо этого пометки показывались по `subst`-пути.

Иногда, производя исключение областей, содержащих рабочие копии, что избавляет TSVNCache от сканирования и мониторинга изменений, вы, тем не менее, желали бы получать наглядное указание на то, что эти папки версированы. Флажок Показывать исключённые папки как нормальные позволяет вам этого достичь: с его помощью версированные папки в любой исключённой области (тип устройства не отмечен, либо специально исключён) будут показываться как нормальные и не устаревшие, с зелёной галочкой. Это напомнит вам, что вы наблюдаете рабочую копию, хотя пометки на папках могут быть и неправильными. Пометки для файлов вообще не показываются. Обратите внимание: контекстные меню всё же работают, даже если пометки не показываются.

В качестве отдельного исключения, диски `A:` и `B:` никогда не учитываются опцией Показывать исключённые папки как нормальные. Это происходит из-за того, что Windows приходиться обращаться к диску, что может привести к задержке в несколько секунд при запуске Проводника, даже если в вашем ПК есть накопитель на гибких магнитных дисках.

4.30.3.1. Выбор набора значков

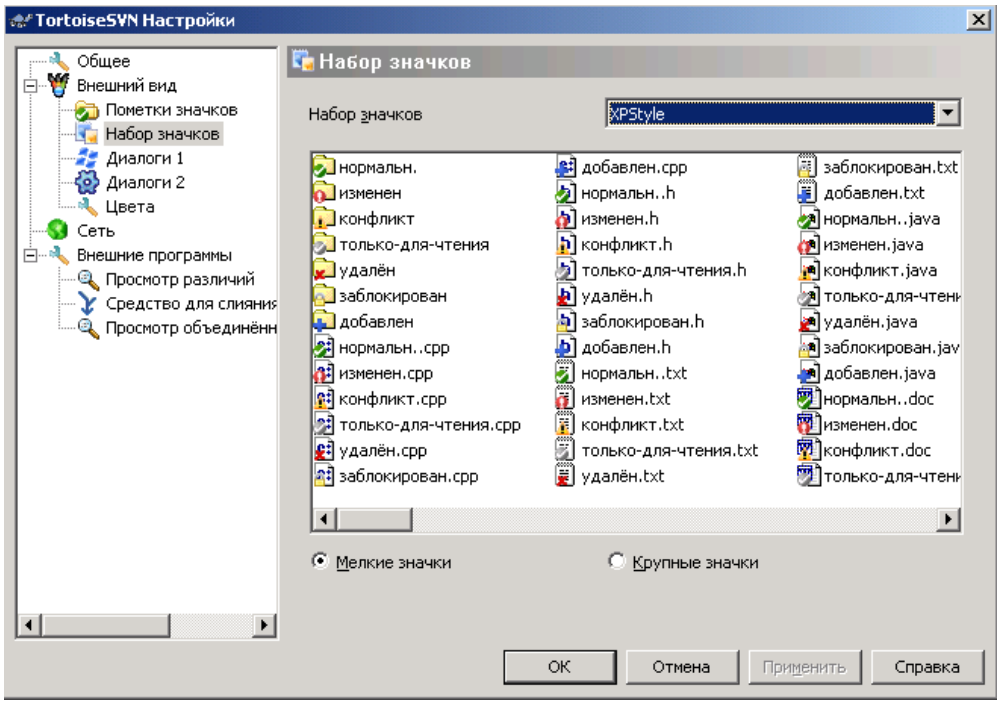


Рисунок 4.59. Страница 'Набор значков' в диалоге настроек

Вы можете изменить набор пометок на тот, который вам больше нравится. Обратите внимание: при изменении набора значков вам, возможно, понадобиться перезагрузить компьютер для того чтобы изменения вступили в силу.

4.30.4. Настройки сети

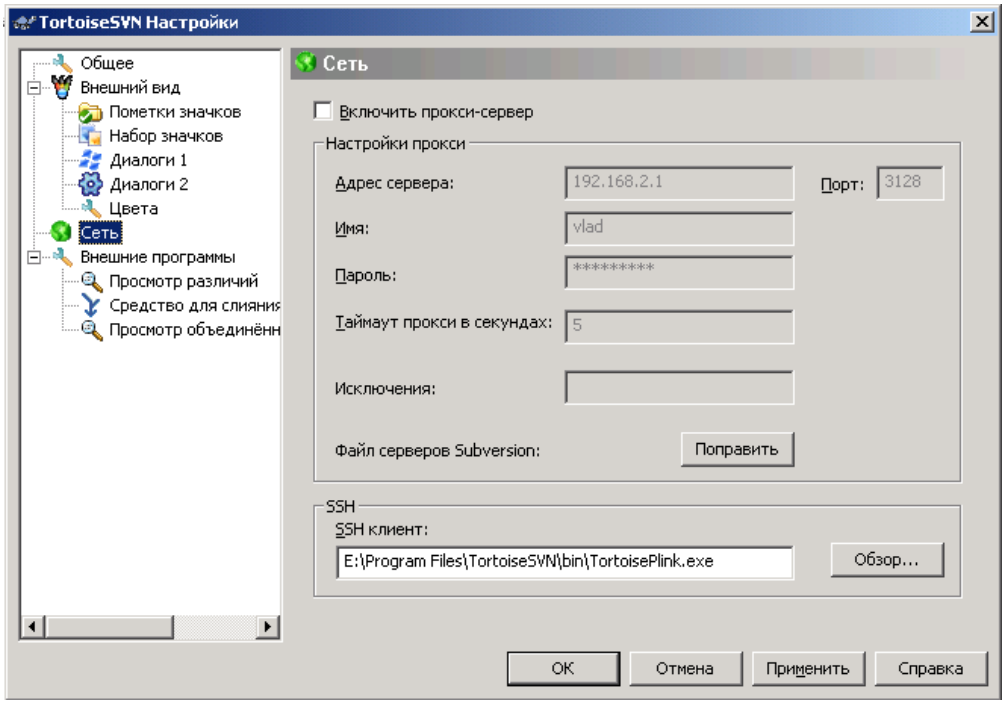


Рисунок 4.60. Страница 'Сеть' в диалоге настроек

Здесь вы можете настроить ваш сервер-посредник, если вам необходимо пройти через межсетевой экран вашей компании.

Если вам необходимо задать настройки серверов-посредников отдельно для каждого хранилища, то для такой конфигурации вам надо будет воспользоваться файлом `servers` Subversion. Получить доступ к нему можно при помощи кнопки Поправить. Обратитесь к разделу *Область настроек времени выполнения (Runtime Configuration Area)* [http://svnbook.red-bean.com/en/1.5/svn.advanced.confarea.html] книги о Subversion для получения подробной информации об использовании этого файла.

Вы также можете указать, какую программу должен использовать TortoiseSVN для установки безопасного соединения с хранилищем по svn+ssh. Мы рекомендуем использовать TortoisePlink.exe. Это версия популярной программы Plink, идущая вместе с TortoiseSVN, только она скомпилирована как безоконное приложение, и благодаря чему окна DOS не будут выскакивать при каждой аутентификации.

Вы должны указать полный путь к выполняемому файлу. Для TortoisePlink.exe это стандартная папка `bin` TortoiseSVN. Воспользуйтесь кнопкой Обзор..., чтобы его найти. Обратите внимание: если путь содержит пробелы, вы должны заключить его в кавычки, например,

```
"C:\Program Files\TortoiseSVN\bin\TortoisePlink.exe"
```

Один побочный эффект безоконного приложения в том, что сообщениям об ошибках негде отображаться, и, если аутентификация завершается неудачей, вы просто получаете сообщение, говорящее что-то вроде «Unable to write to standard output (Невозможно записать в стандартный вывод)». Поэтому мы рекомендуем, чтобы вы сначала всё настроили с использованием стандартного Plink. Затем, когда всё заработает, вы сможете использовать TortoisePlink с точно такими же параметрами.

Для TortoisePlink нет никакой документации, поскольку это только незначительно изменённый вариант Plink. Узнать о параметрах командной строки можно на *Web-сайте PuTTY* [http://www.chiark.greenend.org.uk/~sgtatham/putty/].

Чтобы избежать постоянного запроса пароля, вы можете также рассмотреть применение инструмента кэширования паролей, такого как Pageant. Его также можно скачать с веб-сайта PuTTY.

В заключение, настройка SSH на сервере и клиентах является нетривиальным процессом, выходящим за рамки этой документации. Однако, вы можете найти инструкции в ЧаВо (FAQ) TortoiseSVN в разделе *Subversion/TortoiseSVN SSH How-To* [http://tortoisesvn.net/ssh_howto].

4.30.5. Настройки внешних программ

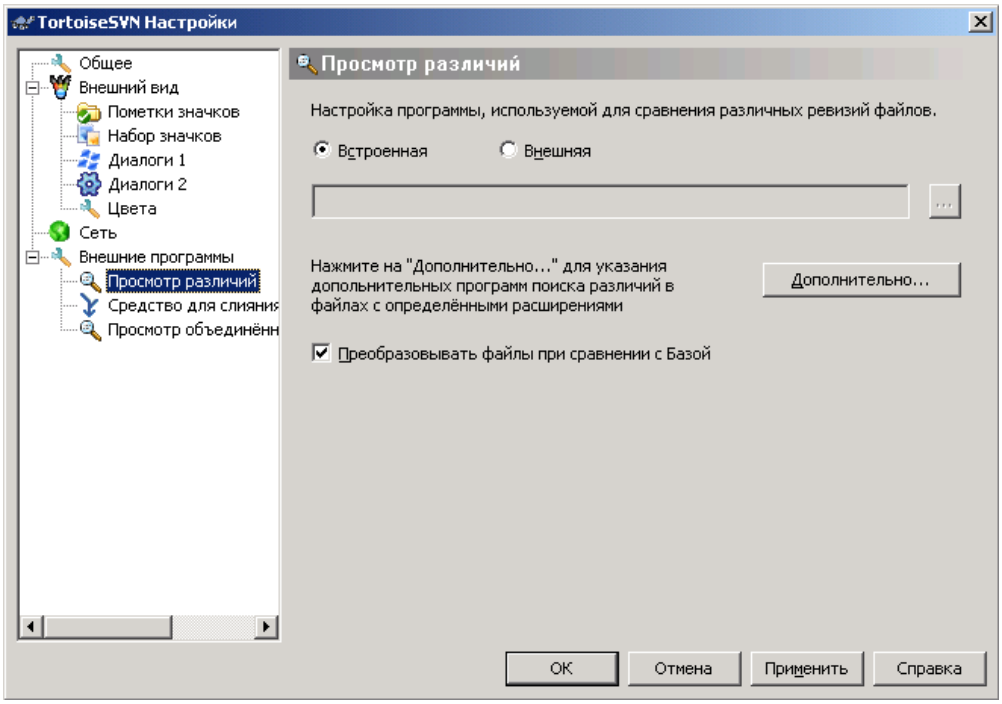


Рисунок 4.61. Страница 'Просмотр различий' в диалоге настроек

Здесь вы можете задать программу для проведения сравнения/слияния, которую будет использовать TortoiseSVN. По умолчанию используется TortoiseMerge, устанавливаемая совместно с TortoiseSVN.

Прочтите Раздел 4.10.5, «Внешние инструменты просмотра различий/слияния», где содержится список некоторых внешних программ сравнения/слияния, которые пользователи используют совместно с TortoiseSVN.

4.30.5.1. Программа просмотра различий

Для сравнения различных ревизий файлов может быть использована внешняя программа сравнения. Необходимо, чтобы эта внешняя программа могла получать имена файлов из командной строки, также как и все другие параметры. В TortoiseSVN используются замещаемые параметры, начинающиеся с `%`; когда TortoiseSVN встречает один из них, он заменяет его на соответствующее значение. Порядок параметров будет зависеть от используемой вами программы сравнения.

%base
: Исходный файл без ваших изменений

%bname
: Заголовок окна для базового файла

%mine
Ваш собственный файл, с вашими изменениями

%yname
Заголовок окна для вашего файла

В заголовках окон отображаются не только имена файлов. TortoiseSVN рассматривает их как имена для отображения, и формирует их в соответствии с этим. Поэтому, например, если вы выполняете сравнение файла ревизии 123 с файлом из вашей рабочей копии, имена будут`имя_файла : ревизия 123` и `имя_файла : рабочая копия`

Например, для ExamDiff Pro:

```
C:\путь\к\ExamDiff.exe %base %mine --left_display_name:%bname
    --right_display_name:%yname
```

или для KDiff3:

```
C:\путь\к\kdiff3.exe %base %mine --L1 %bname --L2 %yname
```

или для WinMerge:

```
C:\путь\к\WinMerge.exe -e -ub -dl %bname -dr %yname %base %mine
```

или для Araxis:

```
C:\путь\к\compare.exe /max /wait /title1:%bname /title2:%yname
        %base %mine
```

Если вы используете свойство `svn:keywords` для подстановки ключевых слов, и, особенно, ревизию файла (ключевое слово `revision`), то могут быть различия между файлами, появившееся только из-за текущего значения ключевого слова. Также, если вы используете `svn:eol-style = native`, то в базовом файле завершения строк будут `LF`, тогда как в вашем файле - `CR-LF`. TortoiseSVN обычно автоматически скрывает эти отличия, предварительно обрабатывая базовый файл и подставляя ключевые слова и завершения строк перед выполнением операции сравнения. Однако, это может занять длительное время с большими файлами. TortoiseSVN будет выполнять предобработку файлов, если отмечен флажок Преобразовывать файлы при сравнении с базой.

Вы также можете указать другой инструмент сравнения, применяемый для свойств Subversion. Поскольку свойства обычно являются простыми короткими текстовыми строками, вы можете использовать для просмотра более простой и компактный инструмент просмотра.

Если вы настроили альтернативный инструмент сравнения, вы можете использовать TortoiseMerge *и* сторонний инструмент из контекстных меню. Контекстное меню → Различия запускает основной сконфигурированный инструмент, а **Shift**+Контекстное меню → Различия - дополнительный.

4.30.5.2. Средство для слияния

Внешняя программа слияния, используемая для улаживания конфликтующих файлов. Замещение параметров применяется таким же образом, как и с программой сравнения.

%base
исходный файл без ваших или чужих изменений

%bname
: Заголовок окна для базового файла

%mine
: ваш собственный файл, с вашими изменениями

%yname
: Заголовок окна для вашего файла

%theirs
: файл, каков он в хранилище

%tname
: заголовок окна для файла из хранилища

%merged
: конфликтующий файл, результат операции слияния

%mname
: заголовок окна для объединённого файла

Например, для Perforce Merge:

```
C:\путь\к\P4Merge.exe %base %theirs %mine %merged
```

или для KDiff3:

```
C:\путь\к\kdiff3.exe %base %mine %theirs -o %merged
    --L1 %bname --L2 %yname --L3 %tname
```

или для Araxis:

```
C:\путь\к\compare.exe /max /wait /3 /title1:%tname /title2:%bname
    /title3:%yname %theirs %base %mine %merged /a2
```

или для WinMerge (2.8 или более поздней):

```
C:\путь\к\WinMerge.exe %merged
```

4.30.5.3. Дополнительные настройки сравнения/слияния

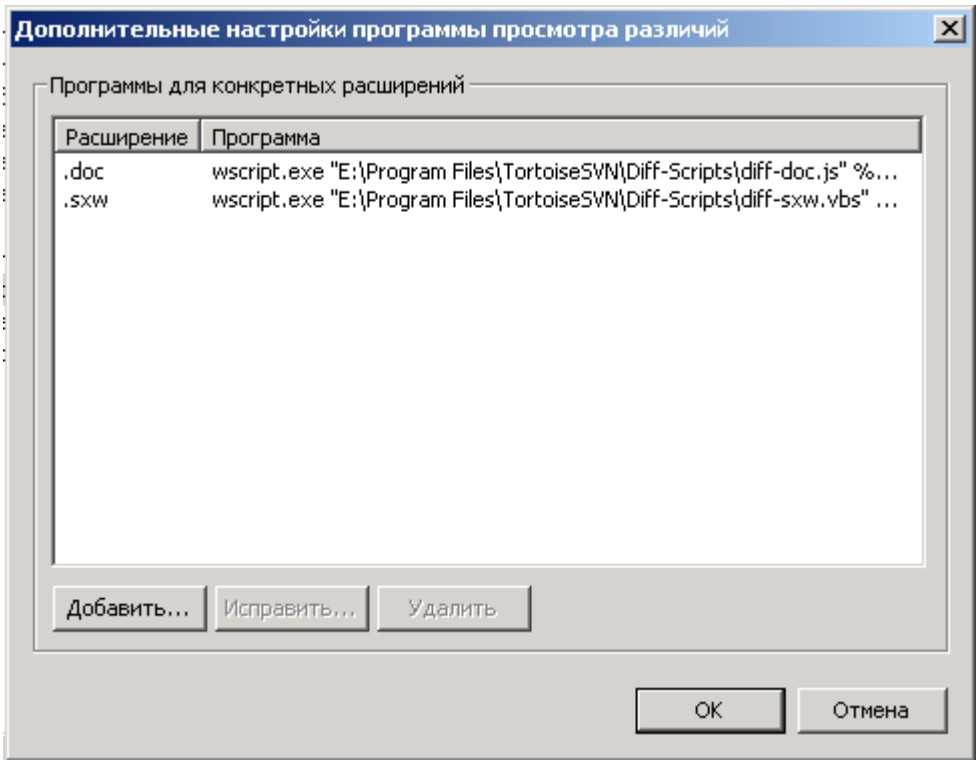


Рисунок 4.62. Окно дополнительных настроек сравнения/слияния в диалоге настроек

В дополнительных настройках вы можете задать различные программы сравнения и слияния для каждого расширения файла. Например, вы можете указать Photoshop как программу «сравнения» для файлов `.jpg` :-) Вы также можете связать свойство `svn:mime-type` с программой сравнения или слияния.

Для задания связи по расширению файла, вам необходимо указать это расширение. Используйте `.bmp` для указания файлов картинок Windows. Для задания связи по свойству `svn:mime-type`, укажите mime-тип, включая косую черту, например `text/xml`.

4.30.5.4. Просмотр объединённых различий

Программа просмотра для файлов объединённых различий (файлов заплаток). Параметры не требуются. При выборе опции По умолчанию сначала производится попытка запуска программы, с которой ассоциированы файлы с расширением `.diff`, затем проверяется `.txt`. Если у вас нет программы просмотра для `.diff`- файлов, то, скорее всего, запустится Блокнот.

Блокнот, изначально входящий в поставку Windows, не очень хорошо работает с файлами, у которых завершения строк отличаются от стандартного CR-LF. Поскольку большинство файлов объединённых различий имеют только LF-завершения строк, они не будут хорошо смотреться в Блокноте. Однако, вы можете загрузить бесплатную замену Блокнота *Notepad2* [http://www.flos-freeware.ch/notepad2.html], который не только правильно показывает завершения строк, но также подсвечивает цветом добавленные и удалённые строки.

4.30.6. Настройки сохранённых данных

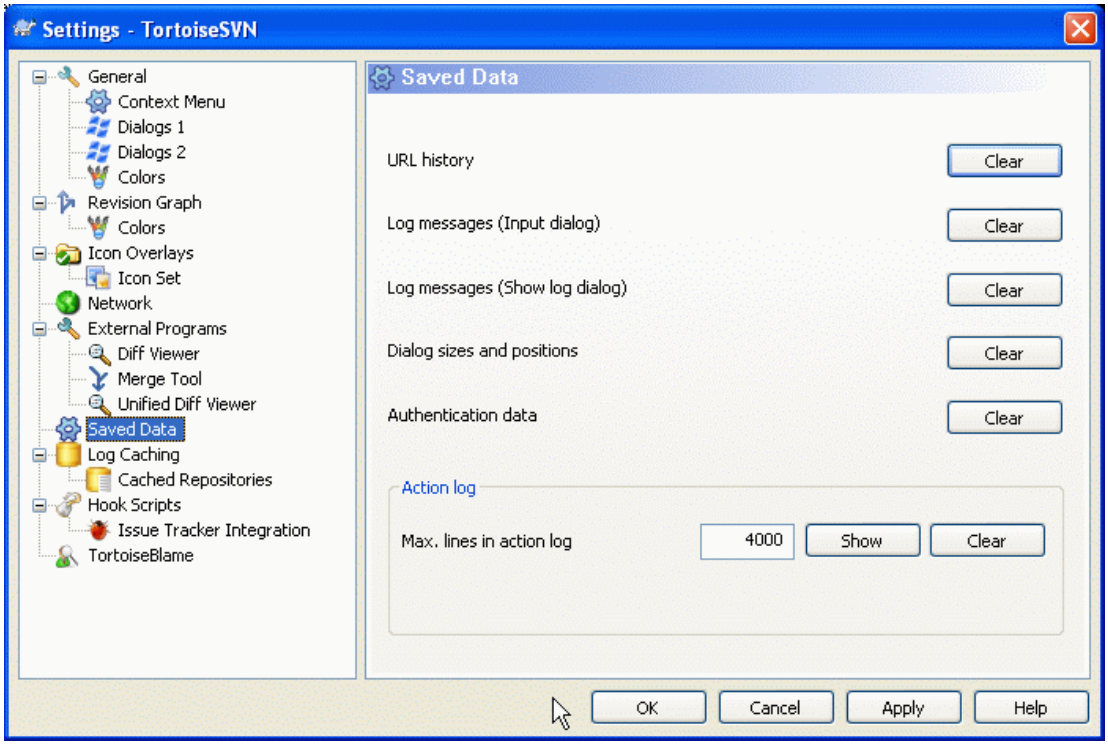


Рисунок 4.63. Страница 'Сохранённые данные' в диалоге настроек

Для вашего удобства TortoiseSVN сохраняет многие из используемых вами настроек, а также запоминает, что вы недавно посещали. Если вы желаете полностью очистить закэшированные данные, вы можете сделать это здесь.

Предыдущие URL
: При каждом извлечении рабочей копии, слиянии изменений и использовании обозревателя хранилища, TortoiseSVN запоминает последние использованные URL и предлагает их в выпадающем списке. Иногда этот список захламляется устаревшими адресами, и бывает полезно периодически его очищать.

  Если вы желаете убрать один из пунктов в каком-нибудь из выпадающих списков, вы можете сделать это прямо в нём. Только щёлкните на стрелке, чтобы открылся список, наведите курсор мыши на тот пункт который вы желаете убрать и нажмите **Shift+Delete**.

Введённые предыдущие сообщения журнала
: TortoiseSVN сохраняет последние введённые вами сообщения журнала. Они сохраняются для каждого хранилища, поэтому, если вы работаете со многими хранилищами, этот список может стать довольно большим.

Полученные кэшированные сообщения журнала
: TortoiseSVN кэширует сообщения журнала, получаемые в диалоге журнала, чтобы сэкономить время при следующем показе журнала. Если кто-либо другой отредактирует уже закэшированное у вас сообщение журнала, вы не увидите это изменение, пока не очистите кэш. Кэширование сообщений журнала включается на закладке Кэширование журнала.

Размеры и расположение окон
: Многие окна запоминают свои размеры и позицию на экране, какие были у них при последнем использовании.

Данные аутентификации
: При вашей аутентификации на сервере Subversion имя пользователя и пароль кэшируются локально, чтобы вам не приходилось вводить их снова и снова. Вы можете пожелать очистить

этот кэш из соображений безопасности, или из-за того, что вы желаете подключится к хранилищу под другим пользователем... а коллега знает, что вы используете его персональный компьютер?

Если вы желаете очистить данные аутентификации только для одного отдельного сервера, прочтите Раздел 4.1.5, «Аутентификация», чтобы узнать, как найти закэшированные данные.

Журнал действий
TortoiseSVN ведёт журнал всего, записываемого в его диалогах выполнения. Это может пригодиться когда, например, вы пожелаете проверить, что происходило в последней команде обновления.

Файл журнала ограничен в размере и, когда он становится слишком большим, наиболее старое содержимое отбрасывается. По умолчанию хранится 4000 строк, но вы можете настроить это число.

Отсюда вы можете посмотреть содержимое файла журнала, а также очистить его.

4.30.7. Кэширование журнала

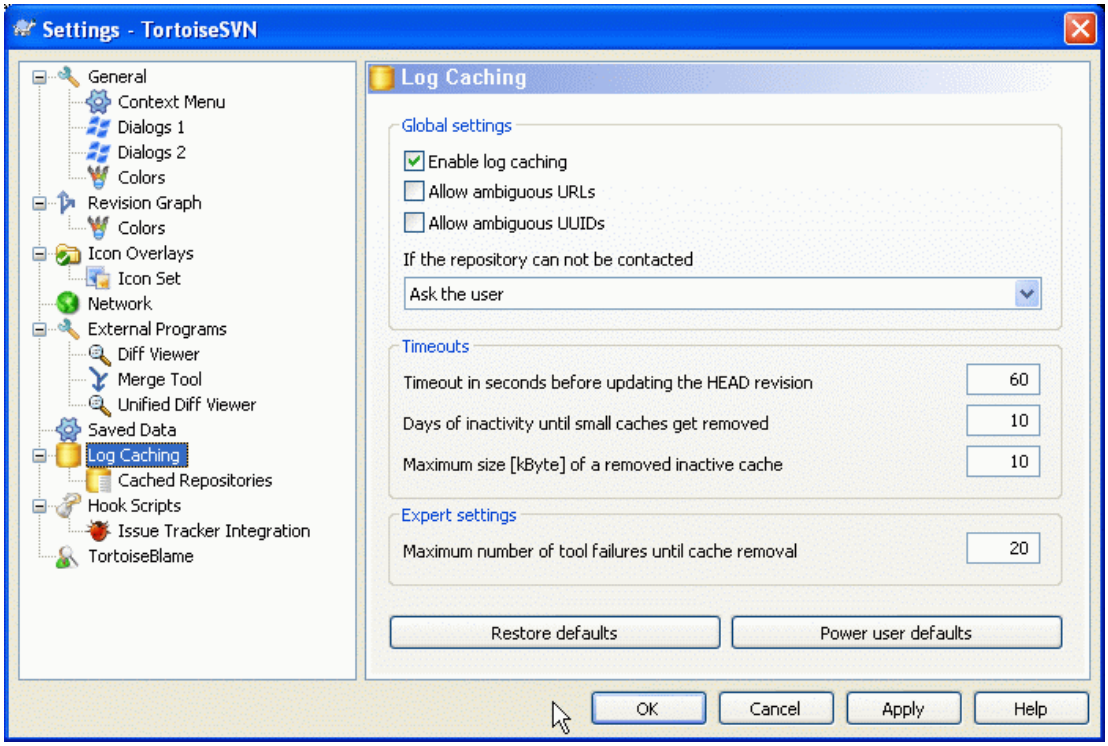


Рисунок 4.64. Страница 'Кэширование журнала' в диалоге настроек

В этом диалоге можно настроить кэширование сообщений журнала в TortoiseSVN, сохраняющее копию сообщений журнала и изменённых путей локально, чтобы избежать занимающих много времени загрузок с сервера. Использование кэширования может значительно ускорить работу диалога журнала и графа ревизий. Другой полезной возможностью является то, что сообщения журнала могут быть доступны при отсутствии подключения к сети, в автономном режиме.

Включить кэширование журнала
Включает кэширование полученных сообщений при получении данных журнала. Если отмечено, данные будут извлечены из кэша, если они там есть, а все незакэшированные сообщения будут получены с сервера и добавлены в кэш.

Если кэширование отключено, данные всегда будут запрашиваться непосредственно с сервера и не будут сохраняться локально.

Допускать неоднозначные URL
: Иногда вам нужно связаться с сервером, который использует один и тот же URL для всех хранилищ. Таковы, например, старые версии `svnbridge`. Если вам необходим доступ к таким хранилищам, вы должны отметить этот флажок. Если не нужен, оставьте его неотмеченным для улучшения производительности.

Допускать неоднозначные UUID
: Некоторые службы размещения сайтов дают всем своим хранилищам одн и тот же UUID. Вы могли сделать это и сами, скопировав папку хранилища при создании нового. В любом случае, это плохая идея - UUID должен быть *уникальным*. Однако, кэш журнала всё равно будет работать в такой ситуации, если отметить этот флажок. Если вам это не нужно, оставьте его неотмеченным для улучшения производительности.

Если невозможно соединиться с хранилищем
: Если вы работаете автономно, или если сервер хранилища не функционирует, то кэш журнала по-прежнему можно использовать для получения сообщений журнала, уже содержащихся в кэше. Конечно же, кэш может быть устаревшим, поэтому есть возможность выбрать, использовать ли эту функциональность.

  Если данные журнала берутся из кэша (без соединения с сервером), то в заголовок окна при использовании этих сообщений будет добавлена информация о работе в автономном режиме.

Таймаут перед обновлением ведущей ревизии (сек)
: Обычно, при вызове диалога журнала, желательно соединиться с сервером, чтобы посмотреть, не появились ли новые сообщения журнала. Если здесь задан ненулевой таймаут, то соединение с сервером будет производиться только по истечении времени таймаута с момента последнего запроса. Это может уменьшить количество обращений (с ожиданием ответа) к серверу, если вы часто открываете диалог журнала для медленно работающего сервера, но данные могут быть не всегда наиболее актуальными. Если вы желаете использовать эту возможность, мы предлагаем использовать для таймаута значение 300 (5 минут) в качестве компромисса.

Удалять неиспользуемые небольшие кэши через (дни)
: Если вы просматриваете множество хранилищ, у вас накопится множество кэшей журналов. Если вы их не используете активно, то их кэши большими не вырастут, и TortoiseSVN по умолчанию очистит их после заданного времени. Используйте этот параметр для управления очисткой кэшей.

Максимальный размер удаляемого неактивного кэша
: Более крупные кэши тяжелее накопить вновь, поэтому TortoiseSVN удаляет только маленькие кэши. Тонко настроить этот порог можно при помощи этого значения.

Максимальное количество отказов до удаления кэша
: Изредка что-то может пойти не так с кэшированием и произойдёт сбой. Если это случается, кэш удаляется автоматически для предотвращения повторного возникновения проблемы. Если вы используете менее стабильные ночные сборки, вы можете выбрать, чтобы кэш в этом случае сохранялся.

4.30.7.1. Закэшированные хранилища

На этой странице расположен список закэшированных локально хранилищ с дисковым пространством, занятым под кэш. Если выбрать какое-либо из хранилищ, можно будет использовать кнопки, находящиеся под списком.

Щёлкните на кнопке Обновление, чтобы полностью обновить кэш и заполнить все существующие пропуски. Для больших хранилищ это может занять очень много времени, но может понадобиться, если вы собираетесь работать автономно, и вам необходим наиболее полный кэш.

Щёлкните на кнопке Экспорт для экспорта всего кэша в виде набора CSV-файлов. Это может пригодиться, если вы желаете обработать данные журнала во внешней программе, хотя это будет полезно в основном разработчикам.

Щёлкните на кнопке Удалить для удаления всех закэшированных данных для выбранных хранилищ. Это не отключает кэширование для этих хранилищ, и поэтому в следующий раз при запросе данных журнала будет создан новый кэш.

4.30.7.2. Статистика кэша журнала

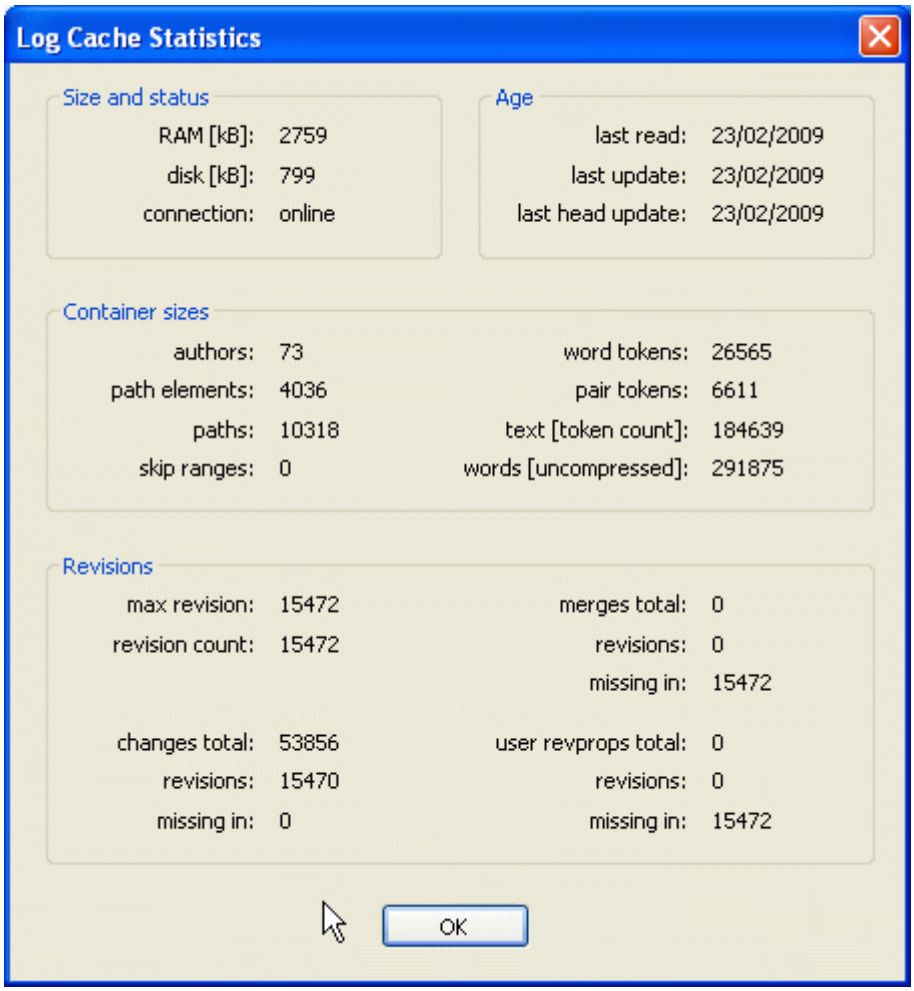


Рисунок 4.65. Окно 'Статистика кэша журнала', открываемое из диалога настроек

Щёлкните на кнопке Подробнее, чтобы увидеть подробную статистику для конкретного кэша. Многие из показываемых здесь полей предоставляет интерес в основном для разработчиков TortoiseSVN, поэтому не все из них описаны подробно.

ОЗУ
: Объём памяти, необходимой для обслуживания этого кэша.

Диск
: Объём дискового пространства, занятого закэшированными данными. Данные хранятся в сжатом виде, и поэтому использование диска обычно довольно скромное.

Соединение
: Показывает, было ли доступно хранилище при последнем использовании кэша.

Возраст: обновление
: Дата последнего изменения содержимого кэша.

Возраст: Ведущая ревизия
: Дата последнего запроса ведущей ревизии с сервера.

Авторы
: Количество различающихся авторов содержащихся в кэше сообщений.

Пути
: Количество зарегистрированных путей, которые были бы выведены при использовании `svn log -v`.

Пропущ.диапаз.
: Количество диапазонов неполученных ревизий, которые просто не были запрошены. Это показатель количества пропусков в кэше.

Max ревизия
: Наибольший номер ревизии, сохранённой в кэше.

Ревизий всего
: Количество ревизий, сохранённых в кэше. Это другой показатель полноты кэша.

4.30.8. Скрипты ловушек, выполняемые на стороне клиента

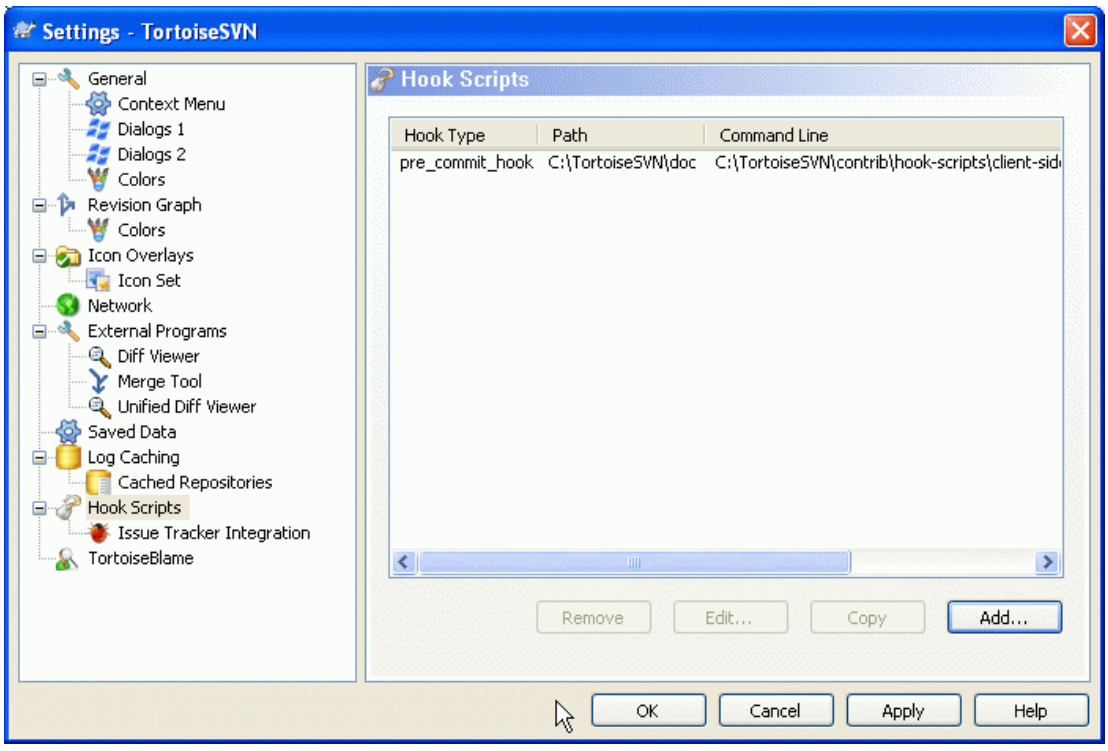


Рисунок 4.66. Страница 'Скрипты ловушек' в диалоге настроек

В этом диалоге можно настроить скрипты ловушек, которые будут запускаться автоматически при совершении определённых действий Subversion. В отличие от скриптов ловушек, описанных в Раздел 3.3, «Скрипты ловушек, выполняемые на стороне сервера», эти скрипты выполняются локально на клиенте.

Одним из применений для таких ловушек может быть запуск программы вроде `SubWCRev.exe` для обновления номеров версий после фиксации, и, возможно, для автоматического перезапуска сборки.

По различным соображениям безопасности и особенностям реализации, скрипты ловушек определяются на машине локально, а не как свойства проекта. То, что будет выполняться, задаёте вы, и это вне зависимости от того, что кто-либо другой может зафиксировать в хранилище. Конечно, вы всегда можете решить вызвать скрипт, который сам находится под управлением версиями.

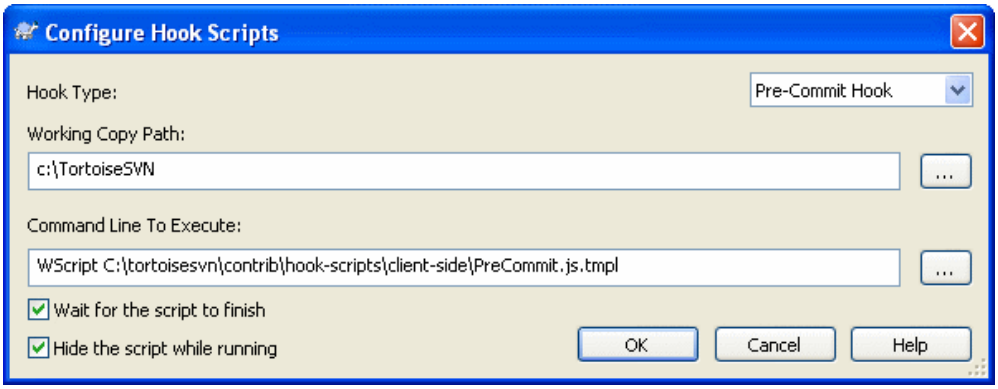


Рисунок 4.67. Окно 'Настройка скрипта ловушки', открываемое из диалога настроек

Чтобы добавить скрипт ловушки, просто щёлкните на Добавить... и заполните необходимые данные.

На данный момент доступно шесть типов скриптов ловушек:

Начало-фиксации
: Вызывается перед показом диалога фиксации. Может понадобиться, если ловушка изменяет версированный файл и оказывает воздействие на список файлов, которые должны быть зафиксированы и/или на сообщение фиксации. Однако, вы дожны учитывать, что из-за того, что ловушка вызывается на ранней стадии, полный список выбранных для фиксации объектов не доступен.

Перед-фиксацией
: Вызывается после того, как пользователь нажмёт кнопку OK в диалоге фиксации, но перед фактическим началом фиксации. Этой ловушке доступен точный список того, что будет фиксироваться.

После-фиксации
: Вызывается после окончания фиксации (независимо, успешного или нет).

Начало-обновления
: Вызывается перед отображением диалога обновить-до-ревизии.

Перед-обновлением
: Вызывается перед фактическим началом операции Subversion 'обновление'.

После-обновления
: Вызывается после окончания операции обновления (независимо, успешного или нет).

Ловушка определяется для конкретного пути в рабочей копии. Достаточно указать путь до папки верхнего уровня; при выполнении операции в её подпапке, TortoiseSVN автоматически производит поиск подходящего пути выше по иерархии[6].

Далее вы должны указать командную строку для выполнения, начав с пути до скрипта ловушки или до выполняемого файла. Можно использовать пакетный файл, выполняемый файл или любой

[6]Для обозначения корня иерархии используется символ * - прим. переводчика

другой файл, если для его типа существует корректное сопоставление в Windows, например, perl-скрипт.

Командная строка содержит несколько параметров, значения которых подставляются TortoiseSVN. Набор доступных для использования параметров зависит от вызываемой ловушки. У каждой ловушки есть свои собственные параметры, передаваемые в следующем порядке:

Начало-фиксации
`PATHMESSAGEFILECWD`

Перед-фиксацией
`PATHDEPTHMESSAGEFILECWD`

После-фиксации
`PATHDEPTHMESSAGEFILEREVISIONERRORCWD`

Начало-обновления
`PATHCWD`

Перед-обновлением
`PATHDEPTHREVISIONCWD`

После-обновления
`PATHDEPTHREVISIONERRORCWD`

Значение каждого из этих параметров описано здесь:

PATH
: Путь к временному файлу, содержащему все пути, для которых была запущена операция. Во временном файле каждый путь расположен в отдельной строке.

DEPTH
: Глубина охвата, с которой выполнялись фиксация/извлечение.

  Возможные значения:

  -2
  : `svn_depth_unknown (неизвестная глубина)`

  -1
  : `svn_depth_exclude (без спуска)`

  0
  : `svn_depth_empty (только этот элемент)`

  1
  : `svn_depth_files (только потомки-файлы)`

  2
  : `svn_depth_immediates (непосредственные потомки, включая папки)`

  3
  : `svn_depth_infinity (полностью рекурсивно)`

MESSAGEFILE
: Путь к файлу, содержащему сообщение журнала для фиксации. Файл содержит текст в кодировке UTF-8. После успешного выполнения ловушки начало-фиксации сообщение журнала считывается заново, благодаря чему ловушка получает возможность его изменить.

REVISION
: Ревизия в хранилище, до которой должно производиться обновление, или же номер ревизии после выполнения фиксации.

ERROR
: Путь к файлу, содержащему сообщение об ошибке. Если ошибки не было, файл будет пустым.

CWD
: Текущая рабочая папка, для которой запускается скрипт. Устанавливается в общую корневую папку всех затронутых путей.

Обратите внимание: хотя мы и указываем для удобства имена параметров, вам нет необходимости ссылаться на эти имена в настройках ловушек. Все параметры, перечисленные для каждой ловушки, передаются всегда, вне зависимости от того, нужны они вам или нет ;-)

Если вы желаете, чтобы операция Subversion откладывалась до окончания работы ловушки, отметьте Ждать окончания работы скрипта.

Обычно бывает желательно скрыть безобразные окна сеансов DOS, открывающиеся при работе скрипта, поэтому флажок Скрыть скрипт во время работы по умолчанию отмечен.

Примеры клиентских скриптов ловушек можно найти в папке `contrib` в *хранилище TortoiseSVN* [http://tortoisesvn.tigris.org/svn/tortoisesvn/trunk/contrib/hook-scripts]. (Раздел 3, «TortoiseSVN бесплатен!» рассказывает, как получить доступ к хранилищу исходного кода TortoiseSVN).

4.30.8.1. Интеграция с системами отслеживания проблем

TortoiseSVN может использовать подключаемый модуль COM для запроса систем отслеживания проблем из диалога фиксации. Использование таких подключаемых модулей описывает Раздел 4.28.2, «Получение информации из системы отслеживания проблем». Если ваш системный администратор предоставил вам такой модуль, который вы уже установили и зарегистрировали, это то место, где указывается, как он будет интегрирован с вашей рабочей копией.

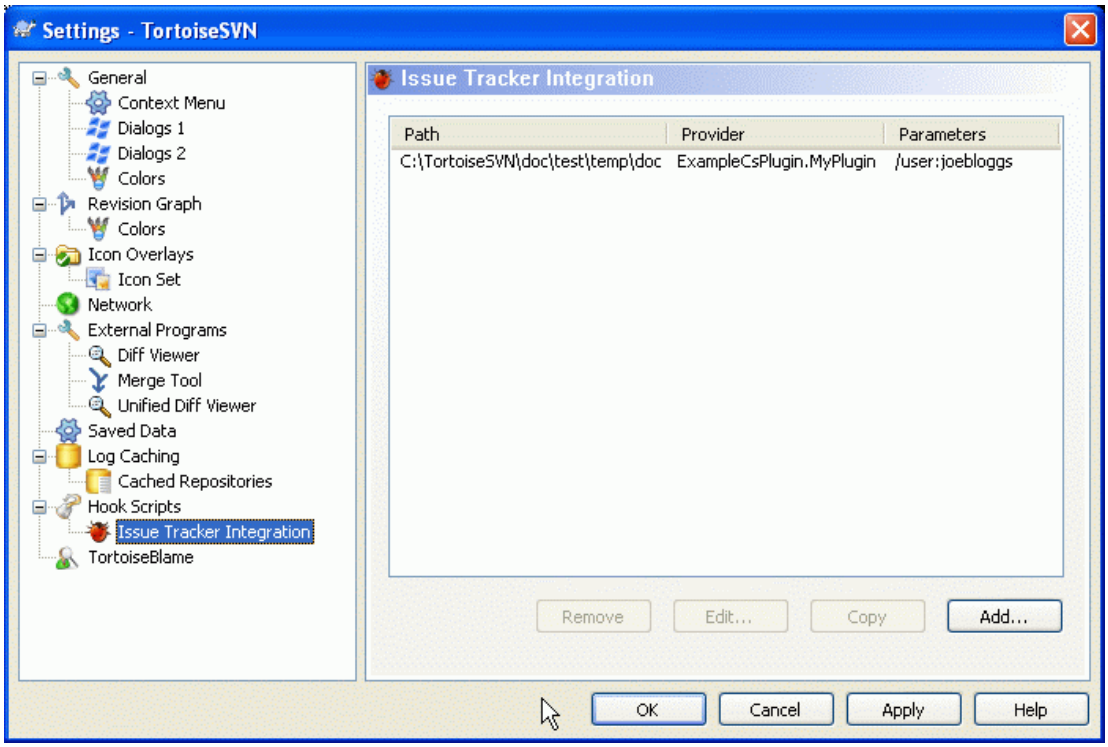


Рисунок 4.68. Страница интеграции с системой отслеживания проблем в диалоге настроек

Щёлкните на Добавить... для использования подключаемого модуля с конкретной рабочей копией. Здесь вы можете указать путь рабочей копии, выбрать из выпадающего списка, какой

подключаемый модуль использовать из всех зарегистрированных, и передаваемые параметры. Набор параметров будет зависеть от подключаемого модуля, но может включать ваше имя пользователя в системе отслеживания проблем, чтобы модуль мог запросить назначенные вам проблемы.

Если вы желаете, чтобы для вашего проекта все пользовали использовали один и тот же подключаемый модуль COM, вы можете указать его также при помощи свойств `bugtraq:provideruuid` и `bugtraq:providerparams`.

bugtraq:provideruuid
: Это свойство определяет COM UUID компонента IBugtraqProvider, например `{91974081-2DC7-4FB1-B3BE-0DE1C8D6CE4E}`. (этот пример - UUID *Поставщика системы отслеживания ошибок Gurtle* [http://code.google.com/p/gurtle/], являющегося поставщиком для системы отслеживания проблем *Google Code* [http://code.google.com/hosting/]).

bugtraq:providerparams
: Это свойство задаёт параметры, передаваемые IBugtraqProvider.

Пожалуйста, загляните в документацию к вашему подключаемому модулю IBugtraqProvider, чтобы узнать, что указывать в этих двух свойствах.

4.30.9. Настройки TortoiseBlame

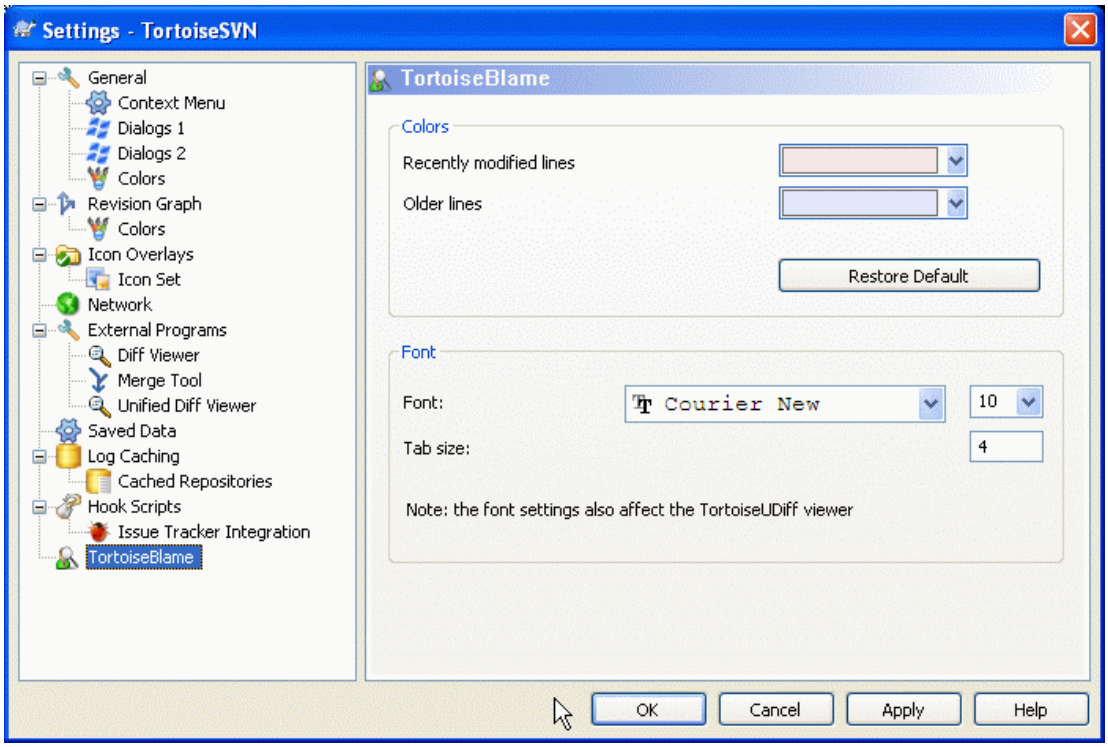


Рисунок 4.69. Страница TortoiseBlame в диалоге настроек

Настройки, используемые TortoiseBlame, задаются через главное контекстное меню, а не из самой TortoiseBlame.

Цвета
: TortoiseBlame может использовать цвет фона для обозначения возраста строк файла. Вы указываете крайние значения, задавая цвета для самых новых и самых старых ревизий, и TortoiseBlame применяет линейную интерполяцию между этими цветами в соответствии с ревизией из хранилища, указанной в каждой строке.

Шрифт
: Вы можете выбрать начертание и размер шрифта, используемого для отображения текста. Этот шрифт будет использован и для содержимого файла, и для информации об авторе и ревизии, показываемой в левой панели.

Размер табул.
: Определяет, сколько пробелов использовать для замены символов табуляции, обнаруженных в файле.

4.30.10. Настройки в реестре

Несколько редко используемых настроек доступны только путём непосредственного редактирования реестра. Безусловно, вы должны редактировать значения в реестре только в случае, если вы знаете, что вы делаете.

Настройка
: Вы можете указать другое месторасположение файла настроек Subversion при помощи ключа `HKCU\Software\TortoiseSVN\ConfigDir`. Это окажет воздействие на все операции TortoiseSVN.

Значок для кэша в трее
: Для добавления в трей значка для программы TSVNCache, создайте ключ `HKCU\Software\TortoiseSVN\CacheTrayIcon` типа `DWORD` со значением 1. На самом деле это пригодиться только разработчикам, поскольку позволяет элегантно завершать программу.

Отладка
: Для отображения параметров командной строки, передаваемых TortoiseProc.exe из расширения оболочки, создайте ключ `HKCU\Software\TortoiseSVN\Debug` типа `DWORD` со значением 1.

Значки в контекстном меню
: Это может пригодиться, если вы используете что-либо отличное от Проводника Windows или если у вас возникают проблемы с корректным отображением контекстного меню. Создайте ключ `HKCU\Software\TortoiseSVN\ShowContextMenuIcons` типа `DWORD` со значением 0, если вы желаете, чтобы TortoiseSVN не показывал значки для пунктов в контекстном меню оболочки. Чтобы значки показывались снова, установите это значение в 1.

Блокирование пометок статуса на значках
: Если вы желаете, чтобы в Проводнике не обновлялись пометки статуса на значках, пока выполняется другая команда TortoiseSVN (такая как Обновление, Фиксация, ...), то создайте ключ `HKCU\Software\TortoiseSVN\BlockStatus` типа `DWORD` со значением 1.

URL для проверки наличия обновений
: `HKCU\Software\TortoiseSVN\UpdateCheckURL` содержит URL, по которому TortoiseSVN пытается скачать текстовый файл, чтобы узнать, не появились ли обновления. Если желаете, вы можете также указать его в `HKLM` вместо `HKCU`, но `HKCU` имеет приоритет перед `HKLM`. Это может пригодится администраторам компаний, которым необходимо, чтобы пользователи не обновляли TortoiseSVN, пока это не будет санкционировано.

Имена файлов без расширений в списке автозавершения
: Список автозавершения, показываемый в редакторе сообщений фиксации, отображает имена файлов, предназначенных для фиксации. Для включения этих же имён, но без расширений, создайте ключ `HKCU\Software\TortoiseSVN\AutocompleteRemovesExtensions` типа `DWORD` со значением 1.

Колонки TortoiseSVN везде в Проводнике
: Дополнительные колонки, добавляемые TortoiseSVN в Проводнике Windows в режиме отображения 'таблица' обычно отображаются только в рабочей копии. Если вы желаете, чтобы

они были доступны везде, а не только в рабочих копиях, создайте ключ `HKCU\Software\TortoiseSVN\ColumnsEveryWhere` типа `DWORD` со значением 1.

Разделитель сообщений журнала при слиянии
: Когда выполняется слияние ревизий из другого ответвления, и доступна информация отслеживания слияний, сообщения журнала для сливаемых ревизий накапливаются, чтобы сформироваться в сообщение журнала при фиксации. Для разделения отдельных сообщений сливаемых ревизий используется предопределённая строка. При желании, вы можете создать ключ `HKCU\Software\TortoiseSVN\MergeLogSeparator` типа `SZ`, содержащий разделительную строку по вашему выбору.

Всегда получать авторство изменений при помощи TortoiseMerge
: TortoiseSVN позволяет указать внешнюю программу просмотра различий. Большинство таких программ, однако, не предназначено для отображения авторства изменений (Раздел 4.23.2, «Авторство различий»), и поэтому вы можете пожелать обратиться к TortoiseMerge в этом случае. Чтобы это сделать, создайте ключ `HKCU\Software\TortoiseSVN\DiffBlamesWithTortoiseMerge` типа `DWORD` со значением 1.

Подсветка текущей ревизии для папок в диалоге журнала
: The log dialog highlights the current working copy revision when the log is shown for a file. To do the same thing for a folder requires a working copy crawl, which is the default action, but it can be a slow operation for large working copies. If you want to change the operation of this feature you must create a `DWORD` registry key at `HKCU\Software\TortoiseSVN\RecursiveLogRev`. A value of 0 disables the feature (no highlighting for folders), a value of 1 (default) will fetch the status recursively (find the highest revision in the working copy tree), and a value of 2 will check the revision of the selected folder itself, but will not check any child items.

Извлечение должно завершаться неудачей, если элемент с таким же именем уже существует
: По умолчанию, если вы извлекаете рабочую копию поверх уже существующей структуры неверсированных папок, как, возможно, вы делаете после импорта, то любое существующее, но отличающееся от хранилища содержимое, будет оставлено в прежнем виде и помечено как изменённое. Когда вы соберётесь фиксировать, ваша локальная копия будет отправлена обратно в хранилище. Некоторые предпочитают, чтобы извлечение закончилось неудачей, если уже существующее содержимое отличается, и в случае, когда два человека добавили один и тот же файл, версия второго не перезаписала первоначальную версию по ошибке. Если вы желаете, чтобы извлечение заканчивалось неудачей в этом случае, вы должны создать ключ `HKCU\Software\TortoiseSVN\AllowUnversionedObstruction` типа `DWORD` со значением 0.

4.30.11. Рабочие папки Subversion

VS.NET 2003 при использовании с веб-проектами не может обрабатывать папки `.svn`, используемые Subversion для хранения своей внутренней информации. Это не ошибка в Subversion, это ошибка в VS.NET 2003 и используемых им расширениях frontpage.

> Обратите внимание: эта ошибка была исправлена в VS 2005 и более поздних версиях.

Начиная с версии 1.3.0 Subversion и TortoiseSVN, вы можете установить переменную окружения `SVN_ASP_DOT_NET_HACK`. Если эта переменная установлена, тогда Subversion будет использовать папки `_svn` вместо папок `.svn`. Вы должны перезагрузить вашу оболочку для того, чтобы эта переменная возымела действие. Обычно это означает, что вы должны перезагрузить ПК. Для облегчения этой задачи, вы можете сделать это со страницы основных установок при помощи простого флажка - подробнее об этом рассказывает Раздел 4.30.1, «Общие настройки».

Для дополнительной информации, а также чтобы узнать о других способах изначального предотвращения этой проблемы, ознакомьтесь со статьёй об этом в нашем *ЧаВо (FAQ)* [http://tortoisesvn.net/aspdotnethack].

4.31. Последний шаг

Вознаграждение

Хотя TortoiseSVN и TortoiseMerge и бесплатны, вы можете посодействовать разработчикам, присылая заплатки и активно участвуя в разработке. Вы также можете помочь нам, подбодрив нас в то нескончаемое время, которое мы проводим за нашими компьютерами.

Во время работы над TortoiseSVN мы любим слушать музыку. И так как мы затрачиваем много времени на проект, нам надо *много* музыки. Поэтому мы составили списки пожеланий с нашими любимыми музыкальными CD и DVD: *http://tortoisesvn.tigris.org/donate.html* Пожалуйста, взгляните также и на список людей, которые внесли вклад в проект, присылая исправления или переводы.

Глава 5. Программа SubWCRev

SubWCRev - это консольная программа Windows, которая может быть использована для чтения статуса рабочей копии Subversion и, при необходимости, для выполнения подстановки ключевых слов в шаблонных файлах. Это часто используется как часть процесса сборки, в качестве средства для внедрения информации из рабочей копии в собираемый объект. В основном это применяется для включения номера ревизии в диалог «О программе».

5.1. Командная строка SubWCRev

По умолчанию, SubWCRev считывает статус Subversion для всех файлов в рабочей копии, за исключением внешних включений. Она запоминает наибольший найденный номер зафиксированной ревизии, дату и время фиксации, а также регистрирует, были ли локальные изменения в рабочей копии, или смешанные обновления ревизий. Номер ревизии, диапазон обновлений ревизий и статус изменений отображается в стандартный вывод (stdout).

SubWCRev.exe вызывается из командной строки или скрипта, и управляется при помощи параметров командной строки.

```
SubWCRev ПутьКРабочейКопии [Файл_Исходной_Версии Файл_Целевой_Версии] [-nmdfe]
```

`ПутьКРабочейКопии` - это путь к проверяемой рабочей копии. Вы можете вызывать SubWCRev только для рабочих копий, но не можете непосредственно для хранилища . Путь может быть как абсолютным, так и относительным к текущей рабочей папке.

Если вы желаете, чтобы SubWCRev выполняла подстановку ключевых слов, чтобы поля вроде ревизии из хранилища и адреса URL сохранялись в текстовый файл, вы должны предоставить шаблонный `Файл_Исходной_Версии` и выходной `Файл_Целевой_Версии`, который будет содержать версию шаблона с произведёнными подстановками.

Есть несколько необязательных параметров, которые влияют на работу SubWCRev. При использовании более одного параметра, они должны указываться в виде единой группы, например `-nm`, а не `-n -m`.

Параметр	Описание
-n	Если указан данный параметр, SubWCRev будет завершаться с `ERRORLEVEL 7`, если рабочая копия содержит локальные изменения. Это может быть использовано для предотвращения сборки в случае наличия незафиксированных изменений.
-m	Если указан этот параметр, SubWCRev будет завершаться с `ERRORLEVEL 8`, если рабочая копия содержит смешанные ревизии. Это может использоваться для предотвращения сборки в случае частично обновленной рабочей копии.
-d	Если указан этот параметр, SubWCRev будет завершаться с `ERRORLEVEL 9`, если целевой файл уже существует.
-f	Если указан этот параметр, SubWCRev будет включать ревизию последнего изменения папки. Поведение по умолчанию - использовать при получении номеров ревизий только файлы.
-e	Если указан этот параметр, SubWCRev будет проверять папки, включённые при помощи `svn:externals`, но только если они из того же хранилища. Поведение по умолчанию - игнорировать внешние включения.
-x	Если указан этот параметр, SubWCRev будет выводить номера ревизий в шестнадцатеричном виде.

Параметр	Описание
-X	Если указан этот параметр, SubWCRev будет выводить номера ревизий в шестнадцатеричном виде, с префиксом '0X'.

Таблица 5.1. Список доступных параметров командной строки

5.2. Подстановка ключевых слов

Если указаны исходный и целевой файлы, SubWCRev копирует исходный файл в целевой, выполняя подстановку ключевых слов следующим образом:

Ключевое слово	Описание
$WCREV$	Заменяется на наибольшую зафиксированную ревизию в рабочей копии.
$WCDATE$	Заменяется на дату/время фиксации наибольшей зафиксированной ревизии. По умолчанию, используется международный формат: `yyyy-mm-dd hh:mm:ss`. Или же вы можете указать собственный формат, который будет использован с `strftime()`, например: `$WCDATE=%a %b %d %I:%M:%S %p$`. Список доступных символов форматирования можно узнать в *онлайн-справочнике* [http://www.cppreference.com/stddate/strftime.html].
$WCNOW$	Заменяется на текущую системную дату/время. Может быть использовано для указания времени сборки. Может быть использован формат даты/времени, описанный для `$WCDATE$`.
$WCRANGE$	Заменяется на диапазон обновлений ревизий в рабочей копии. Если рабочая копия в согласованном состоянии, это будет одна ревизия. Если рабочая копия содержит смешанные ревизии вследствие устаревания, или из-за намеренного обновления-до-определённой-ревизии, то диапазон будет показан в виде 100:200
$WCMIXED$	`$WCMIXED?TText:FText$` заменяется на `TText`, если есть смешанные обновления ревизий, или на `FText`, если нет.
$WCMODS$	`$WCMODS?TText:FText$` заменяется на `TText`, если были локальные изменения, или на `FText`, если не было.
$WCURL$	Заменяется на URL хранилища той рабочей копии, путь к которой был передан SubWCRev.
$WCINSVN$	`$WCINSVN?TText:FText$` заменяется на `TText`, если элемент версирован, или на `FText`, если нет.
$WCNEEDSLOCK$	`$WCNEEDSLOCK?TText:FText$` заменяется на `TText`, если у элемента установлено свойство `svn:needs-lock`, или на `FText`, если нет.
$WCISLOCKED$	`$WCISLOCKED?TText:FText$` заменяется на `TText` если элемент заблокирован, или на `FText`, если нет.
$WCLOCKDATE$	Заменяется на дату блокировки. Может быть использован формат даты/времени, описанный для `$WCDATE$`.
$WCLOCKOWNER$	Заменяется на имя владельца блокировки.
$WCLOCKCOMMENT$	Заменяется на комментарий блокировки.

Таблица 5.2. Список доступных параметров командной строки

Подсказка

Некоторые их этих ключевых слов применяются к отдельным файлам, а не ко всей рабочей копии, поэтому имеет смысл использовать их, когда SubWCRev вызывается для сканирования одного файла. Это относится к `$WCINSVN$`, `$WCNEEDSLOCK$`, `$WCISLOCKED$`, `$WCLOCKDATE$`, `$WCLOCKOWNER$` и `$WCLOCKCOMMENT$`.

5.3. Пример для ключевых слов

Нижеприведённый пример показывает, как происходит замена ключевых слов при переходе от шаблонного файла к целевому.

```
// Пробный файл для SubWCRev: testfile.tmpl

char *Revision = "$WCREV$";
char *Modified = "$WCMODS?Изменён:Не изменён$";
char *Date     = "$WCDATE$";
char *RevRange = "$WCRANGE$";
char *Mixed    = "$WCMIXED?Есть смешанные ревизии:Смешанных ревизий нет$";
char *URL      = "$WCURL$";

#if $WCMODS?1:0$
#error Исходный файл изменён
#endif

// КонецФайла
```

После запуска `SubWCRev.exe путь\к\рабочей\копии testfile.tmpl testfile.txt`, выходной файл `testfile.txt` будет выглядеть подобно этому:

```
// Пробный файл для SubWCRev: testfile.txt

char *Revision = "3701";
char *Modified = "Изменён";
char *Date     = "2005/06/15 11:15:12";
char *RevRange = "3699:3701";
char *Mixed    = "Есть смешанные ревизии";
char *URL      = "http://project.domain.org/svn/trunk/src";

#if 1
#error Исходный файл изменён
#endif

// Конец файла
```

Подсказка

Файл вроде этого будет включён в сборку, поэтому ожидается, что он будет версированным. Убедитесь, что версирован шаблонный файл, а не генерируемый, иначе каждый раз при повторной генерации файла версий вам надо будет фиксировать изменения, что, в свою очередь, означает, что файл версии необходимо обновить.

5.4. COM-интерфейс

Если вам необходимо получить доступ к информации Subversion о ревизиях из других программ, вы можете использовать COM-интерфейс SubWCRev. Объект, который необходимо создать - `SubWCRev.object`, и им поддерживаются следующие методы:

Метод	Описание
.GetWCInfo	Этот метод обходит рабочую копию, собирая информацию о ревизиях. Естественно, вы должны вызвать его до того, как в сможете обратиться к информации при помощи остальных методов. Первый параметр - путь. Второй параметр должен быть true, если вы желаете включить ревизии папок. Эквивалентен ключу командной строки `-f`. Третий параметр должен быть true, если вы желаете включить svn:externals. Эквивалентен ключу командной строки `-e`.
.Revision	Наибольшая зафиксированная ревизия в рабочей копии. Эквивалент `$WCREV$`
.Date	Дата/время фиксации наибольшей зафиксированной ревизии.Эквивалент `$WCDATE$`.
.Author	Автор наибольшей зафиксированной ревизии, т.е. последний человек, зафиксировавший изменения в рабочей копии.
.MinRev	Минимальная обновлённая ревизия, которая показывается в `$WCRANGE$`.
.MaxRev	Максимальная обновлённая ревизия, которая показывается в `$WCRANGE$`.
.HasModifications	True, если есть локальные изменения
.Url	Заменяется на URL хранилища для пути рабочей копии, использованному в `GetWCInfo`. Эквивалент `$WCURL$`
.IsSvnItem	True, если элемент версирован.
.NeedsLocking	True, если у элемента установлено свойство `svn:needs-lock`.
.IsLocked	True, если элемент заблокирован.
.LockCreationDate	Строка, содержащая дату, когда блокировка была создана, или пустая строка, если элемент не заблокирован.
.LockOwner	Строка, содержащая владельца блокировки, или пустая строка, если элемент не заблокирован.
.LockComment	Сообщение, введённое при создании блокировки.

Таблица 5.3. Поддерживаемые методы COM/автоматизации

Следующий пример показывает, как может быть использован этот интерфейс.

```
// testCOM.js - файл javascript
// пробный скрипт для COM/Automation-объекта SubWCRev

filesystem = new ActiveXObject("Scripting.FileSystemObject");

revObject1 = new ActiveXObject("SubWCRev.object");
revObject2 = new ActiveXObject("SubWCRev.object");
revObject3 = new ActiveXObject("SubWCRev.object");
revObject4 = new ActiveXObject("SubWCRev.object");

revObject1.GetWCInfo(
    filesystem.GetAbsolutePathName("."), 1, 1);
revObject2.GetWCInfo(
```

```
    filesystem.GetAbsolutePathName(".."), 1, 1);
revObject3.GetWCInfo(
    filesystem.GetAbsolutePathName("SubWCRev.cpp"), 1, 1);
revObject4.GetWCInfo(
    filesystem.GetAbsolutePathName("..\\.."), 1, 1);

wcInfoString1 = "Ревизия = " + revObject1.Revision +
    "\nMin ревизия = " + revObject1.MinRev +
    "\nMax ревизия = " + revObject1.MaxRev +
    "\nДата = " + revObject1.Date +
    "\nURL = " + revObject1.Url + "\nАвтор = " +
    revObject1.Author + "\nЕстьИзм = " +
    revObject1.HasModifications + "\nЭлементSVN = " +
    revObject1.IsSvnItem + "\nНеобходимаБлокировка = " +
 revObject1.NeedsLocking + "\nЗаблокировано = " +
    revObject1.IsLocked + "\nБлокировкаСоздана = " +
    revObject1.LockCreationDate + "\nВладелецБлокировки = " +
 revObject1.LockOwner + "\nКомментарийБлокировки = " +
 revObject1.LockComment;
wcInfoString2 = "Ревизия = " + revObject2.Revision +
    "\nMin ревизия = " + revObject2.MinRev +
    "\nMax ревизия = " + revObject2.MaxRev +
    "\nДата = " + revObject2.Date +
    "\nURL = " + revObject2.Url + "\nАвтор = " +
    revObject2.Author + "\nЕстьИзм = " +
    revObject2.HasModifications + "\nЭлементSVN = " +
    revObject2.IsSvnItem + "\nНеобходимаБлокировка = " +
 revObject2.NeedsLocking + "\nЗаблокировано = " +
    revObject2.IsLocked + "\nБлокировкаСоздана = " +
 revObject2.LockCreationDate + "\nВладелецБлокировки = " +
 revObject2.LockOwner + "\nКомментарийБлокировки = " +
 revObject2.LockComment;
wcInfoString3 = "Ревизия = " + revObject3.Revision +
    "\nMin ревизия = " + revObject3.MinRev +
    "\nMax ревизия = " + revObject3.MaxRev +
    "\nДата = " + revObject3.Date +
    "\nURL = " + revObject3.Url + "\nАвтор = " +
    revObject3.Author + "\nЕстьИзм = " +
    revObject3.HasModifications + "\nЭлементSVN = " +
    revObject3.IsSvnItem + "\nНеобходимаБлокировка = " +
 revObject3.NeedsLocking + "\nЗаблокировано = " +
    revObject3.IsLocked + "\nБлокировкаСоздана = " +
 revObject3.LockCreationDate + "\nВладелецБлокировки = " +
 revObject3.LockOwner + "\nКомментарийБлокировки = " +
 revObject3.LockComment;
wcInfoString4 = "Ревизия = " + revObject4.Revision +
    "\nMin ревизия = " + revObject4.MinRev +
    "\nMax ревизия = " + revObject4.MaxRev +
    "\nДата = " + revObject4.Date +
    "\nURL = " + revObject4.Url + "\nАвтор = " +
    revObject4.Author + "\nЕстьИзм = " +
    revObject4.HasModifications + "\nЭлементSVN = " +
    revObject4.IsSvnItem + "\nНеобходимаБлокировка = " +
 revObject4.NeedsLocking + "\nЗаблокировано = " +
    revObject4.IsLocked + "\nБлокировкаСоздана = " +
 revObject4.LockCreationDate + "\nВладелецБлокировки = " +
 revObject4.LockOwner + "\nКомментарийБлокировки = " +
 revObject4.LockComment;
```

```
WScript.Echo(wcInfoString1);
WScript.Echo(wcInfoString2);
WScript.Echo(wcInfoString3);
WScript.Echo(wcInfoString4);
```

Глава 6. IBugtraqProvider interface

To get a tighter integration with issue trackers than by simply using the `bugtraq:` properties, TortoiseSVN can make use of COM plugins. With such plugins it is possible to fetch information directly from the issue tracker, interact with the user and provide information back to TortoiseSVN about open issues, verify log messages entered by the user and even run actions after a successful commit to e.g, close an issue.

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. (Раздел 3, «TortoiseSVN бесплатен!» explains how to access the repository).

6.1. The IBugtraqProvider interface

TortoiseSVN 1.5 can use plugins which implement the IBugtraqProvider interface. The interface provides a few methods which plugins can use to interact with the issue tracker.

```
HRESULT ValidateParameters (
  // Parent window for any UI that needs to be
  // displayed during validation.
  [in] HWND hParentWnd,

  // The parameter string that needs to be validated.
  [in] BSTR parameters,

  // Is the string valid?
  [out, retval] VARIANT_BOOL *valid
);
```

This method is called from the settings dialog where the user can add and configure the plugin. The `parameters` string can be used by a plugin to get additional required information, e.g., the URL to the issue tracker, login information, etc. The plugin should verify the `parameters` string and show an error dialog if the string is not valid. The `hParentWnd` parameter should be used for any dialog the plugin shows as the parent window. The plugin must return TRUE if the validation of the `parameters` string is successful. If the plugin returns FALSE, the settings dialog won't allow the user to add the plugin to a working copy path.

```
HRESULT GetLinkText (
  // Parent window for any (error) UI that needs to be displayed.
  [in] HWND hParentWnd,

  // The parameter string, just in case you need to talk to your
  // web service (e.g.) to find out what the correct text is.
  [in] BSTR parameters,

  // What text do you want to display?
  // Use the current thread locale.
  [out, retval] BSTR *linkText
);
```

The plugin can provide a string here which is used in the TortoiseSVN commit dialog for the button which invokes the plugin, e.g., "Choose issue" or "Select ticket". Make sure the string is not too long,

otherwise it might not fit into the button. If the method returns an error (e.g., `E_NOTIMPL`), a default text is used for the button.

```
HRESULT GetCommitMessage (
  // Parent window for your provider's UI.
  [in] HWND hParentWnd,

  // Parameters for your provider.
  [in] BSTR parameters,
  [in] BSTR commonRoot,
  [in] SAFEARRAY(BSTR) pathList,

  // The text already present in the commit message.
  // Your provider should include this text in the new message,
  // where appropriate.
  [in] BSTR originalMessage,

  // The new text for the commit message.
  // This replaces the original message.
  [out, retval] BSTR *newMessage
);
```

This is the main method of the plugin. This method is called from the TortoiseSVN commit dialog when the user clicks on the plugin button. The `parameters` string is the string the user has to enter in the settings dialog when he configures the plugin. Usually a plugin would use this to find the URL of the issue tracker and/or login information or more. The `commonRoot` string contains the parent path of all items selected to bring up the commit dialog. Note that this is *not* the root path of all items which the user has selected in the commit dialog. The `pathList` parameter contains an array of paths (as strings) which the user has selected for the commit. The `originalMessage` parameter contains the text entered in the log message box in the commit dialog. If the user has not yet entered any text, this string will be empty. The `newMessage` return string is copied into the log message edit box in the commit dialog, replacing whatever is already there. If a plugin does not modify the `originalMessage` string, it must return the same string again here, otherwise any text the user has entered will be lost.

6.2. The IBugtraqProvider2 interface

In TortoiseSVN 1.6 a new interface was added which provides more functionality for plugins. This IBugtraqProvider2 interface inherits from IBugtraqProvider.

```
HRESULT GetCommitMessage2 (
  // Parent window for your provider's UI.
  [in] HWND hParentWnd,

  // Parameters for your provider.
  [in] BSTR parameters,
  // The common URL of the commit
  [in] BSTR commonURL,
  [in] BSTR commonRoot,
  [in] SAFEARRAY(BSTR) pathList,

  // The text already present in the commit message.
  // Your provider should include this text in the new message,
  // where appropriate.
  [in] BSTR originalMessage,

  // You can assign custom revision properties to a commit
  // by setting the next two params.
```

```
  // note: Both safearrays must be of the same length.
  //       For every property name there must be a property value!

  // The content of the bugID field (if shown)
  [in] BSTR bugID,

  // Modified content of the bugID field
  [out] BSTR * bugIDOut,

  // The list of revision property names.
  [out] SAFEARRAY(BSTR) * revPropNames,

  // The list of revision property values.
  [out] SAFEARRAY(BSTR) * revPropValues,

  // The new text for the commit message.
  // This replaces the original message
  [out, retval] BSTR * newMessage
);
```

This method is called from the TortoiseSVN commit dialog when the user clicks on the plugin button. This method is called instead of `GetCommitMessage()`. Please refer to the documentation for `GetCommitMessage` for the parameters that are also used there. The parameter `commonURL` is the parent URL of all items selected to bring up the commit dialog. This is basically the URL of the `commonRoot` path. The parameter `bugID` contains the content of the bug-ID field (if it is shown, configured with the property `bugtraq:message`). The return parameter `bugIDOut` is used to fill the bug-ID field when the method returns. The `revPropNames` and `revPropValues` return parameters can contain name/value pairs for revision properties that the commit should set. A plugin must make sure that both arrays have the same size on return! Each property name in `revPropNames` must also have a corresponding value in `revPropValues`. If no revision properties are to be set, the plugin must return empty arrays.

```
HRESULT CheckCommit (
  [in] HWND hParentWnd,
  [in] BSTR parameters,
  [in] BSTR commonURL,
  [in] BSTR commonRoot,
  [in] SAFEARRAY(BSTR) pathList,
  [in] BSTR commitMessage,
  [out, retval] BSTR * errorMessage
);
```

This method is called right before the commit dialog is closed and the commit begins. A plugin can use this method to validate the selected files/folders for the commit and/or the commit message entered by the user. The parameters are the same as for `GetCommitMessage2()`, with the difference that `commonURL` is now the common URL of all *checked* items, and `commonRoot` the root path of all checked items. The return parameter `errorMessage` must either contain an error message which TortoiseSVN shows to the user or be empty for the commit to start. If an error message is returned, TortoiseSVN shows the error string in a dialog and keeps the commit dialog open so the user can correct whatever is wrong. A plugin should therefore return an error string which informs the user *what* is wrong and how to correct it.

```
HRESULT  OnCommitFinished (
  // Parent window for any (error) UI that needs to be displayed.
  [in] HWND hParentWnd,

  // The common root of all paths that got committed.
  [in] BSTR commonRoot,
```

```
  // All the paths that got committed.
  [in] SAFEARRAY(BSTR) pathList,


  // The text already present in the commit message.
  [in] BSTR logMessage,

  // The revision of the commit.
  [in] ULONG revision,


  // An error to show to the user if this function
  // returns something else than S_OK
  [out, retval] BSTR * error
);
```

This method is called after a successful commit. A plugin can use this method to e.g., close the selected issue or add information about the commit to the issue. The parameters are the same as for `GetCommitMessage2`.

```
HRESULT HasOptions(
  // Whether the provider provides options
  [out, retval] VARIANT_BOOL *ret
);
```

This method is called from the settings dialog where the user can configure the plugins. If a plugin provides its own configuration dialog with `ShowOptionsDialog`, it must return TRUE here, otherwise it must return FALSE.

```
HRESULT ShowOptionsDialog(
  // Parent window for the options dialog
  [in] HWND hParentWnd,

  // Parameters for your provider.
  [in] BSTR parameters,

  // The parameters string
  [out, retval] BSTR * newparameters
);
```

This method is called from the settings dialog when the user clicks on the "Options" button that is shown if `HasOptions` returns TRUE. A plugin can show an options dialog to make it easier for the user to configure the plugin. The `parameters` string contains the plugin parameters string that is already set/entered. The `newparameters` return parameter must contain the parameters string which the plugin constructed from the info it gathered in its options dialog. That `paramameters` string is passed to all other IBugtraqProvider and IBugtraqProvider2 methods.

Приложение A. Часто задаваемые вопросы (ЧаВо, FAQ)

Поскольку TortoiseSVN разрабатывается постоянно, иногда тяжело содержать документацию в актуальном состоянии. Мы поддерживаем *размещённый в Сети список ЧаВо (FAQ)* [http://tortoisesvn.tigris.org/faq.html], в котором содержатся подборка из вопросов, наиболее часто задаваемых в списках рассылки TortoiseSVN <dev@tortoisesvn.tigris.org> и <users@tortoisesvn.tigris.org>.

Мы также поддерживаем *систему отслеживания проблем проекта* [http://issues.tortoisesvn.net], в которой говорится о некоторых вещах из нашего списка того, что мы хотели бы сделать, и об уже исправленных ошибках. Если вы думаете, что обнаружили ошибку, или желаете попросить новую возможность, проверьте сначала здесь, чтобы убедиться, не сделал ли это до вас кто-либо другой.

Если у вас есть вопрос, ответа на который вы больше нигде не нашли, лучшее место, где можно спросить - один из наших списков рассылки. <users@tortoisesvn.tigris.org> служит для вопросов по использованию TortoiseSVN. Если у вас есть желание помочь с разработкой TortoiseSVN, тогда вы должны принять участие в дискуссиях на <dev@tortoisesvn.tigris.org>.

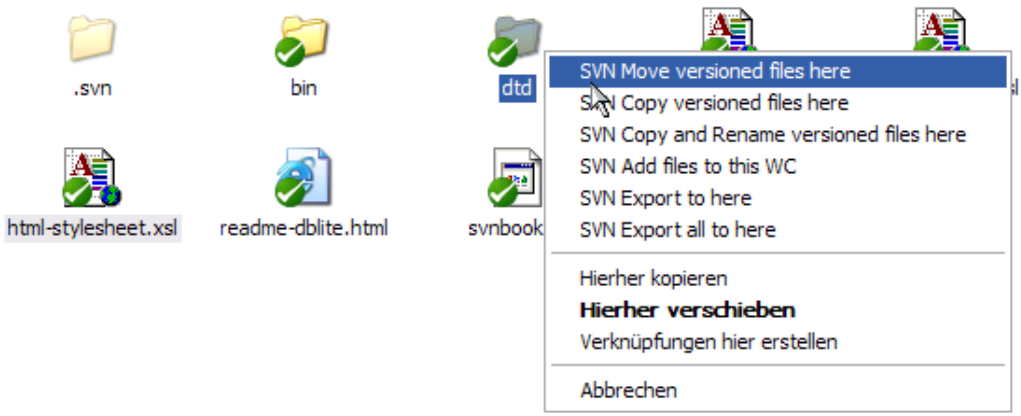
Приложение B. Как я могу...

Это приложение содержит решения проблем/вопросов, которые могут у вас возникнуть при использовании TortoiseSVN.

B.1. Переместить/скопировать множество файлов за один раз

Перемещение/копирование файлов может быть сделано при помощи TortoiseSVN → Переименовать.... Но если вы собираетесь переместить/скопировать много файлов, этот способ может быть слишком медленным и потребовать слишком много работы.

Рекомендуемый способ - это правое перетаскивание файлов в новое местоположение. Просто выполните правый щелчок на файлах, которые вы желаете переместить/скопировать, и, не отпуская кнопку мыши, перетащите файлы в новое место, после чего отпустите кнопку мыши. Появится контекстное меню, в котором вы можете выбрать Контекстное меню → SVN Копировать версированные файлы сюда, или Контекстное меню → SVN Переместить версированные файлы сюда.



B.2. Заставить пользователей вводить сообщение журнала

Есть два способа предотвратить фиксации с пустыми сообщениями журнала. Один из них доступен только в TortoiseSVN, другой работает для всех клиентов Subversion, но требует непосредственного доступа к серверу.

B.2.1. Скрипт ловушки на сервере

Если вы имеете непосредственный доступ к серверу хранилища, вы можете установить скрипт ловушки перед-фиксацией, который будет отклонять все фиксации с пустыми или слишком короткими сообщениями журнала.

На сервере в папке хранилища есть подпапка `hooks`, содержащая несколько примеров скриптов ловушек, которые вы можете использовать. Файл `pre-commit.tmpl` содержит пример скрипта, который отклоняет фиксации при отсутствии сообщения журнала, или в случае, если сообщение слишком короткое. В файле также содержатся комментарии о том, как установить/использовать этот скрипт; просто следуйте инструкциям в этом файле.

Этот метод рекомендуется, если ваши пользователи используют также клиенты, отличные от TortoiseSVN. Недостаток этого метода заключается в том, что фиксация отклоняется сервером, и

из-за этого пользователи получают сообщение об ошибке. Клиент перед выполнением фиксации не знает, что она будет отклонена. Если вы желаете, чтобы TortoiseSVN отключал кнопку OK до тех пор, пока сообщение журнала не достигнет достаточной длины, то воспользуйтесь методом, описанным ниже.

B.2.2. Свойства проекта

TortoiseSVN использует свойства для управления некоторыми своими возможностями. Одно из этих свойств - `tsvn:logminsize`.

Если вы установите это свойство на папке, то TortoiseSVN будет отключать кнопку OK, пока пользователь не введёт сообщение журнала длиной не меньше, чем указано в этом свойстве.

Чтобы получить более подробную информацию об этих свойствах проекта, прочтите Раздел 4.17, «Установки проекта»

B.3. Обновить выбранные файлы из хранилища

Обычно вы обновляете вашу рабочую копию при помощи TortoiseSVN → Обновить. Но если вы желаете получить только несколько добавленных коллегами новых файлов, которые не потребуют выполнения слияния с другими файлами, вам нужен другой подход.

Вызовите TortoiseSVN → Проверить на наличие изменений и нажмите на Проверить хранилище для просмотра того, что изменилось в хранилище. Выберите файлы, которые вы желаете обновить локально, затем воспользуйтесь контекстным меню для обновления только этих файлов.

B.4. Возвратиться к старым ревизиям в хранилище (откат)

B.4.1. При помощи диалога журнала ревизий

Наиболее лёгкий способ убрать изменения, произведённые в одной ревизии, или в диапазоне ревизий, - это применение диалога журнала ревизий. Этот метод также используется, если вы желаете отказаться от последних изменений и сделать более раннюю ревизию новой ведущей.

1. Выберите файл или папку, в которых вы собираетесь убрать изменения. Если вы желаете убрать все изменения, это должна быть папка верхнего уровня.

2. Выберите TortoiseSVN → Журнал для отображения списка ревизий. Возможно, вам понадобиться использовать кнопки Показать все или Следующие 100 для отображения нужных вам ревизий.

3. Выберите ревизию, которую вы желаете убрать. Если вы желаете убрать диапазон ревизий, выберите первую, и, нажав клавишу **Shift**, выберите последнюю. Обратите внимание: для нескольких ревизий диапазон должен быть непрерывным, без пропусков. Выполните правый щелчок на выбранных ревизиях, после чего выберите Контекстное меню → Отменить изменения из этой ревизии.

4. Или, если вы желаете сделать более раннюю ревизию новой ведущей, выполните правый щелчок на выбранной ревизии, затем выберите Контекстное меню → Вернуть к этой ревизии. Это действие отменит *все* изменения после выбранной ревизии.

Вы выполнили отмену изменений внутри вашей рабочей копии. Проверьте результат, затем зафиксируйте изменения.

B.4.2. Используя диалог слияния

Для отмены большего диапазона ревизий вы можете использовать диалог слияния. В предыдущем методе слияние используется негласно, в этом методе оно применяется явно.

1. В вашей рабочей копии выберите TortoiseSVN → Слить...[1].
2. В поле От: введите полный URL папки ответвления/метки, содержащих изменения, которые вы желаете отменить в вашей рабочей копии. Далее этот URL будет упоминаться как URL по умолчанию.
3. В поле От ревизии введите номер вашей текущей ревизии. Если вы уверены, что больше никто не сделает изменений, вы можете указать ведущую ревизию.
4. Убедитесь, что флажок Использовать тот же URL, как и в "От:" отмечен.
5. В поле До ревизии введите номер ревизии, к которой вы желаете вернуться, а именно номер ревизии непосредственно *перед* первой ревизией, которая будет отменена.
6. Нажмите OK для выполнения слияния.

Вы выполнили отмену изменений внутри вашей рабочей копии. Проверьте результат, затем зафиксируйте изменения.

B.4.3. Используя `svndumpfilter`

Поскольку данные в TortoiseSVN никогда не пропадают, ваши «отменённые» ревизии всё ещё существуют как промежуточные ревизии в хранилище. Только ведущая ревизия была изменена к предыдущему состоянию. Если вы желаете полностью убрать ревизии из хранилища и удалить все когда-либо существовавшие следы, вы должны будете применить более экстремальные меры. Это делать *не рекомендуется*, если только у вас нет очень веских оснований. Одной из возможных причин может быть фиксация конфиденциального документа в общедоступном хранилище.

Единственный путь удалить данные из хранилища - это использование инструмента командной строки Subversion `svnadmin`. Описание того, как с ним работать, содержит раздел *Обслуживание хранилища (Repository Maintenance)* [http://svnbook.red-bean.com/en/1.5/svn.reposadmin.maint.html] книги о Subversion.

B.5. Compare two revisions of a file or folder

If you want to compare two revisions in an item's history, for example revisions 100 and 200 of the same file, just use TortoiseSVN → Show Log to list the revision history for that file. Pick the two revisions you want to compare then use Context Menu → Compare Revisions.

If you want to compare the same item in two different trees, for example the trunk and a branch, you can use the repository browser to open up both trees, select the file in both places, then use Context Menu → Compare Revisions.

If you want to compare two trees to see what has changed, for example the trunk and a tagged release, you can use TortoiseSVN → Revision Graph Select the two nodes to compare, then use Context Menu → Compare HEAD Revisions. This will show a list of changed files, and you can then select individual files to view the changes in detail. You can also export a tree structure containing all the changed files, or simply a list of all changed files. Read Раздел 4.10.3, «Сравнение папок» for more information. Alternatively use Context Menu → Unified Diff of HEAD Revisions to see a summary of all differences, with minimal context.

B.6. Включить общий подпроект

Иногда вы желаете включить другой проект в вашу рабочую копию, возможно, некоторый библиотечный код. Вы не хотите делать копию этого кода в вашем хранилище, поскольку в этом случае вы потеряете связь с оригинальным (и поддерживаемым) кодом. Или, может быть, у вас есть несколько проектов, у которых основной код общий. Существуют по крайней мере 3 способа, как это организовать.

B.6.1. Используя svn:externals

Set the `svn:externals` property for a folder in your project. This property consists of one or more lines; each line has the name of a sub-folder which you want to use as the checkout folder for common code, and the repository URL that you want to be checked out there. For full details refer to Раздел 4.18, «Внешние включения».

Зафиксируйте новую папку. Теперь, при обновлении Subversion извлечёт копию этого проекта из хранилища и разместит в вашей рабочей копии. Подпапки при необходимости будут созданы автоматически. Каждый раз при обновлении основной рабочей копии вы также будете получать последнюю версию всех внешних проектов.

Если внешний проект находится в том же хранилище, то любые изменения, которые вы в нём сделаете, будут включены в список фиксации при фиксировании основного проекта.

Если внешний проект находится в другом хранилище, вы будете уведомлены о сделанных вами изменениях во внешнем проекте при фиксировании основного проекта, но вы должны будете зафиксировать эти внешние изменения отдельно.

Из трёх описанных методов, это единственный, не требующий настройки на стороне клиента. Как только внешние ссылки заданы в свойствах папки, у всех клиентов при следующем обновлении папки будут заполнены.

B.6.2. Используя вложенную рабочую копию

Создайте новую папку в вашем проекте, которая будет содержать общий код, но не добавляйте её в Subversion.

Выполните TortoiseSVN → SVN Извлечь... на новой папке, и извлеките в неё копию общего кода. Сейчас у вас есть отдельная рабочая копия, вложенная в вашу основную рабочую копию.

Эти две рабочие копии независимы. При фиксации изменений в родительской рабочей копии изменения во вложенной игнорируются. Точно также, когда вы выполняете обновление родительской рабочей копии, вложенная не обновляется.

B.6.3. Используя относительное месторасположение

Если вы используете один и тот же общий основной код в нескольких проектах, и вы не желаете заводить несколько рабочих копий для него в каждом проекте, который его использует, вы можете просто извлечь его в отдельное месторасположение, связанное со всеми остальными использующими его проектами. Например:

```
C:\Проекты\Проект1
C:\Проекты\Проект2
C:\Проекты\Проект3
C:\Проекты\Общее
```

и ссылаться на общий код при помощи относительного пути, например `..\..\Общее \Ядро_DSP`.

Если ваши проекты разбросаны по нескольким несвязанным местоположениям, вы можете воспользоваться вариант этого решения, в котором общий код помещается в одно место, а затем используется подстановку буквы диска для отображения этого местоположения на то, что вы можете жестко запрограммировать в ваших проектах. Например, извлеките общий код в `D:\Documents\Framework` или `C:\Documents and Settings\{login}\My Documents\framework`, затем используйте

```
SUBST X: "D:\Documents\framework"
```

для создания отображения диска, используемого в вашем исходном коде. После этого ваш код может использовать абсолютные местоположения.

```
#include "X:\superio\superio.h"
```

Этот метод будет работать только в окружении, состоящем только из ПК под управлением Windows, и вам придётся задокументировать необходимые подстановки дисков, чтобы ваша команда знала, где расположены все эти загадочные файлы. Этот метод предназначен строго для использования в условиях закрытой разработки, и не рекомендуется для общего использования.

B.7. Создать ярлык к хранилищу

Если вам часто необходимо открывать обозреватель хранилища для конкретного местоположения, вы можете создать ярлык на рабочем столе, применив интерфейс автоматизации TortoiseProc. Просто создайте новый ярлык и установите поле размещения объекта в

```
TortoiseProc.exe /command:repobrowser /path:"url/адрес/хранилища"
```

Конечно же, вы должны указать реальный URL хранилища.

B.8. Игнорировать файлы, которые уже версированы

Если вы случайно добавили некоторые файлы, которые должны быть проигнорированы, как вы можете убрать их из-под управления версиями, не потеряв их? Возможно, у вас есть собственный файл настроек IDE, который не является частью проекта, но который вы долгое время настраивали под себя.

Если вы пока ещё не зафиксировали добавление, тогда всё, что вы сделать, это выполнить TortoiseSVN → Убрать изменения... для отмены добавления. Затем вы должны добавить файл(-ы) в список игнорирования, чтобы они больше не могли быть позже снова добавлены по ошибке.

Если файлы уже в хранилище, вам необходимо сделать немного больше работы:

1. Нажмите клавишу **Shift** для получения расширенного контекстного меню и выполните TortoiseSVN → Удалить (оставив локально) для того, чтобы пометить файл/папку для удаления из хранилища, но без удаления их локальной копии.
2. Выполните TortoiseSVN → SVN Фиксировать... на родительской папке.
3. Добавьте этот файл/папку в список игнорирования, чтобы не попасть в эту же ситуацию снова.

B.9. Разверсирование рабочей копии

Если у вас есть рабочая копия, которую вы бы желали преобразовать обратно в обычную папку без всех этих папок `.svn`, вы можете просто экспортировать её в саму себя. Прочтите Раздел 4.26.1, «Выведение рабочей копии из-под управления версиями», чтобы узнать как это сделать.

B.10. Удаление рабочей копии

Если у вас есть рабочая копия, которая вам больше не нужна, как корректно от неё избавиться? Легко - просто удалите её в Проводнике Windows! Рабочие копии являются независимыми от других локальными объектами, и они вполне самодостаточны[2].

[2] т.е. сами содержат всё, что к ним относится :) - прим. переводчика

Приложение C. Полезные подсказки для администраторов

Это приложение содержит решения проблем/вопросов, которые могут возникнуть, когда вы ответственны за распространение TortoiseSVN на нескольких клиентских компьютерах.

C.1. Распространение TortoiseSVN через групповые политики

Установщик TortoiseSVN поставляется в виде MSI-файла, и это означает, что у вас не должно быть проблем при добавлении этого MSI-файла в групповые политики вашего контроллера домена.

Хорошее пошаговое руководство о том, как это сделать, можно найти в статье 314934 базы знаний Microsoft: *http://support.microsoft.com/?kbid=314934*.

TortoiseSVN версии 1.3.0 и более поздних должен устанавливаться в разделе *Computer Configuration*, а не в *User Configuration*, поскольку этим версиям нужны новые библиотеки DLL для CRT и MFC, которые могут быть развёрнуты только *для компьютера*, но не *для пользователя*. Если вы действительно должны установить TortoiseSVN только для конкретных пользователей, тогда вам нужно сначала установить пакеты MFC и CRT версии 8 от Microsoft на каждый компьютер, на котором вы планируете в дальнейшем устанавливать TortoiseSVN для конкретных пользователей.

C.2. Перенаправление проверки обновлений

TortoiseSVN каждые несколько дней проверяет, не появилась ли новая версия. Если доступна новая версия, показывается диалог, информирующий об этом пользователя.

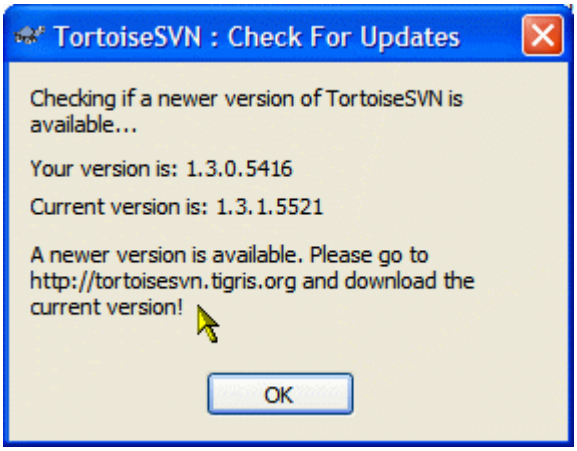


Рисунок C.1. Диалог обновления

Если вы ответственны за большое число пользователей в вашем домене, вы можете пожелать, чтобы пользователи использовали только одобренные версии, а не устанавливали всегда последнюю версию. Возможно, вы желаете, чтобы этот диалог обновления вообще не показывался, поскольку после его появления пользователи тут же идут и обновляются.

TortoiseSVN версии 1.4.0 и более поздних позволяет вам перенаправить проверку обновления на ваш внутренний сервер. Вы можете установить ключ реестра `HKCU\Software\TortoiseSVN\UpdateCheckURL` (строковое значение) в URL, указывающий на текстовый файл в вашей внутренней сети. Этот текстовый файл должен иметь следующий формат:

```
1.4.1.6000
Доступна для скачивания новая версия TortoiseSVN!
http://192.168.2.1/downloads/TortoiseSVN-1.4.1.6000-svn-1.4.0.msi
```

Первая строка в этом файле - это строка версии. Вы должны убедиться, что она точно соответствует строке версии установочного пакета TortoiseSVN. Вторая строка - это произвольный текст, показываемый в диалоге обновления программы. Вы можете написать здесь что угодно, просто помните, что размер диалога обновления ограничен. Слишком длинные сообщения будут обрезаны! Третья строка - это URL нового установочного пакета. Этот URL открывается, когда пользователь щелкает на сообщении в диалоге обновления. Вы также можете отправить пользователя на веб-страницу, вместо указания прямой ссылки на MSI-файл. URL открывается в веб-обозревателе по умолчанию, так что если вы укажете страницу, она будет открыта и показана пользователю. Если вы укажете пакет MSI, обозреватель предложит пользователю сохранить этот MSI-файл локально.

C.3. Установка переменной окружения `SVN_ASP_DOT_NET_HACK`

Начиная с версии 1.4.0, установщик TortoiseSVN больше не предоставляет пользователю возможность установки переменной окружения `SVN_ASP_DOT_NET_HACK`, поскольку это вызывало множество проблем и путаницы у пользователей, всегда устанавливающих *всё*, даже не имея понятия, для чего это предназначено.

Но эта настройка скрыта только от пользователя. Вы всё ещё можете заставить установщик TortoiseSVN установить эту переменную окружения путём установки свойства ASPDOTNETHACK в TRUE. Например, вы можете запустить установщик следующим образом:

```
msiexec /i TortoiseSVN-1.4.0.msi ASPDOTNETHACK=TRUE
```

C.4. Отключение пунктов контекстного меню

Начиная с версии 1.5.0, TortoiseSVN позволяет отключать (а на самом деле просто скрывать) пункты контекстного меню. Поскольку эта возможность не должна использоваться необдуманно, а только в качестве вынужденной меры, то для работы с ней средств в графическом интерфейсе пользователя не предусмотрено, и она может быть задействована путём непосредственного редактирования реестра. Эта возможность может быть применена для отключения определённых команд тем пользователям, которые не должны их использовать. Но учтите пожалуйста, что скрываются только пункты контекстного меню в *Проводнике*, а команды всё равно остаются доступными другими способами, например, через командную строку или даже через другие диалоги в самом TortoiseSVN!

Информация о том, какие пункты меню должны быть показаны, содержится в следующих ключах реестра: `HKEY_CURRENT_USER\Software\TortoiseSVN\ContextMenuEntriesMaskLow` и `HKEY_CURRENT_USER\Software\TortoiseSVN\ContextMenuEntriesMaskHigh`.

Каждый из этих ключей содержит значение типа `DWORD`, в котором каждый бит соответствует какому-либо пункту меню. Установленный бит означает, что соответствующий пункт меню будет отключен.

Значение	Пункт меню
0x0000000000000001	Извлечь
0x0000000000000002	Обновить

Значение	Пункт меню
0x0000000000000004	Фиксировать
0x0000000000000008	Добавить
0x0000000000000010	Убрать изменения
0x0000000000000020	Очистка
0x0000000000000040	Уладить
0x0000000000000080	Параметр
0x0000000000000100	Импорт
0x0000000000000200	Экспорт
0x0000000000000400	Создать здесь хранилище
0x0000000000000800	Ответвление/Метка
0x0000000000001000	Слияние
0x0000000000002000	Удалить
0x0000000000004000	Переименовать
0x0000000000008000	Обновить до ревизии
0x0000000000010000	Различие
0x0000000000020000	Журнал
0x0000000000040000	Редактировать конфликты
0x0000000000080000	Перебазировать
0x0000000000100000	Проверить на наличие изменений
0x0000000000200000	Игнорировать
0x0000000000400000	Обозреватель хранилища
0x0000000000800000	Авторство (Blame)
0x0000000001000000	Создать заплатку
0x0000000002000000	Применить заплатку
0x0000000004000000	Граф ревизий
0x0000000008000000	Блокировка
0x0000000010000000	Снять блокировку
0x0000000020000000	Свойства
0x0000000040000000	Различия с файлом по URL
0x0000000080000000	Удалить неверсированные элементы
0x2000000000000000	Настройки
0x4000000000000000	Справка
0x8000000000000000	О программе

Таблица C.1. Пункты меню и соответствующие им значения

Например: для отключения пунктов меню «Перебазировать», «Удалить неверсированные элементы» и «Настройки», сложите значения, назначенные этим пунктам следующим образом:

```
  0x0000000000080000
+ 0x0000000080000000
+ 0x2000000000000000
```

```
= 0x2000000080080000
```

Значение младшего DWORD (0x80080000) должно быть записано в HKEY_CURRENT_USER\Software\TortoiseSVN\ContextMenuEntriesMaskLow, значение старшего DWORD (0x20000000) - в HKEY_CURRENT_USER\Software\TortoiseSVN\ContextMenuEntriesMaskHigh.

Для того, чтобы снова включить эти пункты меню, просто удалите эти два ключа реестра.

Приложение D. Автоматизация TortoiseSVN

Поскольку всеми командами TortoiseSVN можно управлять при помощи параметров командной строки, вы можете использовать их для автоматизации в пакетных скриптах или же запустить конкретную команду и диалог из других программ (например, из вашего любимого текстового редактора).

Важно

Помните, что TortoiseSVN является клиентом с графическим интерфейсом пользователя, и это руководство по автоматизации показывает, как показывать диалоги TortoiseSVN для получения пользовательского ввода. Если вы хотите написать скрипт, который не требует ввода, вы должны использовать официальный клиент Subversion для командной строки.

D.1. Команды TortoiseSVN

Программа, отображающая интерфейс пользователя TortoiseSVN, называется `TortoiseProc.exe`. Все команды указываются с параметрами `/command:abcd`, где `abcd` - это обязательное имя команды. Большинству из этих команд необходим как минимум один параметр с именем пути, задаваемый при помощи `/path:"некоторый\путь"`. В следующей таблице 'команда' обозначает параметр `/command:abcd` и 'путь' обозначает параметр `/path:"некоторый\путь"`.

Так как некоторые команды могут принимать список целевых путей (например, фиксация нескольких конкретных файлов) в параметре `/path` можно задать несколько путей, разделённых символом `*`.

TortoiseSVN использует временные файлы для передачи множественных параметров между расширением оболочки и основной программой. Начиная с TortoiseSVN 1.5.0, параметр `/notempfile` считается устаревшим и его больше добавлять не надо.

Окно выполнения, используемое при фиксации, обновлении и множестве других команд, обычно остаётся открытым после завершения команды до тех пор, пока пользователь не нажмёт кнопку OK. Это может быть изменено установкой соответствующего параметра в диалоге настроек. Но использование этой настройки будет закрывать окно выполнения независимо от того, запущена ли команда из вашего пакетного командного файла или из контекстного меню TortoiseSVN.

Для указания другого местоположения конфигурационного файла, используйте параметр `/configdir:"путь\до\конигурационной\папки"`. Этот параметр будет использован вместо пути по умолчанию, а также вместо настроек из реестра.

To close the progress dialog at the end of a command automatically without using the permanent setting you can pass the `/closeonend` parameter.

- `/closeonend:0` не закрывать диалог автоматически
- `/closeonend:1` закрывать автоматически, если нет ошибок
- `/closeonend:2` закрывать автоматически, если нет ошибок и конфликтов
- `/closeonend:3` закрывать автоматически, если нет ошибок, конфликтов и слияний

- `/closeonend:4` закрывать автоматически, если нет ошибок, конфликтов и слияний для локальных операций

В нижеприведённой таблице содержатся все команды, доступные при использовании командной строки TortoiseProc.exe. Как описано выше, они должны использоваться в виде `/command:abcd`. В таблице префикс `/command` опущен для экономии места.

Команда	Описание
:about	Показывает диалог 'О программе'. Он же отображается, если команда не указана.
:log	Opens the log dialog. The `/path` specifies the file or folder for which the log should be shown. Three additional options can be set: `/startrev:xxx`, `/endrev:xxx` and `/strict`
:checkout	Открывает диалог извлечения. Параметр `/path` задаёт целевую папку, а `/url` задаёт URL, из которого будет происходить извлечение.
:import	Открывает диалог импорта. Параметр `/path` задаёт папку с данными, которые должны быть импортированы.
:update	Обновляет рабочую копию в `/path` до ведущей ревизии (HEAD). Если указан параметр `/rev`, то показывается диалог, запрашивающий, до какой ревизии должно происходить обновление у пользователя. Чтобы диалог не появлялся, укажите номер ревизии `/rev:1234`. Другие параметры: `/nonrecursive` и `/ignoreexternals`.
:commit	Открывает диалог фиксации. Параметр `/path` задаёт целевую папку или список файлов для фиксации. Вы можете также задать параметр `/logmsg` для указания предопределённого сообщения журнала, которое будет передано в диалог фиксации. Или, если вы не желаете передавать сообщение журнала в командной строке, воспользуйтесь `/logmsgfile:путь`, где `путь` задаёт файл, содержащий сообщение журнала. Для предварительного заполнения поля 'ID ошибки' (в случае, если у вас настроена интеграция с системой отслеживания ошибок) вы можете применить параметр `/bugid:"здесь id ошибки"`.
:add	Добавляет файлы в `/path` под управление версиями.
:revert	Отменяет локальные изменения в рабочей копии. Параметр `/path` указывает, какие элементы должны быть возвращены в прежнее состояние.
:cleanup	Очищает прерванные или отменённые операции и разблокирует рабочую копию в `/path`.
:resolve	Помечает конфликтные файлы, указанные в `/path`, как улаженные. Если задан параметр `/noquestion`, то улаживание производится, не спрашивая предварительно разрешения на выполнение у пользователя.
:repocreate	Создаёт хранилище в папке, указанной параметром `/path`
:switch	Открывает диалог переключения. Параметр `/path` задаёт целевую папку.
:export	Экспортирует рабочую копию из `/path` в другую папку. Если `/path` указывает на неверсированную папку, диалог запросит URL, который надо будет экспортировать в папку, заданную параметром `/path`.
:merge	Открывает диалог слияния. Параметр `/path` задаёт целевую папку. Для слияния диапазона ревизий доступны следующие параметры: `/fromurl:URL`, `/revrange:строка`. Для слияния двух деревьев из хранилища доступны следующие параметры: `/fromurl:URL`, `/tourl:URL`, `/fromrev:xxx` и `/torev:xxx`. Эти параметры будут подставлены в соответствующие поля в диалоге слияния.

Команда	Описание
:mergeall	Открывает диалог 'Слить все'. Параметр `/path` задаёт целевую папку.
:copy	Открывает диалог ответвления/метки. Параметр `/path` задаёт рабочую копию, из которой будет выполнятся создаваться ответвление/метка. Параметр `/url` задаёт целевой URL. Вы можете также задать параметр `/logmsg` для указания предопределённого сообщения журнала, которое будет передано в диалог ответвления/метки. Или, если вы не желаете передавать сообщение журнала в командной строке, воспользуйтесь `/logmsgfile:путь`, где `путь` задаёт файл, содержащий сообщение журнала.
:settings	Открывает диалог настроек.
:remove	Убирает файл(-ы) в `/path` из-под управления версиями.
:rename	Переименовывает файл, заданный параметром `/path`. Новое имя для файла запрашивается при помощи диалога. Для подавления вопроса о переименовании похожих файлов за один приём, укажите `/noquestion`.
:diff	Запускает указанную в настройках TortoiseSVN внешнюю программу сравнения. Параметр `/path` задаёт первый файл. Если указан параметр `/path2`, то программа сравнения запускается с этими двумя файлами. Если параметр `/path2` опущен, то сравнение делается между файлом, задаваемым `/path` и его базой. Для явного задания номеров ревизий используйте параметры `/startrev:xxx` и `/endrev:xxx`. Если параметр `/blame` задан, а `/path2` - нет, то сначала производится получение авторства для файлов в заданных ревизиях, после чего выполняется сравнение.
:showcompare	В зависимости от указанных URL и ревизий для сравнения, будет показаны либо объединённые различия (если указан параметр `unified`), диалог со списком изменённых файлов, либо, если адреса URL указывают на файлы, запускает программу просмотра различий для этих двух файлов. Параметры `url1`, `url2`, `revision1` и `revision2` должны быть указаны. Параметры `pegrevision`, `ignoreancestry`, `blame` и `unified` являются необязательными.
:conflicteditor	Запускает указанный в настройках TortoiseSVN редактор конфликтов для файлов, соответствующих конфликтующему файлу, задаваемому параметром `/path`.
:relocate	Открывает диалог перебазирования. Параметр `/path` указывает путь рабочей копии, который будет перебазирован.
:help	Открывает файл справки.
:repostatus	Открывает диалог проверки на наличие изменений. Параметр `/path` указывает папку рабочей копии.
:repobrowser	Запускает обозреватель хранилища, указывающий на URL из рабочей копии, заданной `/path` или же `/path` может указывать прямо на URL. Для указания ревизии, которую обозреватель хранилища должен показывать, можно использовать дополнительный параметр `/rev:xxx`. Если параметр `/rev:xxx` опущен, то по умолчанию используется ведущая ревизия (HEAD). Если параметр `/path` указывает на URL, то параметр `/projectpropertiespath:путь/к/рабочей/копии` задаёт путь, по которому должны считываться и применяться свойства проекта.

Команда	Описание
:ignore	Добавляет все целевые файлы из `/path` в список игнорирования, т.е. добавляет к этим файлам свойство `svn:ignore`.
:blame	Открывает диалог авторства для файла, указанного в `/path`. Если заданы параметры `/startrev` и `/endrev`, то диалог запроса диапазона ревизий для получения информации об авторстве не отображается, вместо этого используются значения этих параметров. Если задан параметр `/line:nnn`, TortoiseBlame откроется, отображая строку с указанным номером. Также поддерживаются параметры `/ignoreeol` (игнорировать окончания строк), `/ignorespaces` (игнорировать непечатаемые знаки) и `/ignoreallspaces` (игнорировать все непечатаемые знаки).
:cat	Сохраняет файл из URL или пути в рабочей копии, заданному в `/path`, в место, указанное в `/savepath:путь`. Ревизия передается при помощи `/revision:xxx`. Может быть использовано для получения файла нужной ревизии.
:createpatch	Создаёт файл заплатки для пути, указанному в `/path`
:revisiongraph	Отображает граф ревизий для пути, указанному в `/path`.
:lock	Блокирует файл или все файлы в папке, указанной в `/path`. Отображается диалог блокирования, чтобы пользователь мог ввести комментарий для блокировки.
:unlock	Разблокирует файл или все файлы в папке, указанной в `/path`.
:rebuildiconcache	Восстанавливает кэш значков Windows. Используйте только в случае, если значки Windows испорчены. Побочный эффект этой команды (которого нельзя избежать) состоит в переупорядочивании значков на рабочем столе. Чтобы не появлялось окно сообщения, укажите `/noquestion`.
:properties	Показывает диалог работы со свойствами для пути, указанному в `/path`.

Таблица D.1. Список доступных команд и параметров

Примеры (должны быть введены в одной строке):

```
TortoiseProc.exe /command:commit
                 /path:"c:\svn_wc\file1.txt*c:\svn_wc\file2.txt"
                 /logmsg:"test log message" /closeonend:0

TortoiseProc.exe /command:update /path:"c:\svn_wc\" /closeonend

TortoiseProc.exe /command:log /path:"c:\svn_wc\file1.txt"
                 /startrev:50 /endrev:60 /closeonend:0
```

D.2. Команды TortoiseIDiff

У программы-инструмента сравнения картинок есть несколько параметров командной строки, используемых для управления её запуском. Программа называется `TortoiseIDiff.exe`.

В нижеприведённой таблице перечислены все параметры, которые можно передать в командной строке инструменту сравнения картинок.

Параметр	Описание
:left	Путь к файлу, показываемому слева.
:lefttitle	Строка заголовка. Эта строка используется в заголовке картинки вместо полного пути к файлу.
:right	Путь к файлу, показываемому справа.
:righttitle	Строка заголовка. Эта строка используется в заголовке картинки вместо полного пути к файлу.
:overlay	Если указано, инструмент сравнения картинок переключается в режим наложения картинок (альфа-сопряжение).
:fit	Если указано, инструмент сравнения картинок будет вписывать обе картинки в окно.
:showinfo	Показывает окно информации о картинке.

Таблица D.2. Список доступных параметров

Пример (должен быть введён в одной строке):

```
TortoiseIDiff.exe /left:"c:\images\img1.jpg" /lefttitle:"image 1"
                  /right:"c:\images\img2.jpg" /righttitle:"image 2"
                  /fit /overlay
```

Приложение E. Справочник соответствия с интерфейсом командной строки

Иногда это руководство отсылает вас к основной документации Subversion, которая описывает Subversion в терминах интерфейса командной строки - ИКС (Command Line Interface - CLI). Чтобы помочь вам понять, что TortoiseSVN делает за кулисами, мы составили список, показывающий эквиваленты команд ИКС для каждой операции, выполняемой в графическом интерфейсе TortoiseSVN.

> **Замечание**
>
> Даже несмотря на то, что в ИКС есть эквиваленты для операций, выполняемых TortoiseSVN, помните, что TortoiseSVN *не вызывает* ИКС, а использует библиотеку Subversion напрямую.

Если вы думаете, что нашли ошибку в TortoiseSVN, мы можем попросить вас попробовать воспроизвести её при помощи ИКС, чтобы мы могли отделить проблемы с TortoiseSVN от проблем с Subversion. Этот справочник расскажет вам, какую команду надо попробовать.

E.1. Соглашения и основные правила

В последующем описании URL расположения хранилища показывается просто как `URL`, и в качестве примера такого URL может служить `http://tortoisesvn.tigris.org/svn/tortoisesvn/trunk`. Путь к рабочей копии показывается просто как `ПУТЬ`, и примером такого пути может быть `C:\TortoiseSVN\trunk`.

> **Важно**
>
> Из-за того, что TortoiseSVN является расширением Проводника Windows, невозможно использовать обозначение текущей рабочей папки. Все пути к рабочим копиям должны указываться при помощи абсолютных, а не относительных путей.

Определённые элементы являются необязательными, и они часто задаются флажками или переключателями в TortoiseSVN. Эти опции показаны в [квадратных скобках] в определениях командной строки.

E.2. Команды TortoiseSVN

E.2.1. Извлечь

```
svn checkout [-N] [--ignore-externals] [-r rev] URL ПУТЬ
```

Если помечен флажок Извлечь только папку верхнего уровня, используйте параметр `-N`.

Если помечен флажок Пропустить внешние, используйте параметр `--ignore-externals`.

Если вы извлекаете конкретную ревизию, укажите её после URL при помощи параметра `-r`.

E.2.2. Обновить

```
svn info URL_рабочей_копии
svn update [-r rev] ПУТЬ
```

Обновление нескольких элементов в данный момент не является атомарной операцией в Subversion. Поэтому TortoiseSVN сначала находит ведущую ревизию (HEAD) в хранилище, а затем обновляет все элементы до этой ревизии во избежание создания рабочей копии со смешанными ревизиями.

Если для обновления выбран только один элемент, или выбранные элементы не все из одного и того же хранилища, TortoiseSVN просто обновляет до ведущей ревизии.

Здесь параметры командной строки не используются. Обновить до ревизии также реализует команду обновления, но предлагает больше возможностей.

E.2.3. Обновить до ревизии

```
svn info URL_рабочей_копии
svn update [-r rev] [-N] [--ignore-externals] ПУТЬ
```

Если помечен флажок Обновить только папку верхнего уровня, используйте параметр `-N`.

Если помечен флажок Пропустить внешние, используйте параметр `--ignore-externals`.

E.2.4. Фиксировать

В TortoiseSVN диалог фиксации использует несколько команд Subversion. Первая стадия - это проверка статуса, которая определяет элементы вашей рабочей копии, которые потенциально могут быть зафиксированы. Вы можете просмотреть этот список, сравнить файлы с их базой и выбрать элементы, которые вы желаете включить в фиксацию.

```
svn status -v ПУТЬ
```

Если отмечен флажок Показать неверсированные файлы, TortoiseSVN также будет показывать неверсированные файлы и папки в иерархии рабочей копии, учитывая правила игнорирования. Конкретно это свойство не имеет прямого эквивалента в Subversion, поскольку команда `svn status` не заходит в неверсированные папки.

Если вы отметите какие-либо неверсированные файлы и папки, эти элементы сначала будут добавлены к вашей рабочей копии.

```
svn add ПУТЬ...
```

Когда вы нажимаете на OK, Subversion начинает выполнение фиксации. Если вы оставили все флажки, отмечающие файлы, в состоянии по умолчанию, TortoiseSVN использует одну рекурсивную фиксацию рабочей копии. Если вы сняли пометки с некоторых файлов, тогда должна использоваться нерекурсивная фиксация (`-N`), и каждый путь должен быть указан индивидуально в командной строке для фиксации.

```
svn commit -m "СообщениеЖурнала" [-N] [--no-unlock] ПУТЬ...
```

`СообщениеЖурнала` здесь представляет собой содержимое поля ввода сообщения журнала. Оно может быть пустым.

Если помечен флажок Сохранить блокировки, используйте параметр `--no-unlock`.

E.2.5. Различие

```
svn diff ПУТЬ
```

Если вы используете команду 'Различия' из главного контекстного меню, вы сравниваете изменённый файл с его базовой ревизией. Вывод из ИКС вышеприведенной команды тоже это выполняет и производит выдачу в формате объединённых различий. Однако, TortoiseSVN это не использует. TortoiseSVN применяет TortoiseMerge (или программу сравнения по вашему выбору) для наглядного отображения различий между текстовыми файлами, поэтому прямого эквивалента ИКС нет.

Вы можете также сравнить любые два файла при помощи TortoiseSVN, независимо от того, находятся ли они под управлением версиями. TortoiseSVN просто скармливает эти два файла в выбранную программу сравнения и позволяет ей определить, где находятся различия.

E.2.6. Журнал

```
svn log -v -r 0:N --limit 100 [--stop-on-copy] ПУТЬ
  или
svn log -v -r M:N [--stop-on-copy] ПУТЬ
```

По умолчанию, TortoiseSVN пытается извлечь 100 сообщений журнала, используя метод --limit. Если установки заставляют использовать старые API, тогда для получения сообщений журнала для 100 ревизий из хранилища используется вторая форма.

Если отмечен флажок Остановиться на копировании/переименовании, используйте параметр `--stop-on-copy`.

E.2.7. Проверка на наличие изменений

```
svn status -v ПУТЬ
  или
svn status -u -v ПУТЬ
```

Начальная проверка статуса смотрит только на вашу рабочую копию. Если вы нажмёте на Проверить хранилище, тогда проверяется также и хранилище, чтобы посмотреть, какие файлы будут изменены при обновлении, и это требует параметра `-u`.

Если отмечен флажок Показать неверсированные файлы, TortoiseSVN также будет показывать неверсированные файлы и папки в иерархии рабочей копии, учитывая правила игнорирования. Конкретно это свойство не имеет прямого эквивалента в Subversion, поскольку команда `svn status` не заходит в неверсированные папки.

E.2.8. Граф ревизий

Граф ревизий - это возможность, предоставляемая только TortoiseSVN. Аналога в клиенте командной строки нет.

Что TortoiseSVN при этом делает:

```
svn info URL_рабочей_копии
svn log -v URL
```

где URL - это *корень* хранилища, и затем анализирует возвращённые данные.

E.2.9. Обозреватель хранилища

```
svn info URL_рабочей_копии
```

```
svn list [-r rev] -v URL
```

Вы можете использовать `svn info` для определения корня хранилища: это верхний уровень, отображаемый в обозревателе хранилища. Вы не можете перемещаться `выше` этого уровня. Также, эта команда возвращает всю информацию о блокировках, отображаемую в обозревателе хранилища.

Вызов `svn list` покажет список содержимого папки, для указанного URL и ревизии.

E.2.10. Редактировать конфликты

Эта команда не имеет эквивалента в ИКС. Она вызывает TortoiseMerge или внешний инструмент трёхстороннего различия/слияния для просмотра файлов, вовлечённых в конфликт и отбора строк, которые должны быть использованы.

E.2.11. Улажено

```
svn resolved ПУТЬ
```

E.2.12. Переименовать

```
svn rename ТЕКУЩИЙ_ПУТЬ НОВЫЙ_ПУТЬ
```

E.2.13. Удалить

```
svn delete ПУТЬ
```

E.2.14. Убрать изменения

```
svn status -v ПУТЬ
```

Первая стадия - проверка статуса, определяющая элементы в вашей рабочей копии, в которых потенциально могут убраны изменения. Вы можете просмотреть список, сравнить файлы с базой и выбрать элементы, в которых вы желаете убрать изменения.

Когда вы нажмёте на ОК, Subversion уберёт изменения. Если вы оставили все флажки выбора файлов в состоянии по умолчанию, TortoiseSVN использует одиночную рекурсивную (`-R`) отмену изменений в рабочей копии. Если вы снимете пометки с некоторых файлов, тогда каждый путь должен быть указан индивидуально в командной строке для удаления изменений.

```
svn revert [-R] ПУТЬ...
```

E.2.15. Очистка

```
svn cleanup ПУТЬ
```

E.2.16. Заблокировать

```
svn status -v ПУТЬ
```

Первая стадия - проверка статуса, при которой определяются файлы в вашей рабочей копии, которые потенциально могут быть заблокированы. Вы можете выбрать элементы, которые вы желаете заблокировать.

```
svn lock -m "СообщениеБлокировки" [--force] ПУТЬ...
```

`СообщениеБлокировки` представляет собой содержимое поля сообщения блокировки. Оно может быть пустым.

Если отмечен флажок Перехватить блокировки, используйте параметр `--force`.

E.2.17. Снятие блокировки

```
svn unlock ПУТЬ
```

E.2.18. Ответвление/Метка

```
svn copy -m "СообщениеЖурнала" URL URL
  или
svn copy -m "СообщениеЖурнала" URL@rev URL@rev
  или
svn copy -m "СообщениеЖурнала" ПУТЬ URL
```

Диалог Ответвление/Метка выполняет копирование в хранилище. Есть 3 переключаемые кнопки:

- Ведущая ревизия в хранилище (HEAD)
- Указанной ревизии в хранилище
- Рабочая копия

которые соответствуют трем вышеприведённым вариантам командной строки.

`СообщениеЖурнала` здесь представляет собой содержимое поля ввода сообщения журнала. Оно может быть пустым.

E.2.19. Параметр

```
svn info URL_рабочей_копии
svn switch [-r rev] URL ПУТЬ
```

E.2.20. Слияние

```
svn merge [--dry-run] --force От_URL@revN До_URL@revM ПУТЬ
```

Кнопка Пробное выполняет такое же слияние с параметром `--dry-run`.

```
svn diff От_URL@revN До_URL@revM
```

Кнопка Объедин. различия показывает результат операции различия, который будет использован для выполнения слияния.

E.2.21. Экспорт

```
svn export [-r rev] [--ignore-externals] URL ПУТЬ_Экспорта
```

Эта форма используется, когда вызывается из неверсированной папки, и эта папка используется как адресат.

Экспорт рабочей копии в другое место делается без использования библиотеки Subversion, так что подходящего эквивалента командной строки нет.

Всё, что делает TortoiseSVN, - это копирует все файлы в новое место, показывая ход выполнения операции. Дополнительно также могут быть экспортированы неверсированные файлы/папки.

В обоих случаях, если помечен флажок Игнорировать внешние, используйте параметр `--ignore-externals`.

E.2.22. Перебазировать

```
svn switch --relocate Из_URL На_URL
```

E.2.23. Создать здесь хранилище

```
svnadmin create --fs-type fsfs ПУТЬ
```

E.2.24. Добавить

```
svn add ПУТЬ...
```

Если вы выделили папку, TortoiseSVN сначала рекурсивно просмотрит её для поиска элементов, которые могут быть добавлены.

E.2.25. Импорт

```
svn import -m СообщениеЖурнала ПУТЬ URL
```

`СообщениеЖурнала` здесь представляет собой содержимое поля ввода сообщения журнала. Оно может быть пустым.

E.2.26. Авторство (Blame)

```
svn blame -r N:M -v ПУТЬ
svn log -r N:M ПУТЬ
```

Если вы используете TortoiseBlame для просмотра информации об авторстве, также потребуется журнал файла для отображения сообщений журнала в всплывающих подсказках. Если вы просматриваете эту информацию как текстовый файл, этот файл не требуется.

E.2.27. Добавить в список игнорирования

```
svn propget svn:ignore ПУТЬ > tempfile
{внести новый игнорируемый элемент в tempfile}
svn propset svn:ignore -F tempfile ПУТЬ
```

Поскольку в свойстве `svn:ignore` часто содержится несколько строк, здесь показано, как его изменять через текстовый файл, а не непосредственно через командную строку.

E.2.28. Создать заплатку

```
svn diff ПУТЬ > файл-заплатка
```

TortoiseSVN создаёт файл заплатки в формате объединённых различий, сравнивая рабочую копию с её базовой версией.

E.2.29. Применить заплатку

Применение заплатки - хитрое дело, если только у заплатки и рабочей копии не одна и та же ревизия. К счастью для вас, можно использовать программу TortoiseMerge, которая не имеет прямого эквивалента в Subversion.

Приложение F. Подробности реализации

Это приложение содержит более подробное описание реализации некоторых возможностей TortoiseSVN.

F.1. Пометки на значках

Каждому файлу и папке соответствует значение статуса Subversion, сообщаемое библиотекой Subversion. В клиенте командной строки статус обозначается однобуквенным кодом, но в TortoiseSVN они показываются графически, в виде пометок на значках. Поскольку количество пометок сильно ограничено, каждая пометка может представлять один из нескольких значений статуса.

Пометка *Конфликтующее* применяется для обозначения состояния `конфликта`, в котором обновление или переключение привело к конфликту между локальными изменениями и изменениями, полученными из хранилища. Она также используется для обозначения `мешающего` состояния, которое может возникнуть при невозможности завершения операции.

Пометка *Изменённое* обозначает состояние `изменено` (вы сделали локальные изменения), состояние `слито` (изменения из хранилища были слиты с локальными), и состояние `замещён` (файл был удалён и заменён другим, отличающимся файлом с таким же именем).

Пометка *Удалённое* обозначает состояние `удалено`, в котором элемент запланирован для удаления, либо `отсутсвующее` состояние, когда элемент отсутствует. Конечно же, пометки у отсутствующего элемента быть не может, но эта пометка может быть у родительской папки, когда отсутствует один из её дочерних элементов.

Пометка *Добавленное* используется просто для обозначения состояния `добавлено`, когда элемент был добавлен под управление версиями.

Пометка *В Subversion* применяется для обозначения элемента в `нормальном` состоянии, а также используется для версированного элемента, чьё состояние пока неизвестно. Поскольку TortoiseSVN использует фоновый кэширующий процесс для получения статуса, может потребоваться несколько секунд для обновления пометки.

Пометка *Требуется блокировка* применяется для обозначения того, что у файла установлено свойство `svn:needs-lock`. Для рабочих копий, созданных при помощи Subversion 1.4.0 или более поздних версий, состояние `svn:needs-lock` кэшируется локально Subversion, что и используется для определения того, когда показывать пометку. Для рабочих копий в формате версий до 1.4.0, TortoiseSVN показывает эту пометку, когда файл имеет статус только-для-чтения. Обратите внимание, Subversion автоматически обновляет рабочие копии при выполнении

операции обновления, хотя кэширование свойства `svn:needs-lock` может не произойти, пока не будет обновлён сам файл.

Пометка *Заблокированное* применяется для файла, который был заблокирован из локальной рабочей копии.

Пометка *Игнорируемое* используется для обозначения элемента в `игнорируемом` состоянии, возникающем вследствие применения или глобального шаблона игнорирования, или свойства `svn:ignore` родительской папки. Это необязательная пометка.

Пометка *Неверсированное* используется просто для обозначения элемента в `неверсированном` состоянии. Это элемент, располагающийся в версированной папке, но в то же время сам не находящийся под управлением версиями. Это необязательная пометка.

Если элемент имеет статус Subversion `none` (элемент вне рабочей копии), то пометка не показывается. Если вы отключили пометки *Игнорируемое* и *Неверсированное*, то и для этих файлов никаких пометок показано не будет.

У элемента может быть только одно значение статуса Subversion. Например, файл может быть изменён локально и в то же время помечен для удаления. Subversion вернёт единственное значение статуса - в данном случае `удалён`. Эти приоритеты определены внутри самой Subversion.

Когда TortoiseSVN показывает статус рекурсивно (это настройка по умолчанию), для каждой папки выбирается пометка, отражающая её собственное состояние и состояние всех её дочерних элементов. Для того, чтобы показать одну *итоговую* пометку, мы используем вышеприведённый порядок приоритетов для определения того, какую пометку показывать, при этом пометка *Конфликтующее* имеет наивысший приоритет.

Вы можете обнаружить, что не все из этих пометок фактически используются в вашей системе. Это происходит из-за того, что число пометок, доступных в Windows, ограничено 15-ю. Из них 4 использует Windows, остальные 11 могут быть использованы другими приложениями. Если доступных позиций для размещения пометок недостаточно, TortoiseSVN попытается быть «Добропорядочным Гражданином (TM)» и ограничит своё использование пометок, оставляя эту возможность и другим программам.

- *Нормальный*, *Изменено* и *Конфликтующее* всегда загружаются и отображаются.
- *Удалено* загружается по возможности, и отображается как *Изменено*, если свободных позиций не хватает.
- *Только-для-чтения* загружается по возможности, или же отображается как *Нормальный*, когда свободных позиций не хватает.
- *Заблокировано* загружается, только если уже загружено меньше 13 пометок. Отображается как *Нормальный*, если свободных позиций не хватает.
- *Добавлено* загружается, только если уже загружено менее 14 пометок. Показывается как *Изменено*, когда не хватает свободных позиций.

Приложение G. Организация защиты Svnserve при помощи SSH

В этом разделе приведено руководство, рассказывающее о том, как шаг за шагом настроить Subversion и TortoiseSVN для использования протокола `svn+ssh`. Если вы уже используете соединения, аутентифицируемые SSH, для входа на ваш сервер, то вам это руководство не пригодится и вы можете найти дополнительные подробности в книге о Subversion. Если вы пока не начали использовать SSH, но хотели бы это сделать для защиты вашей установки Subversion, это руководство предоставит простой метод, который не требует создания на сервере для каждого пользователя Subversion отдельной пользовательской учётной записи для доступа через SSH.

В этой реализации мы создаём одну учётную запись пользователя для доступа через SSH для всех пользователей Subversion, и используем различные ключи аутентификации для разграничения между реальными пользователями Subversion.

В этом приложении мы предполагаем, что у вас уже установлены инструменты Subversion, и что вы уже создали хранилище, как описано в другом месте руководства. Обратите внимание: вы *не должны* запускать svnserve как службу или демон при использовании с SSH.

Большая часть приведённой здесь информации поступила из обучающего документа, подготовленного Марком Логеманном (Marc Logemann), который может быть найден по адресу *www.logemann.org* [http://www.logemann.org/2007/03/13/subversion-tortoisesvn-ssh-howto/] Дополнительная информация по настройке севера Windows была предоставлена Торстеном Мюллером (Thorsten Müller). Спасибо, ребята!

G.1. Настройка Linux-сервера

Вам необходимо включить SSH на сервере, и здесь мы предполагаем, что вы будете использовать OpenSSH. В большинстве дистрибутивов оно уже установлено. Чтобы убедиться, наберите:

```
ps xa | grep sshd
```

и посмотрите наличие работ ssh.

Ещё один момент: если вы собирали Subversion из исходного кода и не указывали аргументов для `./configure`, Subversion создаёт папку `bin` в `/usr/local` и помещает свои исполняемые файлы туда. Если вы собираетесь использовать режим туннелирования с SSH, вы должны понимать, что пользователь, зашедший через SSH, должен запустить программу svnserve и некоторые другие исполняемые файлы. По этой причине, либо поместите `/usr/local/bin` в переменную `PATH`, либо создайте символьные ссылки ваших исполняемых файлов в папке `/usr/sbin`, или в любой другой папке, которая обычно есть в `PATH`.

Чтобы проверить, что всё в порядке, зайдите как целевой пользователь через SSH и наберите:

```
which svnserve
```

Эта команда должна продемонстрировать, что svnserve доступна.

Создайте нового пользователя, которого мы будем использовать для доступа к хранилищу svn:

```
useradd -m svnuser
```

Убедитесь, что предоставили этому пользователю права на полный доступ к хранилищу.

G.2. Настройка Windows-сервера

Установите демона SSH из Cygwin, как описано здесь: *http://pigtail.net/LRP/printsrv/cygwin-sshd.html*

Создайте новую учётную запись пользователя Windows `svnuser`, которую мы будем использовать для доступа к хранилищу. Убедитесь, что предоставили этому пользователю права на полный доступ к хранилищу.

Если файла паролей пока нет, создайте его из консоли Cygwin при помощи следующей команды:

```
mkpasswd -l > /etc/passwd
```

G.3. Инструменты клиента SSH для использования с TortoiseSVN

Скачайте инструменты, необходимые для использования клиента SSH под Windows, с этого сайта: *http://www.chiark.greenend.org.uk/~sgtatham/putty/* Просто идите в раздел загрузок (download) и загрузите `Putty`, `Plink`, `Pageant` и `Puttygen`.

G.4. Создание сертификатов OpenSSH

Следующим шагом будет создание пары ключей для аутентификации. Есть два возможных способа для создания ключей. Первый - создать ключи при помощи PuTTYgen на клиенте, загрузить открытый ключ на ваш сервер и использовать секретный ключ с PuTTY. Другой - создать пару ключей с помощью инструмента OpenSSH ssh-keygen, сказать секретный ключ на ваш клиент и преобразовать его в совместимый с PuTTY тип.

G.4.1. Создание ключей при помощи ssh-keygen

Зайдите на сервер под `root` или `svnuser` и наберите:

```
ssh-keygen -b 1024 -t dsa -N парольная_фраза -f файл_ключа
```

подставив настоящую фразу пароля (которую знаете только вы) и файл ключа. Мы только что создали SSH2 DSA ключ с 1024-битной ключевой фразой. Если вы наберёте

```
ls -l файл_ключа*
```

вы увидите два файла, `файл_ключа` и `файл_ключа.pub`. Как вы можете догадаться, файл `.pub` является файлом открытого ключа, другой - секретного.

Присоедините открытый ключ к уже существующим в папке `.ssh` в домашней папке пользователя `svnuser`:

```
cat файл_ключа.pub >> /home/svnuser/.ssh/authorized_keys
```

Для того, чтобы мы могли использовать сгенерированный секретный ключ, необходимо преобразовать его в формат putty. Это необходимо, поскольку формат файла секретного ключа стандартом не определяется. После того, как вы скачали секретный ключ на ваш клиентский ПК, запустите PuTTYgen и выберите Conversions → Import key. Перейдите к вашему файлу `файл_ключа`, который вы получили с сервера, и введите фразу пароля, которую вы использовали при создании ключа. В завершение щёлкните на Save private key и сохраните файл как `файл_ключа.PPK`.

G.4.2. Создание ключей при помощи PuTTYgen

Используйте PuTTYgen для генерации пары открытый-ключ/секретный-ключ и сохраните их. Скопируйте открытый ключ на сервер и присоедините его к уже существующим в папке `.ssh` в домашней папке пользователя `svnuser`:

```
cat файл_ключа.pub >> /home/svnuser/.ssh/authorized_keys
```

G.5. Проверка при помощи PuTTY

Для проверки соединения мы воспользуемся PuTTY. Запустите программу и во вкладке Session задайте имя сервера или его ip-адрес в поле Host name (or IP address), протокол (connection type) - SSH и сохраните сессию как `SvnConnection` или под любым другим именем, которое вам нравится. На вкладке SSH укажите предпочитаемую версию протокола SSH (preferred SSH protocol version) как 2 и во вкладке Auth задайте полный путь до секретного ключа `.PPK`, который вы получили ранее. Вернитесь ко вкладке Sessions и нажмите кнопку Save. Теперь вы увидите `SvnConnection` в списке сохранённых сессий (saved sessions).

Щёлкните на Open и вы должны увидеть приглашение ко входу в стиле telnet. Используйте `svnuser` в качестве имени пользователя (user name) и если всё хорошо, вы должны тотчас же подключиться без запроса пароля.

Возможно, вам потребуется отредактировать `/etc/sshd_config` на сервере. Исправьте строки как указано далее и после этого перезапустите службу SSH.

```
PubkeyAuthentication yes
PasswordAuthentication no
PermitEmptyPasswords no
ChallengeResponseAuthentication no
```

G.6. Проверка SSH при помощи TortoiseSVN

Пока мы проверили только то, что вы можете зайти на сервер при помощи SSH. Теперь нам надо убедиться, что SSH-соединение действительно может выполнять svnserve. На сервере измените `/home/svnuser/.ssh/authorized_keys` как указано далее, чтобы разрешить нескольким авторам Subversion использовать одну и ту же системную учётную запись, `svnuser`. Обратите внимание: каждый автор Subversion использует одну и ту же учётную запись, но различные ключи аутентификации, таким образом, вы должны добавить по одной строке для каждого автора.

Заметьте: всё это вводится одной, очень длинной, строкой.

```
command="svnserve -t -r <ПутьККорнюХранилищ>
            --tunnel-user=<автор>",
            no-port-forwarding,no-agent-forwarding,no-X11-forwarding,
            no-pty ssh-rsa <ОткрытыйКлюч> <Комментарий>
```

Есть несколько значений, которые вы должны указать в соответствии с вашими настройками.

`<ПутьККорнюХранилищ>` должно быть заменено на путь к папке, содержащей ваши хранилища. Это позволяет избежать указания полных путей на сервере внутри URL. Заметьте, вы должны использовать прямую косую черту даже на сервере Windows, например, `c:/svn/reposroot`. В примерах ниже мы предполагаем, что папка вашего хранилища расположена в корне для хранилищ, называемого `repos`.

`<автор>` должно быть заменено автором svn, который будет регистрироваться при фиксации. Это также позволяет svnserve использовать собственные права доступа из `svnserve.conf`.

<ОткрытыйКлюч> должно быть заменено открытым ключом, сгенерированным вами ранее.

<Комментарий> может быть любым комментарием по вашему выбору, но бывает полезно связать имя автора svn с реальным именем человека.

Щёлкните правой кнопкой на любой папке в Проводнике Windows и выберите TortoiseSVN → Обозреватель хранилища. У вас будет запрошен URL, введите его в следующем виде:

```
svn+ssh://svnuser@SvnConnection/repos
```

Что означает этот URL? Имя схемы `svn+ssh` говорит TortoiseSVN как обрабатывать запросы на сервер. После двойной косой черты вы указываете пользователя для соединения с сервером, в нашем случае `svnuser`. После `@` мы указываем имя нашей сессии PuTTY. Эта сессия содержит все подробности, вроде того, где искать секретный ключ и ip-адрес или имя сервера. Наконец, мы должны указать путь к хранилищу относительно корня хранилищ на сервере, как указано в файле `authorized_keys`.

Щёлкните на OK и вы должны получить возможность просматривать содержимое хранилища. Если получилось, значит теперь у вас есть работающий SSH-туннель в соединении с TortoiseSVN.

Обратите внимание: по умолчанию TortoiseSVN для соединения использует собственную версию Plink. Это позволяет избежать появления консольного окна при каждой попытке аутентификации, но это также означает, что сообщениям об ошибке негде появляться. Если вы получите ошибку «Unable to write to standard output» (невозможно записать в стандартный вывод), вы можете попробовать указать Plink в качестве клиента в сетевых настройках TortoiseSVN. Это позволит вам увидеть настоящую ошибку, выдаваемую Plink.

G.7. Варианты конфигурации SSH

Один из способов упростить URL в TortoiseSVN - указать пользователя внутри сессии PuTTY. Для этого вы должны загрузить вашу уже подготовленную сессию `SvnConnection` в PuTTY и во вкладке Connection - Data указать имя пользователя в поле Auto login user, например, `svnuser`. Сохраните вашу сессию PuTTY как раньше и попробуйте следующий URL в TortoiseSVN:

```
svn+ssh://SvnConnection/repos
```

В этот раз мы указываем только сессию PuTTY `SvnConnection` клиенту SSH, используемому TortoiseSVN (TortoisePlink.exe). Этот клиент получит из сессии все необходимые подробности.

На время написания этих строк PuTTY не проверяет все сохранённые конфигурации, поэтому, если у вас есть несколько конфигураций с одним и тем же именем сервера, она выберет первую же подходящую. Также, если отредактировать и сохранить конфигурацию по умолчанию, имя пользователя для автоматического входа *не сохранится*.

Многим нравится использовать Pageant для хранения своих ключей. Поскольку сессия PuTTY способна сохранять ключ, вам не всегда нужен Pageant. Но представьте, что вы храните ключи для нескольких различных серверов; в этом случае вам придётся редактировать сессию PuTTY снова и снова, в зависимости от сервера, к которому вы собираетесь подключиться. В этой ситуации Pageant определённо имеет смысл, поскольку когда PuTTY, Plink, TortoisePlink или любой другой основанный на PuTTY инструмент пытается соединиться с сервером SSH, он проверяет все секретные ключи, содержащееся в Pageant, для установления соединения.

Для просто запустите Pageant и добавьте секретный ключ. Это должен быть тот же самый секретный ключ, что вы задавали в сессии PuTTY выше. Если вы используете Pageant для содержания секретных ключей, вы можете удалить ссылку на файл секретного ключа в вашей сохранённой сессии PuTTY. Конечно же, вы можете добавить дополнительные ключи для других серверов, или для других пользователей.

Если вы не желаете повторять эту процедуру после каждой перезагрузки клиента, вы должны поместить Pageant в группу автозапуска вашей установки Windows. Вы можете добавить ключи, каждый с полным путём, как параметры командной строки Pageant.exe

Последний способ подсоединиться к серверу SSH - просто использовать этот URL в TortoiseSVN:

```
svn+ssh://svnuser@100.101.102.103/repos
svn+ssh://svnuser@mydomain.com/repos
```

Как вы видите, мы не используем сохранённой сессии PuTTY, но IP-адрес (или имя домена) как цель соединения. Мы также указали пользователя, но вы можете поинтересоваться, как же будет найден секретный ключ. Поскольку TortoisePlink.exe является всего лишь слегка изменённой версией стандартного инструмента Plink из набора PuTTY, TortoiseSVN также будет пробовать все ключи, хранящиеся в Pageant.

Если вы воспользовались этим последним методом, убедитесь, что у вас в PuTTY не задано имя по умолчанию. Нам сообщали об ошибках в PuTTY, приводящих в этом случае к закрытию соединения. Чтобы убрать пользователя по умолчанию, просто очистите `HKEY_CURRENT_USER\Software\SimonTatham\Putty\Sessions\Default%20Settings\HostName`

Глоссарий

BDB	Berkeley DB. Хорошо протестированная база данных для хранилища, которая не может быть использована на сетевых разделяемых ресурсах. В версиях до 1.2 используется для хранилищ по умолчанию.
FSFS	Собственная внутренняя файловая система Subversion для хранилищ. Может быть использована на сетевых разделяемых ресурсах. Используется по умолчанию для хранилищ, начиная с версии 1.2.
GPO	Объект групповой политики
SVN	Часто используемое сокращение для Subversion. Имя собственного протокола Subversion, используемого сервером хранилища «svnserve».
Авторство (Blame)	Эта команда предназначена только для текстовых файлов. Она снабжает каждую строку информацией о том, в какой ревизии хранилища строка была последний раз изменена, и об авторе, выполнившем это изменение. Наша реализация с графическим интерфейсом пользователя называется TortoiseBlame и показывает также дату/время фиксации и сообщение журнала при наведении указателя мыши на номер ревизии.
Базовая ревизия (BASE)	Текущая базовая ревизия файла или папки в вашей *рабочей копии*. Эта та ревизия, в которой файл или папка были при последнем извлечении, обновлении или фиксации. Базовая ревизия обычно не равна ведущей (HEAD) ревизии.
Блокировка	Когда вы выполняете блокировку версированного элемента, вы помечаете его в хранилище как недоступный для фиксации никому, кроме той рабочей копии, которая произвела блокировку.
Ведущая ревизия (HEAD)	Последняя ревизия файла или папки в *хранилище*.
Добавить	Команда Subversion, которая используется для добавления файла или папки в вашу рабочую копию. Эти новые элементы будут добавлены в хранилище при фиксации.
Журнал	Показывает историю ревизий файла или папки. Также известен как «История».
Заплатка (Patch)	Если рабочая копия содержит изменения только в текстовых файлах, можно воспользоваться командой 'Различия' Subversion для генерации одного сводного файла, содержащего все эти изменения в формате объединённых различий. Файлы этого типа часто называются «Заплатками», и они могут быть отправлены по электропочте кому-либо ещё (или в список рассылки) и применены к другой рабочей копии. Кто-нибудь без возможности фиксировать изменения может подготовить изменения и отправить файл заплатки для применения лицу, обладающему правом фиксации. Или, если нет уверенности в изменении, можно отправить заплатку другим для рецензирования.

Извлечь	Команда Subversion, создающая локальную рабочую копию в пустой папке путём загрузки версированных файлов из хранилища.
Импорт	Команда Subversion для импорта в хранилище целой иерархии папок за одну ревизию.
История	Показывает историю ревизий файла или папки. Также известна как «Журнал».
Конфликт	Когда изменения из хранилища сливаются с локальными, иногда случается, что они затрагивают одни и те же строки. В этом случае Subversion не может самостоятельно решить, какую из версий использовать и файл считается находящимся в конфликтном состоянии. Вы должны вручную отредактировать файл и уладить конфликты перед тем, как вы сможете зафиксировать любые дальнейшие изменения.
Копирование	В хранилище Subversion вы можете создать копию одного файла или целого дерева. Это реализовано через «лёгкие копии», которые ведут себя подобно ссылке на оригинал: они почти не занимают места. Создание копии сохраняет историю элемента в копии, так что вы можете отследить изменения, сделанные до создания копии.
Обновить	Эта команда Subversion вносит последние изменения из хранилища в вашу рабочую копию, сливая все сделанные другими изменения с локальными изменениями из рабочей копии.
Ответвление	Термин, часто используемый в системах управления версиями для описания ситуации, когда разработка разветвляется в определённой точке и следует по двум различным путям. Можно создать ответвление из основной линии разработки - для реализации новой возможности без приведения главной линии в нестабильное состояние, или можно ответвить стабильную версию, в которой делаются только исправления ошибок, в то время как новые разработки ведутся в нестабильном стволе. В Subversion ответвления реализованы как «лёгкие копии».
Очистка	Цитата из Книги о Subversion: «Рекурсивно очищает рабочую копию, удаляя блокировки и возобновляя незаконченные действия. Если вы когда-либо столкнётесь с ошибкой *рабочая копия заблокирована*, то запустите эту команду для удаления подвисших блокировок и приведения вашей рабочей копии в работоспособное состояние.» Заметьте, что в этом контексте *блокировка* относится к блокированию в локальной файловой системе, а не в хранилище.
Параметр	Как «обновить-до-ревизии» изменяет временное окно, через которое смотрит рабочая копия, на другую точку в истории, так и «переключить» изменяет пространственное окно рабочей копии так, чтобы оно указывало на другую часть хранилища. Это особенно полезно при работе в стволе и ответвлениях, отличающихся всего несколькими файлами. Вы можете переключать вашу рабочую копию между двумя местами, и передаваться будут только изменённые файлы.
Перебазировать	Когда ваше хранилище перемещается, возможно из-за переноса в другую папку на сервере, или изменяется доменное имя

	сервера, вам необходимо «перебазировать» вашу рабочую копию, чтобы содержащийся в ней URL хранилища указывал на новое местоположение. Обратите внимание: вы должны использовать эту команду только в случае, если ваша рабочая копия относится к тому же местоположению в том же хранилище, но само хранилище было перемещено. В любом другом случае вам, вероятно, вместо этой команды необходимо использовать команду «Переключить».
Рабочая копия	Это ваша локальная «песочница», область, где вы работаете с версированными файлами, и она обычно находится на вашем локальном жёстком диске. Вы создаёте рабочую копию путём «извлечения» из хранилища, и возвращаете изменения назад в хранилище при помощи «фиксации».
Различие	Обозначение для «Показать различия». Очень полезно, когда вы желаете увидеть, какие именно изменения были сделаны.
Ревизия	Каждый раз при фиксации набора изменений создаётся новая «ревизия» в хранилище. Каждая ревизия представляет состояние дерева хранилища в определённой точке его истории. При желании, вы можете вернуться назад во времени и изучить хранилище, каким оно было в ревизии N. Иначе говоря, на ревизию можно ссылаться как на набор изменений, которые были сделаны при создании этой ревизии.
Свойство	Помимо версирования ваших папок и файлов, Subversion позволяет добавлять версированные метаданные, - называемые «свойствами», - к каждому версированному файлу или папке. У каждого свойства есть имя и значение, почти как у ключа реестра. В Subversion есть несколько специальных свойств для внутреннего использования, таких как `svn:eol-style`. И у TortoiseSVN тоже есть такого рода свойства, вроде `tsvn:logminsize`. Вы можете добавить ваши собственные свойства с именем и значением по вашему выбору.
Свойство ревизии (revprop)	Так же как и файлы, каждая ревизия в хранилище может обладать свойствами. Некоторые специальные свойства добавляются автоматически при создании ревизии, а именно `svn:date svn:author svn:log`, представляющие дату/время фиксации, того, кто её произвёл и сообщение журнала, соответственно. Эти свойства могут быть отредактированы, но они не версируются, поэтому любое изменение является постоянным и не может быть отменено.
Слияние	Процесс, при помощи которого изменения из хранилища добавляются в вашу рабочую копию без разрушения уже сделанных в ней изменений. Иногда изменения не могут быть приведены в соответствие автоматически, и рабочая копия считается находящейся в состоянии конфликта. Слияние случается автоматически при обновлении рабочей копии. Вы можете слить отдельные изменения из другого ответвления при помощи команды слияния TortoiseSVN.
Убрать изменения	Subversion локально сохраняет «нетронутую» копию каждого файла, каким этот файл был при последнем обновлении рабочей копии. Если вы сделали какие-то изменения и решили их

	откатить, вы можете использовать команду «Убрать изменения» для того, чтобы вернуться к нетронутой копии.
Удалить	Когда вы удаляете версированный элемент (и фиксируете это изменение), элемент больше не существует в хранилище после зафиксированной ревизии. Но, конечно, он всё ещё существует в более ранних ревизиях хранилища, так что вы всё ещё можете получить к нему доступ. При необходимости, вы можете скопировать удалённый элемент и «воскресить» его вместе с историей.
Уладить	В случае, когда файлы в рабочей копии находятся в состоянии конфликта после слияния, эти конфликты должны быть разрешены человеком при помощи редактора (или, возможно, TortoiseMerge). Этот процесс называется «улаживанием конфликтов». После его выполнения вы можете отметить конфликтующие файлы как улаженные, что позволит их зафиксировать.
Фиксировать	Эта команда Subversion используется для передачи изменений из вашей рабочей копии обратно в хранилище, при этом создавая в нём новую ревизию.
Хранилище	Хранилище - центральное место, где данные хранятся и обслуживаются. Хранилище может быть местом, в котором размещаются несколько баз данных или файлов для распространения по сети, или хранилище может быть местом, которое непосредственно доступно пользователю без необходимости блуждания по сети.
Экспорт	Эта команда создаёт копию версированной папки, почти как рабочая копия, но без локальных папок `.svn`.

Предметный указатель

Symbols

A

C

F

G

I

M

N

S

T

U

V

W