Настройки - TortoiseSVN

Клиент Subversion для Windows

Version 1.10

Стефан Кюнг
Люббэ Онкен
Саймон Ладж

Настройки - TortoiseSVN: Клиент Subversion для Windows: Version 1.10

Стефан Кюнг, Люббэ Онкен, Саймон Ладж
Перевод: Vladimir Serdyuk (vserd@users.sourceforge.net), Станислав Петраков (stannic@gmail.com), Siarhei Niakhai (s.niakhai@gmail.com)

Дата публикации 2018/03/17 15:14:17 (r28148)

Содержание

Список иллюстраций

Список таблиц

Предисловие

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. Что такое TortoiseSVN?

TortoiseSVN — это бесплатный Windows-клиент с открытыми исходным кодом для системы управления версиями *Apache™ Subversion®*. То есть TortoiseSVN управляет файлами и директориями во времени. Файлы хранятся в центральном *хранилище*. Хранилище больше похоже на обычный файловый сервер, кроме того он запоминает каждое изменение когда-либо сделанное в ваших файлах и директориях. Это позволяет вам восстановить старые версии ваших файлов и проверить историю изменений — как, когда и кто изменял ваши данные. Вот почему многие думают о Subversion, и вообще о системах управления версиями, как о своего рода «машине 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.

2. 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.

Графический интерфейс пользователя
: При просмотре списка изменений файла или папки вы можете кликнуть на ревизию, чтобы увидеть комментарии для этой фиксации. Также доступен список измененных файлов - всего лишь сделайте двойной клик на файле, чтобы увидеть, какие конкретно изменений были внесены.

  Диалог фиксации это список, в котором перечислены все файлы и папки, которые будут включены в фиксацию. У каждого элемента списка имеется флажок, чтобы вы могли выбрать именно то, что вы хотите включить в фиксацию. Неверсированные файлы также могут быть представлены в этом списке, чтобы вы не забыли добавить в фиксацию новый файл или папку.

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.

3. Лицензия

"TortoiseSVN это открытое программное обеспечение, разработанное под лицензией GNU (GPL). Оно бесплатно как для скачивания, так и для персонального или коммерческого использования на любом количестве компьютеров.

Несмотря на то, что большинство загружают лишь установочный файл, вы имеете полный доступ на чтение к исходному коду этой программы. Вы можете просмотреть его по ссылке https://sourceforge.net/p/tortoisesvn/code/HEAD/tree/. Самая последняя версия, над которой мы работаем в данный момент, находится в `/trunk/`, ранее выпущенные версии находятся в `/tags/`.

4. Разработчики

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.

4.1. 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 исходный код TortoiseCVS и использовал его как основу для TortoiseSVN. Потом он основал проект, зарегистрировал сайт `tortoisesvn.org` и выложил исходный код в Интернет.

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 руководить переводами.

Со временем другие системы управления версиями получили свои собственные Tortoise-клиенты, что привело к проблеме с оверлейными иконками в проводнике: количество таких оверлеев ограничено и даже один Tortoise-клиент легко может превысить этот лимит. Тогда Stefan Küng реализовал компонент TortoiseOverlays, который позволяет всем Tortoise-клиентам использовать одни и те же оверлейные иконки. Теперь все Tortoise-клиенты с открытым исходным кодом и даже некоторые не Tortoise-клиенты используют этот общедоступный компонент.

4.2. 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 иконки, логотип, отлов ошибок, за перевод и координацию деятельности по переводу

Саймону Ладжу (Simon Large)
: за работу над документацией

Stefan Fuhrmann
: за кеширование журнала и граф ревизий

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 и новые идеи. И за помощь, оказанную другим в виде ответов в нашем списке рассылки

Нашим дарителям
: за многие часы удовольствия от присланной нам музыки

5. Структура книги

Эта книга написана для тех, кто, владея компьютерной грамотой, хочет использовать Subversion для управления своими данными, но чувствует себя неуютно, применяя для этого клиенты командной строки.

Поскольку TortoiseSVN - расширение оболочки Windows, предполагается, что пользователь знаком с Проводником Windows и знает, как его использовать.

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

В главе Глава 1, *Приступая к работе* излагается, как установить TortoiseSVN на ваш компьютер и сразу начать им пользоваться.

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.

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 6, *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`.

6. Используемая терминология

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

Выбор меню обозначен стрелкой. TortoiseSVN → Журнал означает: выберите *Журнал* в контекстном меню *TortoiseSVN*.

Использование контекстного меню обозначается следующим образом: Контекстное меню → Сохранить как...

Кнопки пользовательского интерфейса обозначаются так: Нажмите OK для продолжения.

Действия пользователя обозначены полужирным шрифтом.**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 другое место.

Ввод и вывод командной строки также показан при помощи `специального` шрифта.

Важно

Важные примечания отмечены таким значком.

Подсказка

Делают вашу жизнь проще.

Внимание

Нужно быть особенно осторожным.

Предупреждение

Необходимо проявить максимальную внимательность. При игнорировании таких предупреждений возможно повреждение данных или другие неприятности.

Глава 1. Приступая к работе

Этот раздел написан для тех, кто хотел бы разобраться, что же такое TortoiseSVN и первый раз им воспользоваться. Раздел объясняет, как установить TortoiseSVN и как настроить локальное хранилище, а также дает обзор самых часто используемых операций.

1.1. Установка TortoiseSVN

1.1.1. Требования к системе

TortoiseSVN работает под операционной системой Vista или выше, и доступен как в 32-битной, так и в 64-битной версии. Установщик для 64-битной Windows также включает 32-битную часть. Что означает что вы не должны устанавливать 32-битную версию отдельно чтобы контекстное меню и оверлей TortoiseSVN работало в 32-битных приложениях.

Поддержка Windows 98, Windows ME и Windows NT4 была прекращена начиная с версии 1.2.0 и Windows 2000 и XP до SP2 начиная с 1.7.0. Поддержка Windows XP с SP3 была прекращена начиная с версии 1.9.0. Вы все еще можете загрузить и установить старые версии, если они вам понадобятся.

1.1.2. Установка

TortoiseSVN поставляется в виде простого в использовании установочного файла. Сделайте на нем двойной клик и следуйте инструкциям - остальное он сделает за вас. Не забудьте перезагрузить компьютер после установки.

Важно

Вам необходимы привилегии администратора для установки TortoiseSVN. Установщик запросит данные администратора при необходимости.

Доступны языковые пакеты, которые переводят интерфейс TortoiseSVN на множество языков. Пожалуйста, посетите Приложение G, *Языковые пакеты и проверка правописания* для дополнительной информации о том, как их установить.

If you encounter any problems during or after installing TortoiseSVN please refer to our online FAQ at https://tortoisesvn.net/faq.html.

1.2. Основная концепция

Перед тем, как мы погрузимся в работу с настоящими файлами, важно получить представление о том, как работает Subversion и какие термины используются.

Хранилище
: Subversion использует центральную базу данных, которая содержит все ваши версированные файлы с их полной историей. Эта база данных называется *хранилищем*. Хранилище обычно находится на файловом сервере, на котором установлен Subversion, по запросу поставляющий данные клиентам Subversion (например, TortoiseSVN). Если вы делаете резервное копирование, то копируйте ваше хранилище, так как это оригинал всех ваших данных.

Рабочая копия
: Это именно то место, где вы работаете. Каждый разработчик имеет собственную рабочую копию, иногда называемую песочницей, на своем локальном компьютере. Вы можете получить из хранилища последнюю версию файлов, поработать над ней локально, никак не взаимодействуя с кем-либо еще, а когда вы будете уверены в изменениях вы можете зафиксировать эти файлы обратно в хранилище.

  Рабочая копия не содержит историю проекта, но содержит копию всех файлов, которые были в хранилище до того, как вы начали делать изменения. Это обозначает, что можно легко узнать, какие конкретно изменения вы сделали.

Вам также нужно знать, где найти TortoiseSVN, потому что в меню "Пуск" его нет. Это так потому, что TortoiseSVN - расширение проводника Windows, поэтому для начала нужно запустить проводник. Сделайте правый клик на папке в проводнике и вы увидите новые пункты в контекстном меню, такие как эти:

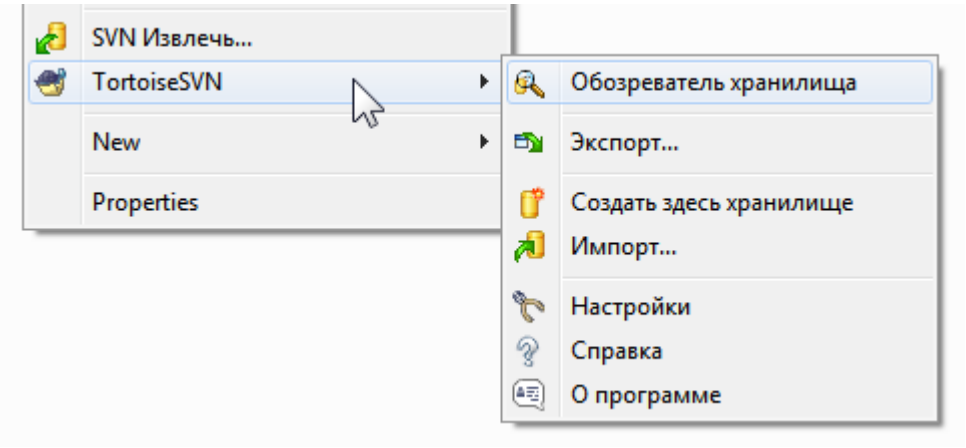


Рисунок 1.1. Меню TortoiseSVN для неверсированных папок

1.3. Тест-драйв

Этот раздел познакомит вас с наиболее используемыми функциям на примере маленького тестового хранилища. Он, конечно, не объясняет всего - это ведь всего лишь краткое руководство. После ознакомления вам нужно запастись временем, чтобы прочесть оставшуюся часть данного руководства, которая более детально освещает функции приложения. Она к тому же объясняет как правильно настроить сервер Subversion.

1.3.1. Создание хранилища

Для настоящего проекта вам понадобится хранилище,созданное в безопасном месте, и сервер Subversion, чтобы управлять им. Для обучения мы будем использовать функцию локального хранилища Subversion, которая разрешает прямой доступ к хранилищу, созданного на вашем жестком диске, и не требует наличия сервера.

Сначала создайте новую пустую директорию на вашем ПК. Она может быть где угодно, но в этом руководстве мы собираемся назвать её `C:\svn_repos`. Теперь сделайте правый клик на новой папке и в контекстном меню выберите TortoiseSVN → Создать здесь хранилище.... Хранилище, созданное внутри папки, готово к использованию. Также мы создадим внутренную структуру папок нажав кнопку Создать структуру каталогов.

Важно

Функция локального хранилища очень полезна для тестирования и оценивания, но в других случаях, за исключением случая когда вы единственный разработчик работающий за одним компьютером, вы всегда должны использовать настроенный сервер Subversion. В маленьких компаниях, конечно, заманчиво избежать работы по созданию и настройке сервера и работать просто с общим сетевым ресурсом. Никогда так не делайте! Вы потеряете данные. Прочтите Раздел 3.1.4, «Доступ к хранилищу на сетевом ресурсе» и узнаете, почему это является плохой идеей и как создать и настроить сервер.

1.3.2. Импорт проекта

Сейчас у нас есть хранилище, но оно совершенно пустое в данный момент. Давайте предположим, что у меня есть набор файлов в `C:\Projects\Widget1`, который я хотел бы добавить. Перейдите к папке

`Widget1` в Проводнике и сделайте правый клик на ней. Теперь выберите пункт TortoiseSVN → Импорт..., который вызовет диалог

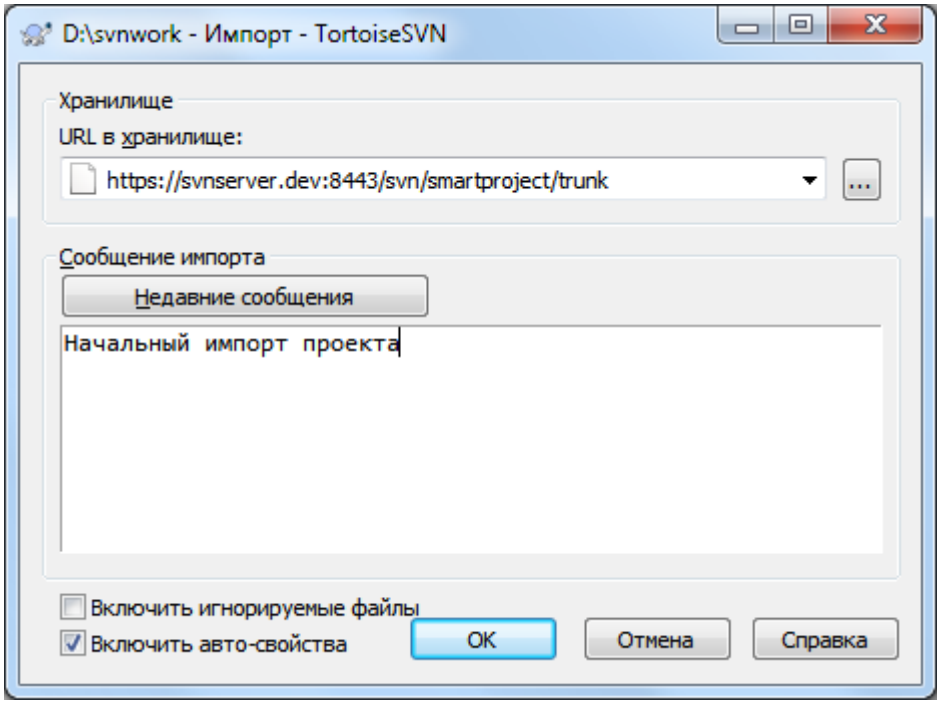


Рисунок 1.2. Диалог импорта

К хранилищу Subversion обращаются по URL-адресу, который позволяет нам указать хранилище где угодно в Интернете. В данном случае нам нужно указать на наше локальное хранилище, которое имеет URL-адрес `file:///c:/svn_repos/trunk`, и к которому мы добавляем имя нашего проекта `Widget1`. Обратите внимание, что после `file:` есть 3 слэша и везде используются прямые слэши.

Другая важная функция данного диалога - это окно Сообщение импорта, в которое вы можете добавить сообщение о том, что вы делаете. Когда вам понадобится просмотреть историю проекта, эти сообщения будут ценным подспорьем для просмотра какие изменения и когда были сделаны. В нашем случае мы напишем что-нибудь простое как «Импорт проекта Виджет1». Нажмите OK, чтобы добавить папку в ваше хранилище.

1.3.3. Извлечение рабочей копии

Сейчас у нас есть проект в нашем хранилище, и нам надо создать рабочую копию для текущей работы. Заметьте, что имортирование папки не превращает автоматически эту папку в рабочую копию. Для создания свежей рабочей копии в Subversion используется термин `Извлечь`. Мы собираемся извлечь папку Widget1 из нашего хранилища в папку для разработки называемую `C:\Projects\Widget1-Dev`.

Создайте эту папку, затем сделате правый клик на ней и выберите пункт TortoiseSVN → Извлечь.... Затем введите URL-адрес для извлечения, в данном случае `file:///c:/svn_repos/trunk/Widget1`, и кликните на OK. Наша папка для разработки заполнится файлами из хранилища.

Важно

При настройках по умолчанию пункт меню "Извлечь" размещён не в подменю TortoiseSVN, а показан на верхнем уровне меню проводника. Команды TortoiseSVN, которые находятся не в подменю начинаются с `SVN`: SVN Извлечь...

Вы заметите что внешний вид этой папки отличается от обычной папки. У каждого файла появился зелёный флажок в левом углу. Это значки статуса TortoiseSVN, которые присутствуют только в рабочей копии. Зелёный статус означает, что файл не отличается от версии файла, находящегося в хранилище.

1.3.4. Внесение изменений

Можно приступать к работе. В папке `Widget1-Dev` мы начинаем редактировать файлы - предположим, мы вносим изменения в файлы `Widget1.c` и `ReadMe.txt`. Обратите внимание, что значки на этих файлах теперь стали красными и показывают, что изменения были сделаны локально.

Но какие были изменения? Нажмите правой кнопкой на одном из изменённых файлов и выберите команду TortoiseSVN → Различия. Запустится инструмент TortoiseSVN для сравнения файлов и покажет какие точно строки в файлах были изменены.

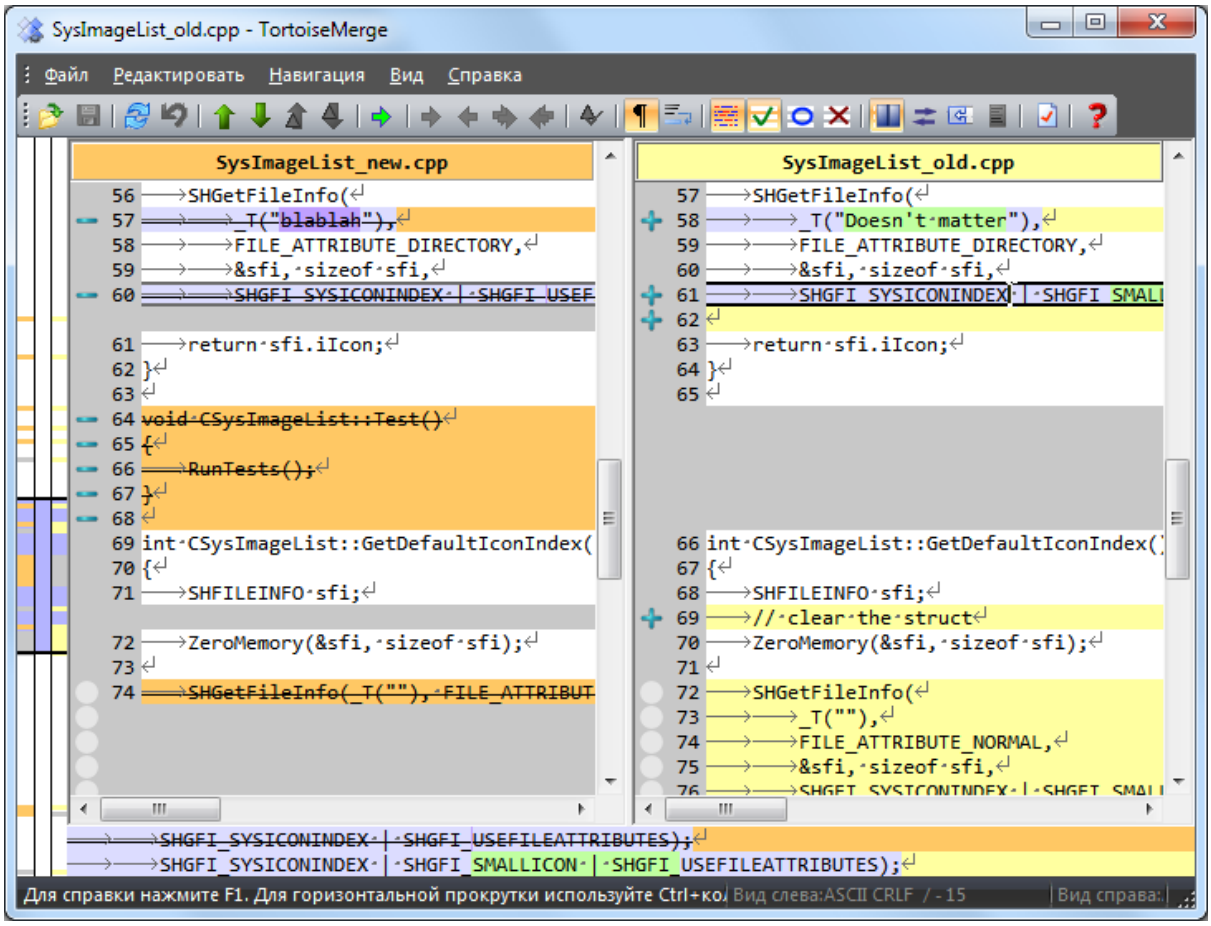


Рисунок 1.3. Просмотрщик изменений в файлах

Ок, нас устраивают изменения, поэтому давайте обновим хранилище. Это действие называется `Фиксировать` изменения. Нажмите правой кнопкой на папке `Виджет1-Дев` и выберите команду TortoiseSVN → Фиксировать. Появится диалог фиксации со списком изменённых файлов и напротив каждого будет галочка. Вы можете выбрать лишь несколько файлов из списка для фиксации, но в нашем случае мы будем фиксировать изменения в обоих файлах. Введите сообщение с описанием сделанных изменений и нажмите OK. Появится диалог с прогрессом процесса фиксации файлов в хранилище и мы закончили фиксацию.

1.3.5. Добавление новых файлов

Во время работы над проектом, вам понадобится добавлять новые файлы - предположим вы добавили новые функции в файле `Экстра.c` и добавили справку в существующем файле `Создатьфайл`. Нажмите правой кнопкой на папке и выберите команду TortoiseSVN → Добавить. Диалог добавления показывает все неверсированные файлы и вы можете выбрать те файлы, которые вы хотите добавить. Другой способ добавления файлов - это нажать правой кнопкой на самом файле и выбрать команду TortoiseSVN → Добавить.

Теперь, если вы откроете папку для фиксации, новый файл будет отображаться как *Добавлен* и существующий файл как *Изменён*. Обратите внимание, что вы можете дважды нажать на изменённый файл, чтобы просмотреть какие именно изменения были сделаны.

1.3.6. Просмотр истории проекта

Одной из самых полезных функций TortoiseSVN является диалоговое окно журнала. Оно показывает список всех фиксаций изменений в файле или папке, а также все подробные сообщения, которые вы вводили при фиксации изменений (вы же *ввели* сообщение фиксации как вам советовали? Если нет, то сейчас вы видите почему это важно).

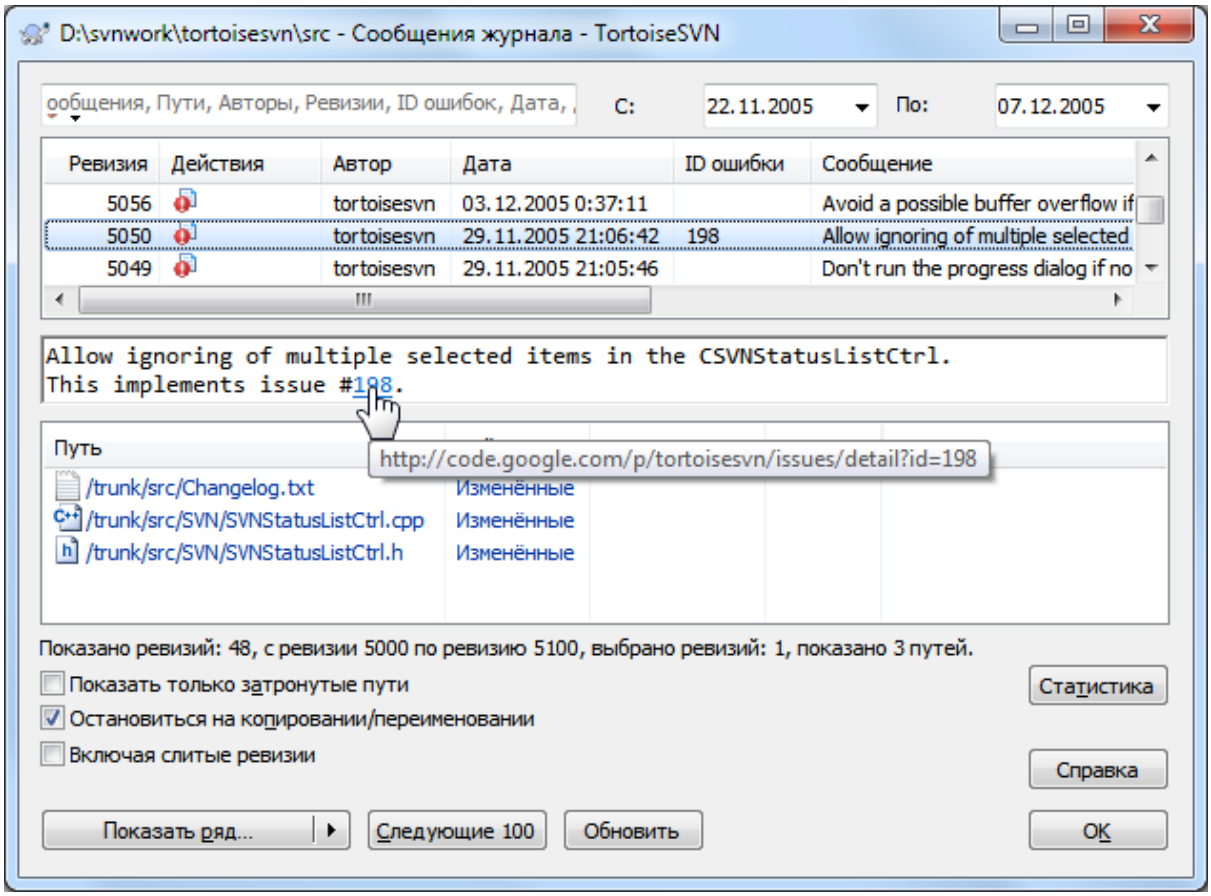


Рисунок 1.4. Диалоговое окно журнала

Ладно, тут я немного схитрил и взял снимок экрана из хранилища TortoiseSVN.

Верхняя панель показывает список всех фиксированных ревизий вместе с началом сообщения фиксации. Если вы выберите одну из этих ревизий, то средняя панель отобразит полное сообщение журнала для той ревизии и нижняя панель покажет список измененных файлов и папок.

У каждой из этих панелей есть контекстное меню, которое предоставляет много других способов использования информации. В нижней панели вы можете дважды нажать на файл, чтобы просмотреть какие именно изменения были внесены в той ревизии. Прочтите Раздел 4.10, «Диалоговое окно журнала ревизий», чтобы узнать больше.

1.3.7. Отмена изменений

Одной общей функцией всех систем управления ревизиями является функция, которая позволяет вам отменить изменения, которые вы внесли ранее. Как вы и догадались, в TortoiseSVN это легко сделать.

Если вы хотите избавиться от изменений, которые вы еще не успели фиксировать и восстановить нужный файл в том виде, в котором он был перед началом изменений, то выберите команду TortoiseSVN → Убрать

изменения. Это действие отменит ваши изменения (в Корзину) и вернет фиксированную версию файла, с которой вы начинали. Если же вы хотите убрать лишь некоторых изменения, то вы можете использовать инструмент TortoiseMerge для просмотра изменений и выборочного удаления измененных строк.

Если вы хотите отменить действия определённой ревизии, то начните с диалогового окна журнала и найдите проблемную ревизию. Выберите команду Контекстное меню → Отменить изменения из этой ревизии и те изменения будут отменены.

1.4. Далее ...

Это руководство кратко ознакомило вас с самыми важными и полезными функциями TortoiseSVN, но, конечно, есть еще много чего с чем мы вас не познакомили. Мы очень рекомендуем вам посвятить часть своего времени, чтобы прочесть данное руководство до конца, особенно Глава 4, *Руководство по ежедневному использованию*, которая содержит более подробную информацию об операциях для ежедневного использования.

Нам пришлось сильно потрудиться, чтобы сделать это руководство информативным и удобочитаемым, но мы признаём, что есть ещё много чего рассказать! Запаситесь временем и не бойтесь пробовать всё на тестовом хранилище по мере продвижения вперёд. Лучший способ изучить что-то - это использовать это!

Глава 2. Основные понятия управления версиями

Эта глава - слегка изменённая версия такой же главы из книги о Subversion. Онлайн 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 идеям, лежащим в основе Subversion, и покажем несколько простых примеров использования Subversion.

Несмотря на то, что примеры из этой главы описывают совместную работу над исходным кодом программ, помните, что Subversion подходит для любых типов файлов, и не является специализированным инструментом программистов.

2.1. Хранилище

Subversion - это централизованная система для совместной работы. В её основе лежит *хранилище*, которое содержит данные в форме *дерева файловой системы* - обычной иерархии файлов и папок. *Клиенты* подключаются к хранилищу, и читают или изменяют эти файлы. Записывая данные, клиент делает информацию доступной для остальных; читая данные, клиент получает информацию от других.

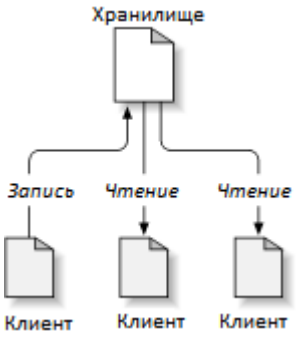


Рисунок 2.1. Типичная система Клиент/Сервер

Но чем эта система отличается от обычного файлового сервера? И действительно, хранилище *является* разновидностью файлового 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 перемещение файлов и каталогов.

Когда клиент просто читает данные из хранилища, он получает последнюю версию дерева файловой системы. Но, помимо этого, клиент имеет возможность посмотреть *предыдущие* состояния файловой системы. Например, клиенту интересно, «Что содержал эта папка в прошлый вторник? » или «Кто последним изменил этот файл и какие изменения внес? » Вопросы подобного типа являются основными для любой *системы управления версиями*: системы, разработанной для записи и отслеживания изменений данных во времени.

2.2. Модели версирования

Все системы управления версиями встают перед вопросом: как система позволит пользователям обмениваться информацией, не мешая друг другу? Ведь довольно просто случайно задеть чужие изменения при совместной работе над одними и теми же данными.

2.2.1. Проблема совместного использования файлов

Предположим такую ситуацию: в компании есть два сотрудника, работающие над одним проектом: Игорь и Света. Они одновременно решили поправить один и тот же файл. Если Игорь сохранит свои изменения первым, тогда Света (сохранившись несколькими секундами позже) может непреднамеренно их переписать своей новой версией файла. Несмотря на то, что версия Игоря не будет потеряна навсегда (потому что система запоминает все изменения), внесённых Игорем правок *не будет* в новой версии файла Светы, ведь она их никогда не видела. Работа Игоря фактически потеряна - или, по крайней мере, отсутствует в последней версии файла. А это как раз то, чего мы хотим избежать!

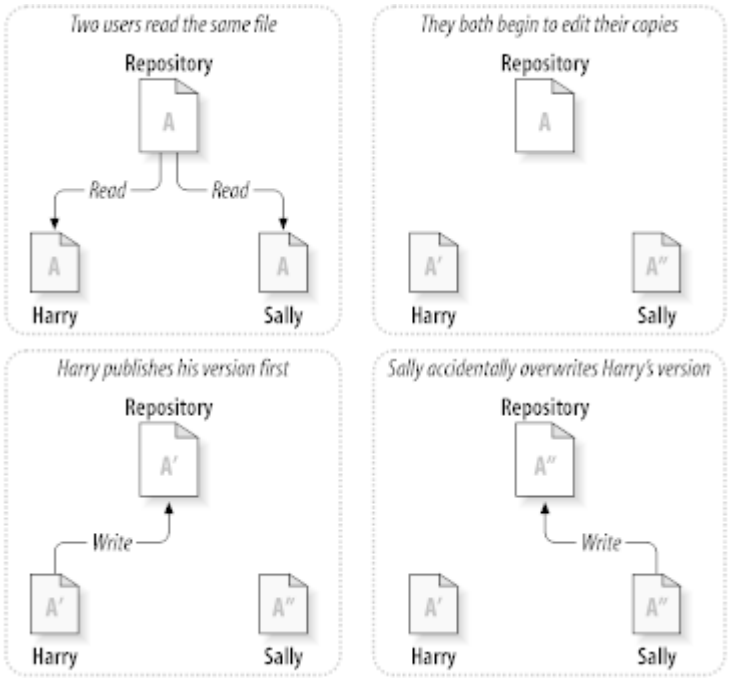


Рисунок 2.2. Проблема потери изменений

2.2.2. Модель Блокирование-Изменение-Разблокирование

Многие системы управления версиями разрешают эту проблему использованием простой модели *блокирование-изменение-разблокирование*. В такой системе хранилище разрешает вносить изменения в файл только одному человеку за раз. До того, как Игорь сможет внести изменения в файл, он должен сначала его *заблокировать*. Блокирование файла подобно взятию книги в библиотеке: если Игорь заблокировал файл, Света не сможет изменить его - хранилище отклонит запрос на блокировку файла. Всё, что она сможет - прочитать файл и ждать, пока Игорь закончит свою работу и снимет блокировку. После того, как Игорь разблокирует файл, Света сможет заблокировать его и внести свои изменения.

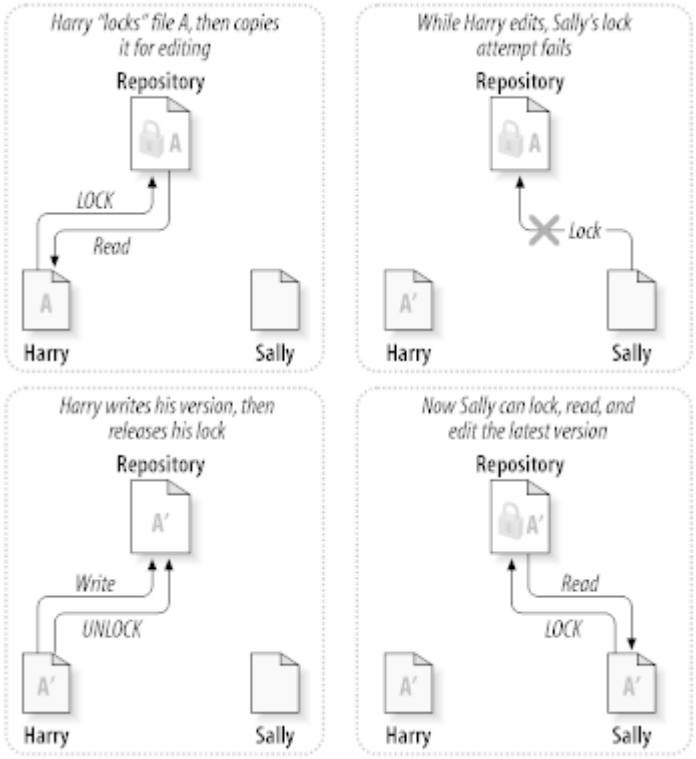


Рисунок 2.3. Модель Блокирование-Изменение-Разблокирование

Проблема с моделью блокирование-изменение-разблокирование в ее жестокости, она может создать следующие неудобства пользователям:

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

- *Блокирование может вызвать излишнюю поочерёдность.* Что, если Игорь редактирует начало большого файла, а Света хочет подправить конец? Эти изменения вообще не пересекаются. Они могли бы легко работать одновременно и никакого вреда это бы не принесло (предполагая корректное слияние изменений).

- *Блокирование может вызвать ложное чувство безопасности.* Предположим, что Игорь заблокировал и редактирует файл А, в то время, как Света заблокировала и редактирует файл Б. Но допустим, что А и Б зависят друг от друга и изменения сделанные в каждом не совместимы. Внезапно А и Б перестают вместе работать. Блокирующая система бессильна в предотвращении проблемы - вместо этого она обеспечила ложное чувство безопасности. Игорь со Светой запросто могут решить, что, блокируя свой файл, каждый начинает безопасную изолированную задачу и это препятствует заблаговременному обсуждению их несовместимых изменений.

2.2.3. Модель Копирование-Изменение-Слияние

Subversion, CVS и другие системы управления версиями используют модель *копирование-изменение-слияние* вместо блокирования. 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, одновременно изменяя свои личные копии. В конце концов, личные копии сливаются в новую, финальную версию. Обычно система управления версиями выполняет слияние автоматически, но, естественно, в общем случае необходимо присутствие человека.

Вот пример: Игорь и Света создали свои рабочие копии одного и того же проекта. Они работают одновременно, и вносят изменения в файл `A` в своих рабочих копиях. Света первой сохраняет свои

изменения в хранилище. Затем, когда Игорь пытается сохранить свои, хранилище информирует его, что его файл A *устарел*. Другими словами, файл A в хранилище был изменён с тех пор, как Игорь получил его. Тогда Игорь выполняет *слияние* (merge) любых изменений хранилища со своей рабочей копией. Вероятно, что изменения Светы не пересекаются с его собственными, и, поскольку теперь его рабочая копия содержит оба набора изменений, он записывает её обратно в хранилище.

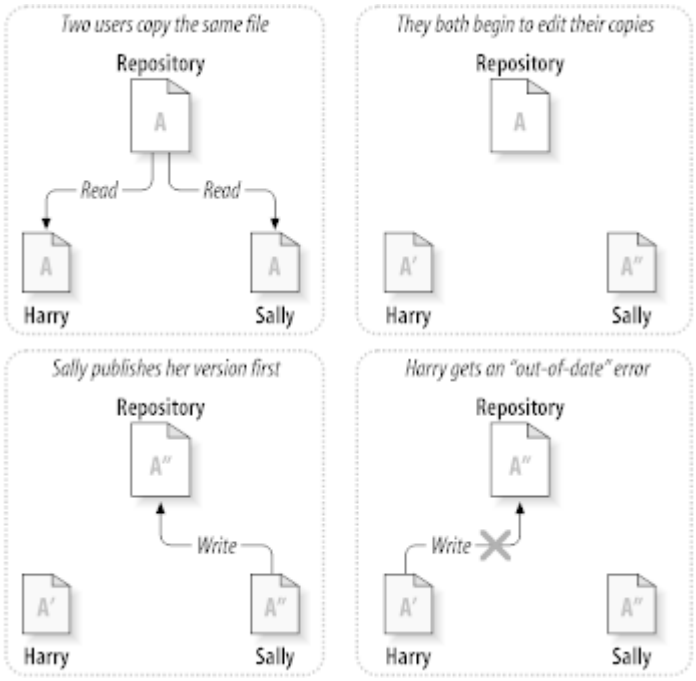


Рисунок 2.4. Модель Копирование-Изменение-Слияние

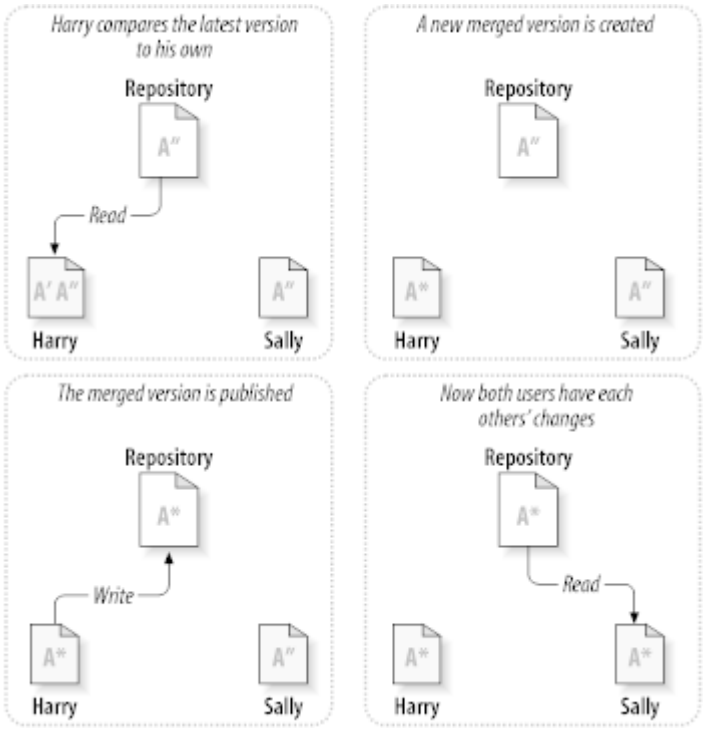


Рисунок 2.5. ...Копирование-Изменение-Слияние. Продолжение

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

Модель копирование-изменение-слияние может выглядеть немного беспорядочно, но на практике все работает гладко. Пользователи могут работать одновременно, не тратя время на ожидание других. Обычно оказывается, что большинство одновременно вносимых изменений в файл вообще не пересекается; конфликты бывают редко. И время, потраченное на их разрешение, значительно меньше времени, отнимаемого блокировками системы.

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

Существует ситуация, когда модель блокирование-изменение-разблокирование оказывается удобнее: если вы имеете дело с файлами, не поддающимися слиянию. Например, если ваше хранилище содержит изображения, и два человека изменяют их в одно и тоже время, то нет возможности слить эти изменения вместе. Всё равно, кто-то потеряет 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 предоставляет 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 содержит несколько дополнительных файлов, созданных и поддерживаемых клиентом Subversion, чтобы помочь ему в работе. Например, ваша рабочая копия содержит подпапку `.svn` - *административную папку* рабочей копии. Файлы в этой папке помогают svn-клиенту распознать, какие файлы содержат неопубликованные изменения, а какие устарели. До версии 1.7 Subversion создавал административные папки `.svn` в каждой подпапке вашей рабочей копии. Subversion 1.7 использует совершенно другой подход. Теперь каждая рабочая копия содержит только одну административную папку в корне рабочей директории.

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.

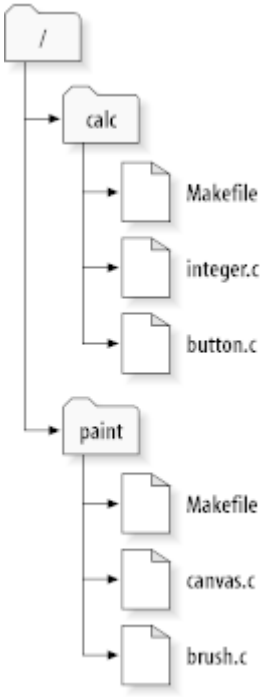


Рисунок 2.6. Файловая система хранилища

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:/path/to/repos
...
file:///X|/path/to/repos
...
```

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

Вы можете получить доступ к хранилищу FSFS через сетевой ресурс, но это *не* рекомендовано по разным причинам:

- Вы даете прямой доступ на запись всем пользователям, таким образом они могут случайно удалить или повредить файловую систему репозитория.
- Не все протоколы для общего доступа к файлам по сети поддерживают блокировку, которую требует Subversion. Однажды вы обнаружите, что ваше хранилище слегка *повреждено*.
- Вы должны настроить разрешения на доступ именно так, как нужно. SAMBA особенно сложен в этом отношении.
- Если кто-то установит более свежую версию клиента, который обновит формат хранилища, то все остальные не смогут получить доступ к хранилищу пока также не обновят клиента до новой верии.

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.

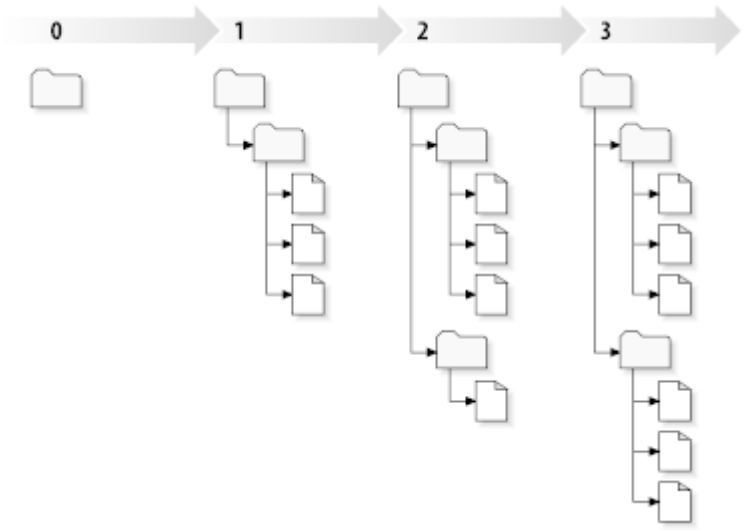


Рисунок 2.7. Хранилище

> **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 -- ревизии N и M одного и того же файла, таким образом, могут *не иметь* отличий.

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

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 fsfs MyNewRepository
```

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

3.1.2. Создание хранилища при помощи TortoiseSVN

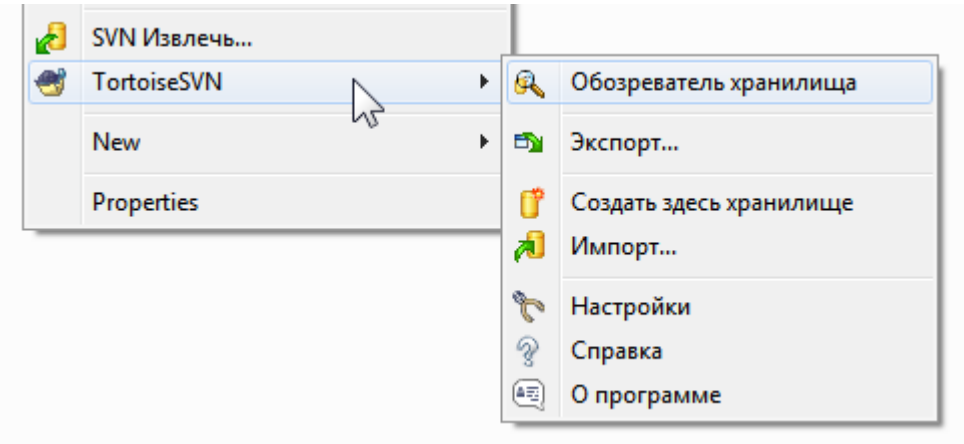


Рисунок 3.1. Меню TortoiseSVN для неверсированных папок

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.

   У вас также спросят хотите ли вы создать структуру директорий внутри хранилища. О возможных вариантах структуры узнайте в Раздел 3.1.5, «Организация данных в хранилище».

   Когда TortoiseSVN создаст хранилище, то установит значок специальной папки, что вы могли проще определить локальные хранилища. Если вы создаете хранилище с помощью официального клиента командной строки, то этот значок папки не назначается.

Подсказка

Мы также рекомендуем, 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.

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 не поддерживаем такое использование по разным причинам:

- 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.
- Если кто-то установит более свежую версию клиента, который обновит формат хранилища, то все остальные не смогут получить доступ к хранилищу пока также не обновят клиента до новой верии.

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
```

Поскольку эта схема часто используется, то когда вы создаете новое хранилище с помощью TortoiseSVN, вам также будет предложено создать структуру директорий.

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.... В диалоге импорта укажите путь к вашему хранилищу и щелкните ОК. 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.

Наиболее простой (но не рекомендуемый) способ - скопировать каталог хранилища на резервный носитель. При этом вы должны быть полностью уверены, что ни один процесс не имеет доступа к данным. Под доступом подразумевается *любой* доступ вообще. Если во время копирования к вашему хранилищу имеет доступ какой-либо процесс (остался открытым веб-браузер, WebSVN и др.), то резервное копирование станет бессмысленным.

Для создания копии вашего хранилища безопасным способом рекомендуется выполнить

```
svnadmin hotcopy путь/к/хранилищу путь/к/резервной/копии
```

После этого резервная копия будет готова.

The `svnadmin` tool is installed automatically when you install the Subversion command line client. The easiest way to get this is to check the option to include the command line tools when installing TortoiseSVN, but if you prefer you can download the latest version of command line tools directly from the *Subversion* [https://subversion.apache.org/packages.html#windows] website.

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/1.8/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 4.31.8, «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`

Переопределение ловушек

Если скрипт ловушек отклоняет вашу фиксацию, то его решение окончательно. Но вы можете создать механизм переопределения внутри самого скрипта используя технику *Волшебное Слово*. Если скрипт хочет отклонить операцию, то он сначала ищет в сообщении журнала специальную фразу, либо заданную фразу, либо имя файла с префиксом. Если он находит волшебное слово, то поволяет фиксации выполнится. Если фраза не найдена, то он может блокировать фиксацию с сообщением «You didn't say the magic word». :-)

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>
```

Of course it would look even better if you included a suitable picture. You can use the *TortoiseSVN logo* [https://tortoisesvn.net/images/TortoiseCheckout.png] or you can provide your own image.

```
<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.

If you are using a public hosting service such as *SourceForge* [https://sourceforge.net] or your server has already been setup by someone else then there is nothing else you need to do. Move along to 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.

Более подробную информацию о серверных настройках Subversion и как выбрать лучшую архитектуру в вашей ситуации вы сможете найти в книге о Subversion в разделе *Server Configuration* [http://svnbook.red-bean.com/en/1.8/svn.serverconfig.html].

In the early days of Subversion, setting up a server required a good understanding of server configuration and in previous versions of this manual we included detailed descriptions of how to set up a server. Since then things have become easier as there are now several pre-packaged server installers available which guide you through the setup and configuration process. These links are for some of the installers we know about:

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

You can always find the latest links on the *Subversion* [https://subversion.apache.org/packages.html] website.

You can find further How To guides on the *TortoiseSVN* [https://tortoisesvn.net/usefultips.html] website.

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 получить максимальную пользу 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. Основные Возможности

Этот раздел описывает некоторые возможности TortoiseSVN, которые касаются практически всего в этом руководстве. Обратите внимание, что многие из этих возможностей будут видны только в рабочей копии Subversion.

4.1.1. Пометки на значках

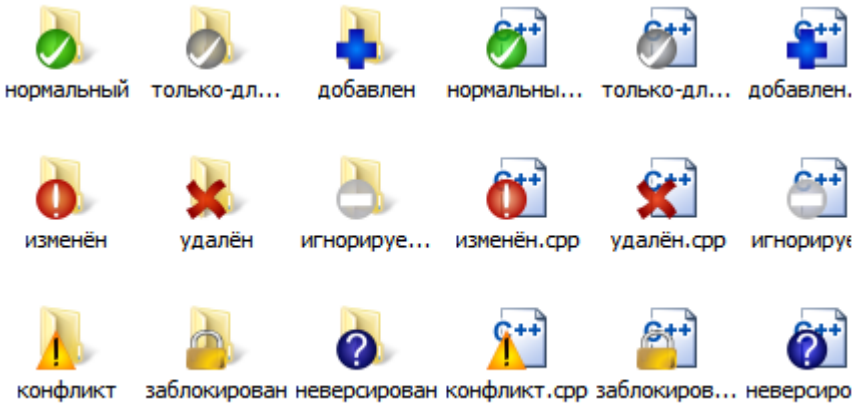


Рисунок 4.1. Проводник с пометками на значках

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

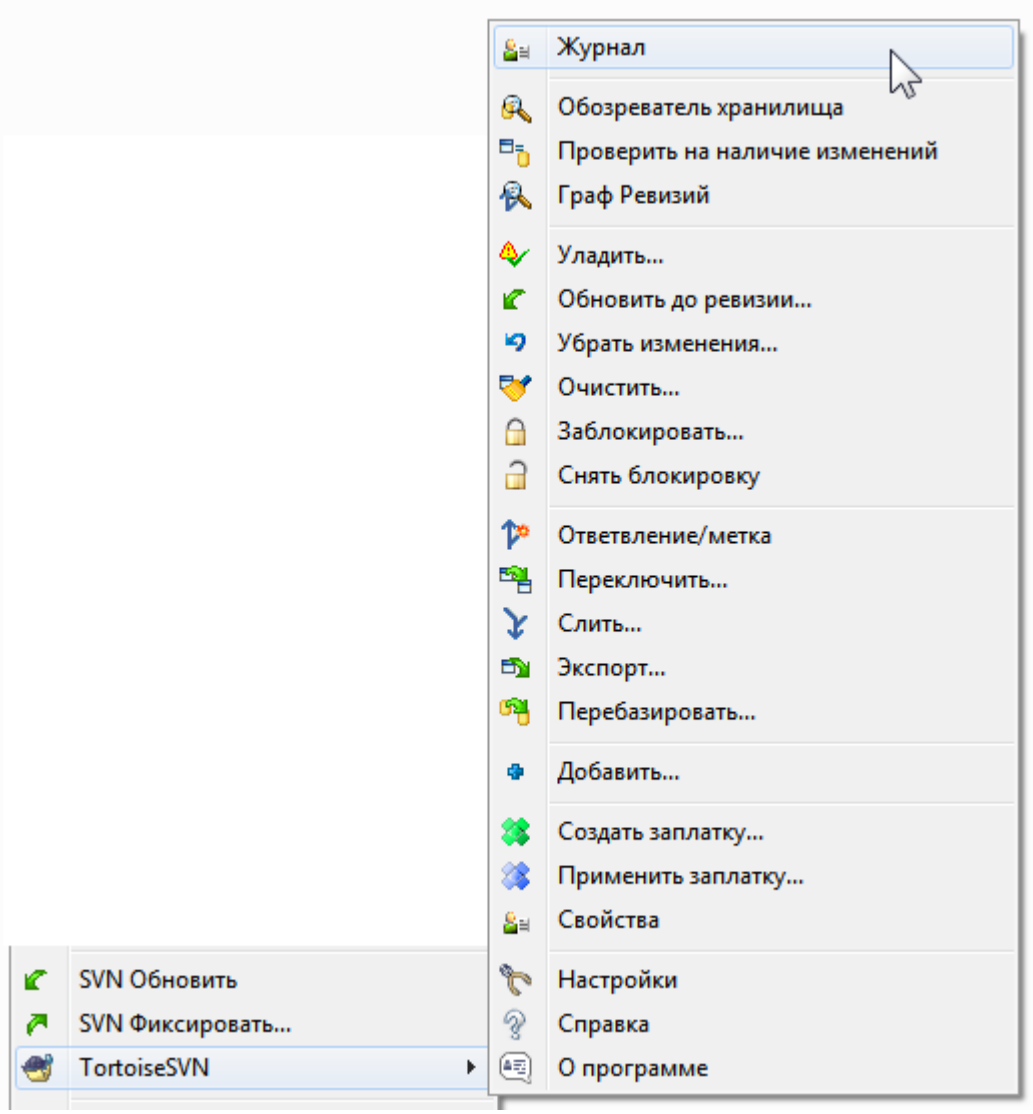


Рисунок 4.2. Контекстное меню для папки, находящейся под управлением версиями

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 вызова расширенного контекстного меню нажмите и держите клавишу **Shift** при правом щелчке мыши.

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

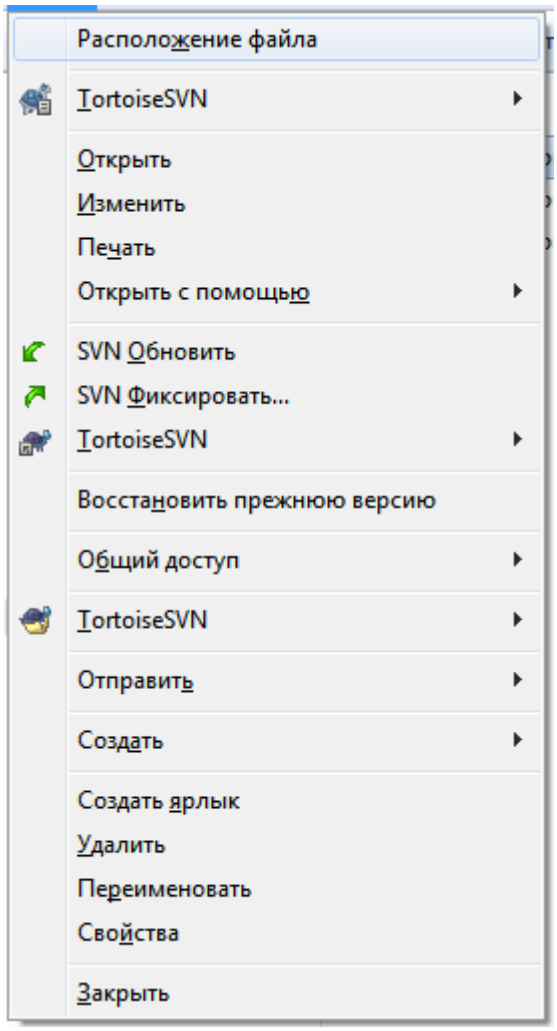


Рисунок 4.3. Меню "Файл" Проводника для ярлыка в версированной папке

Этот пример для неверсированного ярлыка внутри версированной 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.

4.1.3. Перетаскивание мышью

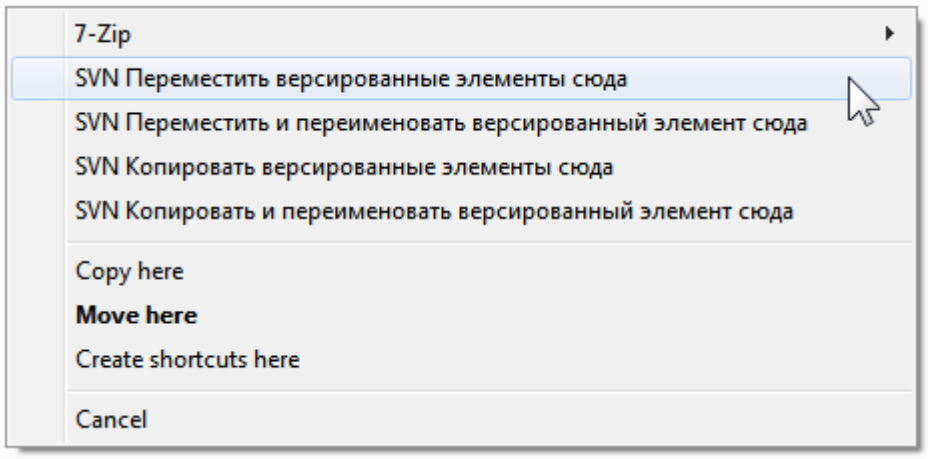


Рисунок 4.4. Меню при перетаскивании правой клавишей мыши для папки под управлением версиями

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
: Скопировать выбранный текст. В случае если текст не выбран, но выбран, например, элемент списка или окно сообщения, то содержимое элемента или окна копируется в буфер обмена.

4.1.5. Аутентификация

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

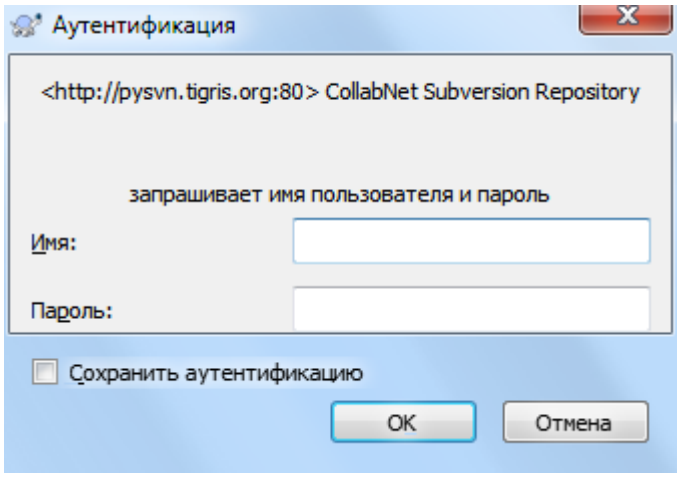


Рисунок 4.5. Диалог аутентификации

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). Обратите внимание: эти пароли хранятся при помощи WinCrypt API, а не в виде простого текста.
- `svn.ssl.server` содержит серверные сертификаты SSL.
- `svn.username` содержит учётные данные для аутентификации только по имени пользователя (без пароля).

Очистить кэш аутентификации можно со страницы Сохранённые данные диалога настроек TortoiseSVN. Кнопка Очистить всё очистит все закэшированные данные аутентификации для всех хранилищ. Кнопка Очистить... тем не менее покажет диалог, в котором вы можете выбрать какие закэшированные данные аутентификации должны быть удалены. Смотрите Раздел 4.31.6, «Настройки сохранённых данных ».

Некоторым людям хочется, чтобы данные об аутентификации удалялись когда они выходят из Windows или при выключении. Это можно сделать с помощью скрипта выключения удалив директорию `%APPDATA%\Subversion\auth`. Например:

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

Описание того как установить такие скрипты вы можете найти в 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 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 Subversion импорт, которая разработана для импортирования в хранилище всей иерархии директории за один раз. Несмотря на то, что эта команда выполняет свою работу, все же 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 не рекомендуем вам использовать команду импорта вообще, а вместо этого лучше воспользоваться двухшаговым методом описанным в Раздел 4.2.2, «Импорт на месте», кроме тех случаев, когда вы создаете простейшую начальную структуру `/trunk /tags /branches` в своем хранилище. Раз уж вы здесь, то вот как работает простейший 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:

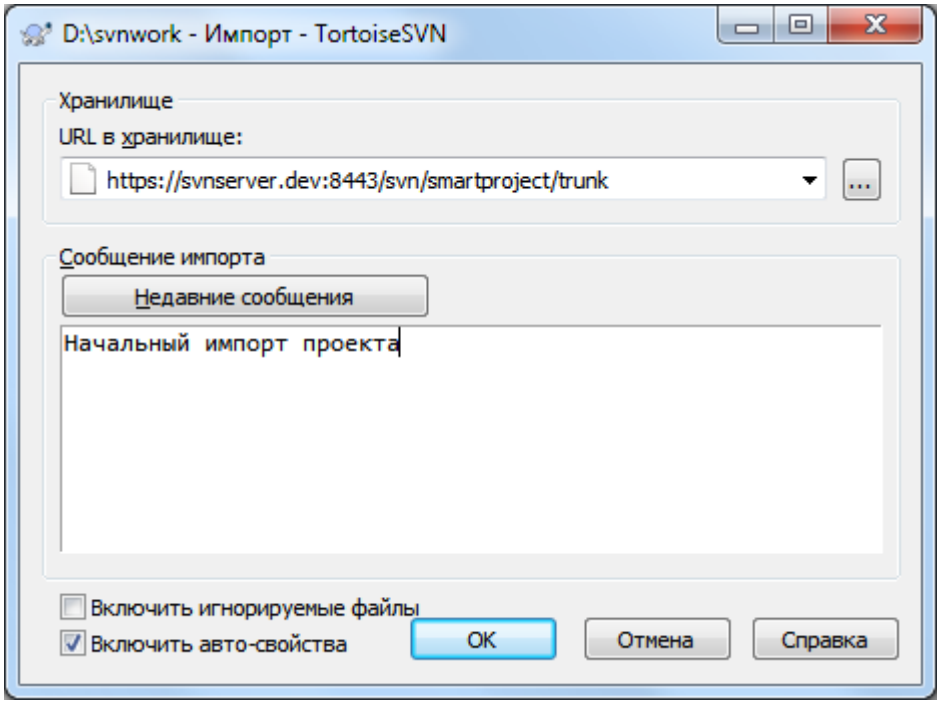


Рисунок 4.6. Диалог импорта

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*. Вы можете изменить это поведение при помощи флажка Включить игнорируемые файлы. Более детальная информация по заданию глобальных шаблонов игнорирования содержится в Раздел 4.31.1, «Общие настройки».

Как только вы нажмёте OK, 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. Чтобы создать папку нового проекта прямо в хранилище используйте обозреватель хранилища. Если вы используете одну из стандартных схем, то возможно вы захотите создать её как подпапку в trunk нежели в корне хранилища. Обозреватель хранилища показывает его структуру прямо как проводник Windows, так что вы можете видеть как там всё организовано.

2. Извлеките эту новую папку поверх той папки, которую вы желаете импортировать. Появится предупреждение о том, что локальная папка не пуста. Не обращайте внимания на предупреждение. 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 `default.build.user`, который не существует в хранилище - только `default.build.user.tmpl`. `default.build.user.tmpl` - это шаблон файла, который каждый разработчик получает из хранилища и переименовывает его в `default.build.user`. 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 `default.build.user` 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:

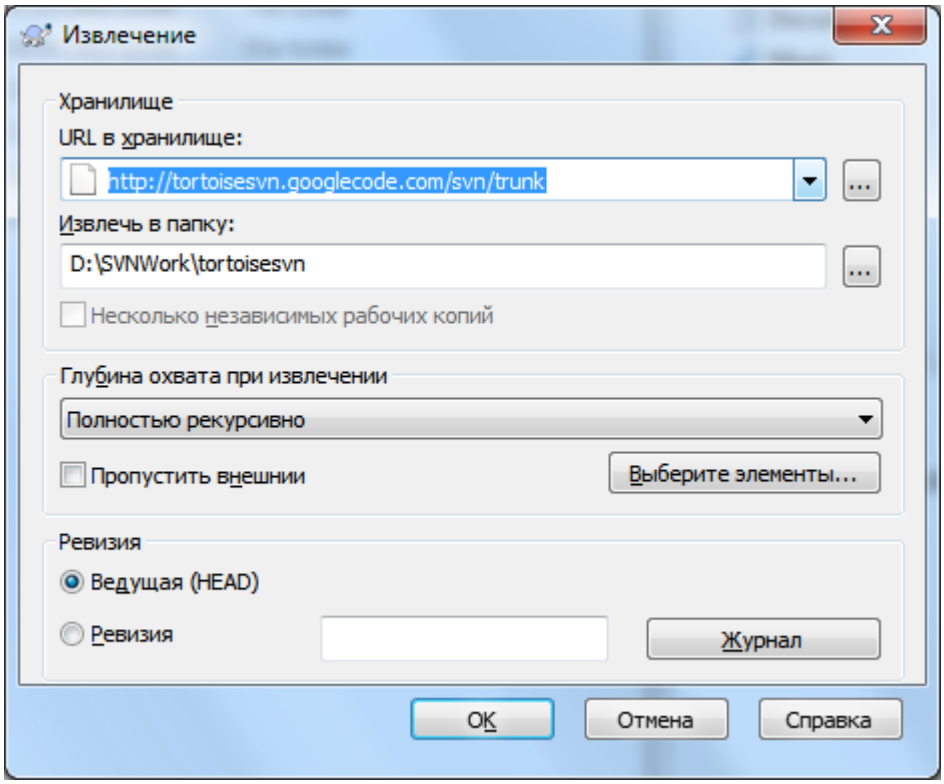


Рисунок 4.7. Диалог извлечения

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

Важно

При настройках по умолчанию пункт меню "Извлечь" размещён не в подменю TortoiseSVN, а показан на верхнем уровне меню проводника. Команды TortoiseSVN, которые находятся не в подменю начинаются с `SVN`: SVN Извлечь...

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.

Для легкого выбора элементов, которые вы хотите фиксировать, и заставить получаемую рабочую копию хранить только эти элементы, кликните кнопку Choose items.... Это действие откроет новый диалог, в котором вы можете отметить все элементы, которые вы хотите хранить в вашей рабочей копии и снять пометку для остальных. Получаемая рабочая копия будет `неполным извлечением`. Обновление такой рабочей копии не будет получать отсутствующие файлы и папки, а только обновлять существующие в рабочей копии.

Если вы извлекаете неполную рабочую копию (т.е. выбирая что-то кроме `полностью рекурсивно` для глубины извлечения), то вы можете легко добавить или удалить подпапки позже используя один из следующих методов.

4.3.1.1. Неполное извлечение используя Обновить до ревизии

Правый клик на извлеченной папке, затем используйте TortoiseSVN → Обновить до ревизии и нажмите Выберите элементы.... При этом откроется такой же диалог, как и при начальном извлечении и позволит вам отобрать или отменить выбор элементов для включения в извлечение. Этот метод очень гибкий, но может оказать медленным, т.к. каждый элемент в папке обновляется по отдельности.

4.3.1.2. Неполное извлечение используя Обозреватель хранилища

Правый клик на извлеченной папке, затем используйте TortoiseSVN → Обозреватель хранилища для вызова обозревателя хранилища. Найдите подпапку, которую вы хотите добавить в вашу рабочую копию, затем выберите Контекстное меню → Обновить элемент до ревизии....

4.3.1.3. Неполное извлечение используя Проверить на наличие изменений

В диалоге проверки на наличие изменений сначала удерживая **shift** 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`. Сделайте правый щелчок на папке (папках), которую вы желали бы добавить в вашу рабочую копию, и выполните 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 есть 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

Однажды вы извлекли рабочую копию на определенную глубину, позже чтобы получить больше или меньше содержания вы можете изменить эту глубину с помощью Context menuОбновить элемент до ревизии...

Использование старого сервера

Серверы версий до 1.5 не понимали запросов с глубиной рабочей копии, так что они не всегда могут эффективно обработать их. Команда будет по прежнему работать, но старые серверы могут посылать все данные оставляя клиенту фильтровать ненужное, что может означать большой сетевой трафик. Если это возможно, то вам следует обновить сервер до версии как минимум 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 4.27, «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....

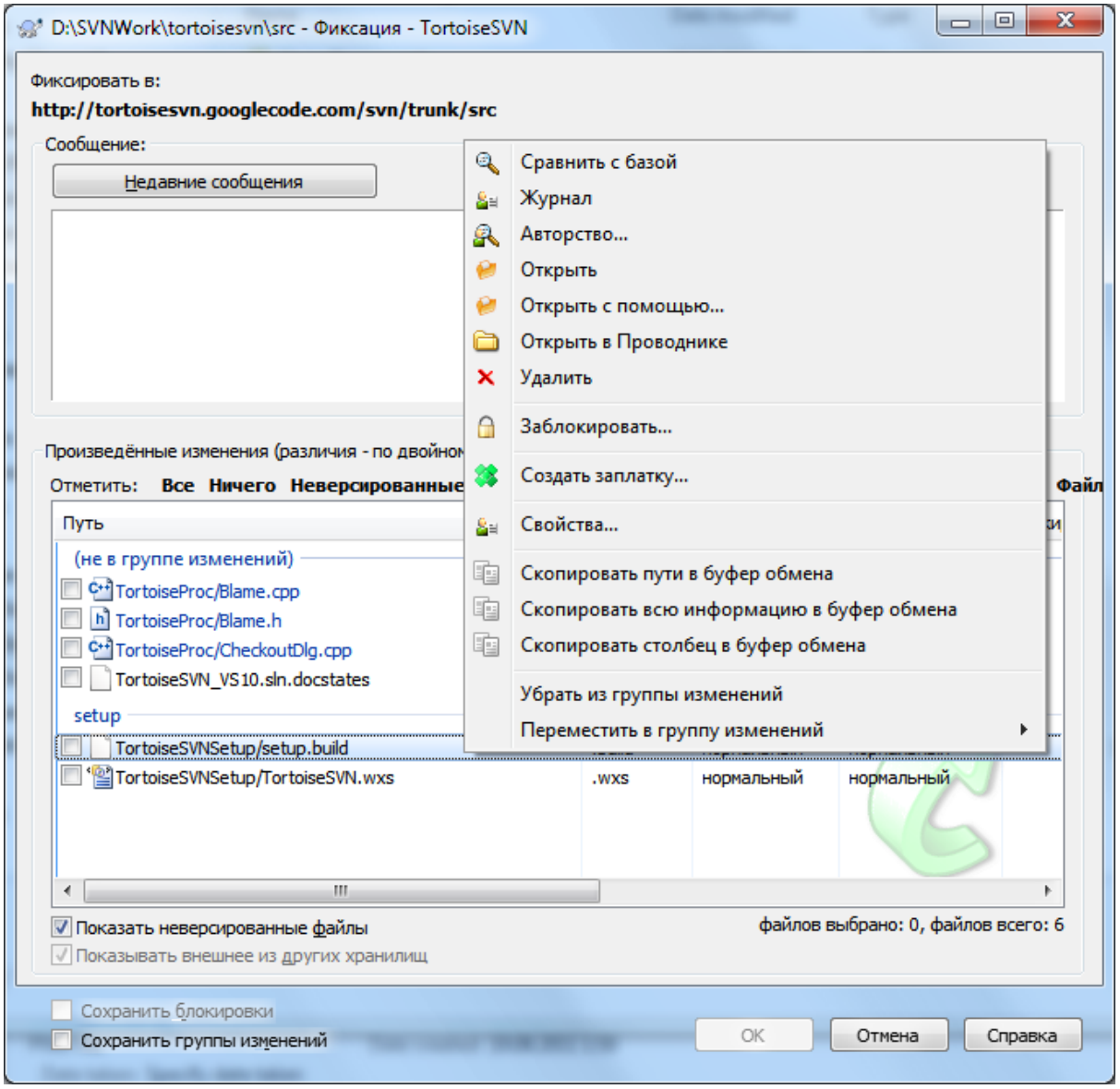


Рисунок 4.8. Диалог фиксации

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.

Чтобы быстро отметить или снять пометку с типов файлов, таких как, все версированные файлы или все изменённые файлы, кликните на ссылки над списком отображаемых элементов.

Для информации о расцветке и оверлейных значках соответствующих статусу обратитесь к Раздел 4.7.3, «Локальный и удалённый статус».

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.

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.
- добавить файл в список `svn:global-ignores`, используя TortoiseSVN → Добавить в список игнорирования (рекурсивно). Это затронет папку, для которой вы устанавливаете свойство `svn:global-ignores`, и также все подпапки.

Для дополнительной информации прочтите Раздел 4.14, «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 4.31.1, «Общие настройки», чтобы узнать, как редактировать конфигурационный файл Subversion.

Предупреждение при фиксации в ветку tags

Обычно фиксации делаются в ветки trunk или branch, но не в tags. Всё же, ветка tag считается неизменной и не должна изменяться.

При попытке фиксации в URL ветки tag TortoiseSVN показывает диалог с подтверждением, чтобы удостовериться действительно ли это то, что надо. Потому что в большинстве случаев такие фиксации делаются случайно.

Однако, эта проверка работает только если схема хранилища является одной из рекомендованных, т.е. использует названия `trunk`, `branches` и `tags` для обозначения трех главных областей. Если структура отличается, то определение того, что является tag/branch/trunk (известно также как `шаблоны классификации`) может быть настроено в диалоге настроек: Раздел 4.31.2, «Настройки графа ревизий»

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.

Перетягивание файлов из списка внизу диалога фиксации в поле ввода сообщения журнала вставит пути в виде простого текста в это поле. Это полезно если вы хотите написать сообщение журнала, которое включает пути, которые эта фиксация затрагивает.

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.

Исправление внешних копирований

Если вы скопировали файл, но забыли, что это надо было сделать при помощи команды Subversion, вы можете исправить такое копирование, чтобы новый файл не потерял свою историю. Просто выберите оба файла: и со старым именем (нормальный или изменённый), и с новым именем (неверсированный) и используйте Контекстное меню → Поправить копирование, чтобы обозначить эту пару файлов как копирование.

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. Фиксировать только части файлов

Иногда вы хотите фиксировать только часть изменений сделанных вами в файле. Обычно, такая ситуация возникает, когда вы работаете над чем-то и вдруг требуется внести срочное исправление и зафиксировать, и это исправление в том же файле, с которым вы работаете.

Щёлкните правой кнопкой мыши на файле и используйте Context Menu → Восстановить после фиксации. При этом будет создана копия файла. Затем вы можете редактировать файл, например, в текстовом редакторе и отменить все изменения, которые вы не хотите фиксировать. После сохранения всех таких изменений вы можете фиксировать файл.

Использование TortoiseMerge

Если вы используете TortoiseMerge для редактирования файла вы можете либо редактировать изменения как обычно, либо отметить все изменения, которые вы хотите добавить. Щёлкните правой кнопкой мыши на изменённом блоке и воспользуйтесь командой Контекстное меню → Пометить это изменение чтобы добавить это изменение. Наконец, щёлкните правой кнопкой мыши и воспользуйтесь командой Контекстное меню → Оставить только помеченные изменения, которая изменит правый вид, добавив только ранее помеченные изменения и убрав неотмеченные.

После того, как фиксация выполнена, копия файла восстанавливается автоматически и у вас есть файл со всеми вашими изменениями, которые не были зафиксированы.

4.4.4. 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.

4.4.5. 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.

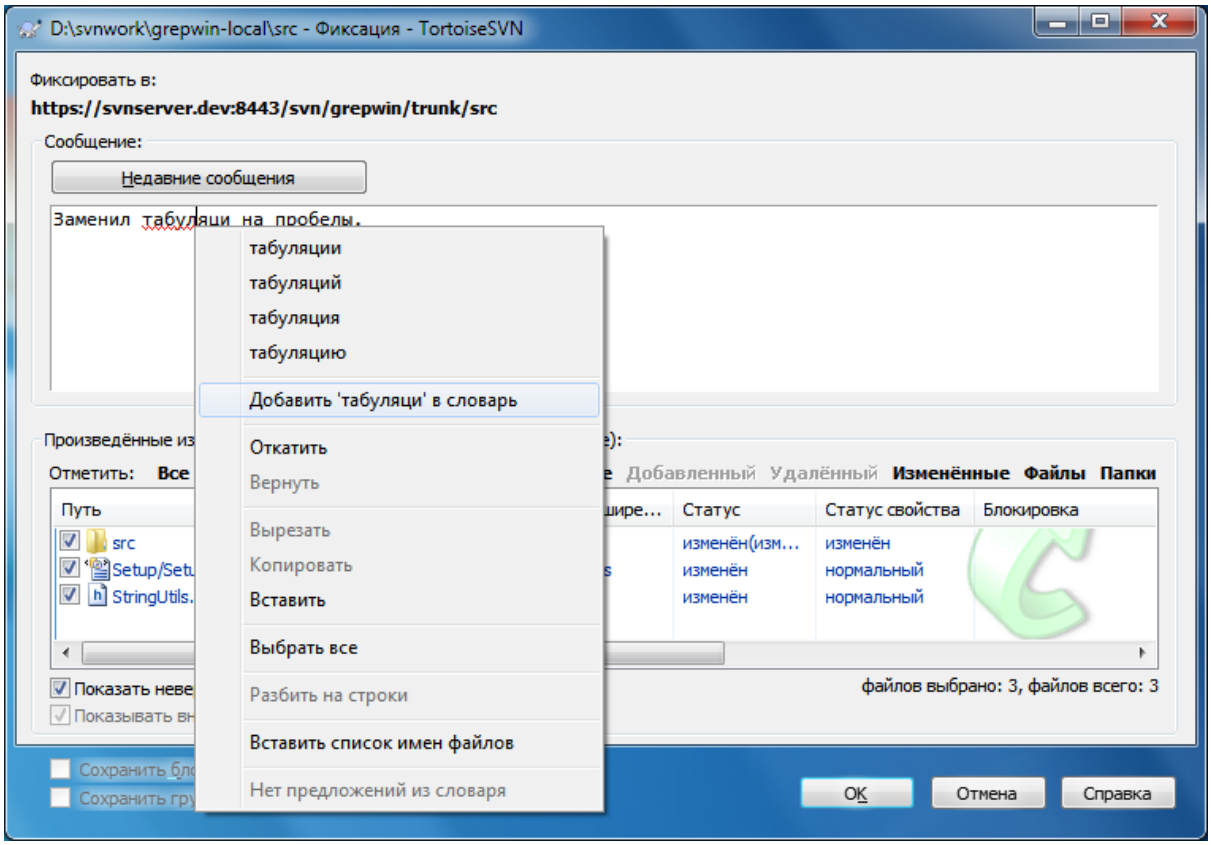


Рисунок 4.9. Проверка правописания в диалоге фиксации

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.

The log message window also includes a filename and function auto-completion facility. This uses regular expressions to extract class and function names from the (text) files you are committing, as well as the filenames themselves. If a word you are typing matches anything in the list (after you have typed at least 3 characters, or pressed **Ctrl+Space**), a drop-down appears allowing you to select the full name. The regular expressions supplied with TortoiseSVN are held in the TortoiseSVN installation `bin` folder. You can also define your own regexes and store them in `%APPDATA%\TortoiseSVN\autolist.txt`. Of course your private autolist will not be overwritten when you update your installation of TortoiseSVN. If you are unfamiliar with regular expressions, take a look at the introduction at https://en.wikipedia.org/wiki/Regular_expression, and the online documentation and tutorial at http://www.regular-expressions.info/.

Получить правильное регулярное выражение бывает сложно. Поэтому для поиска подходящего выражения есть тестовый диалог, в котором вы вводите выражение и набираете имена файлов для проверки. Запускайте его из командной строки с помощью команды `TortoiseProc.exe /command:autotexttest`.

Окно сообщения журнала также поддерживает фрагменты для сообщения фиксации. Эти фрагменты показываются в выпадающем списке автоподстановки когда вы нажмете горячую клавишу, и после выбора фрагмента в выпадающем списке вставляется полный текст фрагмента. Фрагменты поддерживаемые TortoiseSVN сохраняются в папке `bin` установки TortoiseSVN. Вы также можете определить собственные фрагменты и хранить их в файле `%APPDATA%\TortoiseSVN\snippet.txt`. Символ `#` – это комментарий. Разделители строк вставляются следующим образом: `\n` и `\r`. Чтобы вставить обратный слэш: `\\`.

Вы можете повторно использовать введённые ранее сообщения журнала: достаточно щёлкнуть на кнопку Недавние сообщения и появится список из нескольких последних введённых вами 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 4.18, «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 4.29, «Интеграция с системами отслеживания ошибок/проблем».

4.4.6. Ход выполнения фиксации

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

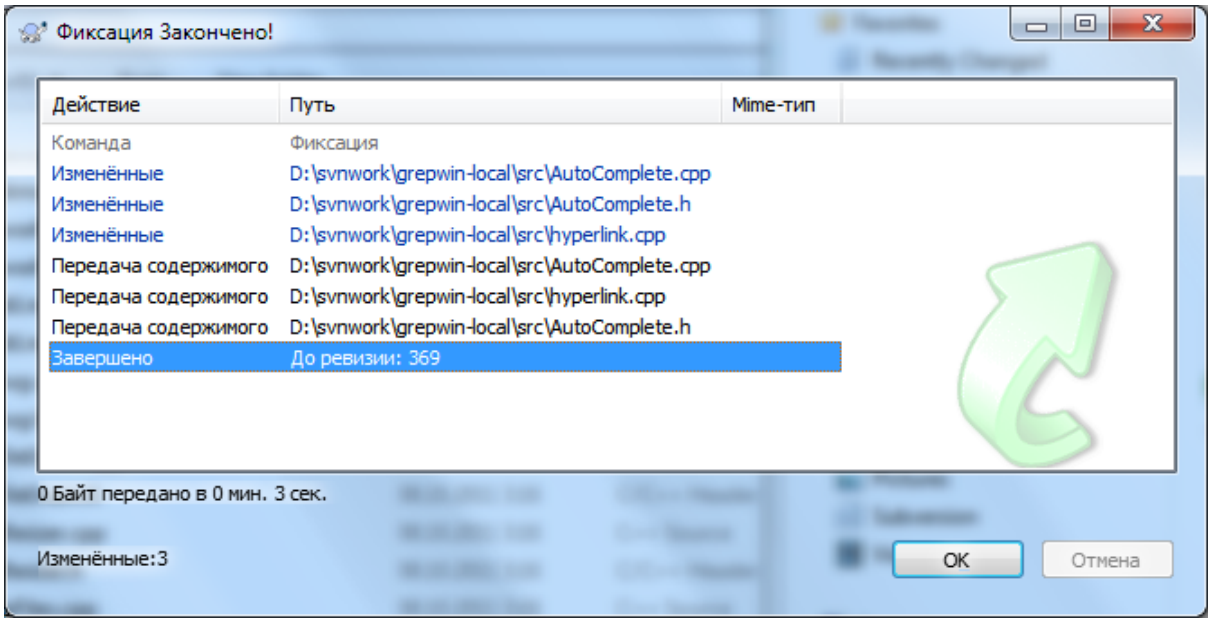


Рисунок 4.10. Диалог выполнения, отображающий ход выполнения фиксации

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 4.31.1.5, «Настройки цветов в TortoiseSVN».

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

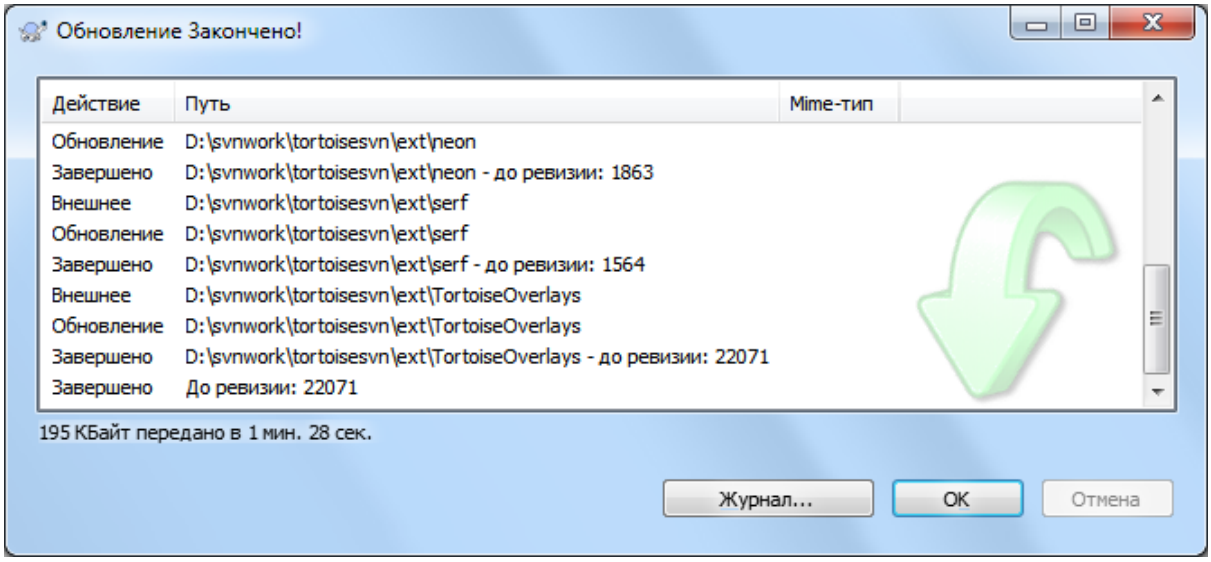


Рисунок 4.11. Окно выполнения, отображающее законченное обновление

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 4.31.1.5, «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 ревизии HEAD хранилища, что является наиболее часто используемым вариантом. Если вы хотите больше управлять процессом обновления, то должны использовать вместо этого TortoiseSVN → Обновить до ревизии.... Это позволит вам обновить вашу рабочую копию до определённой ревизии, а не только до последней. Представим, что ваша рабочая копия имеет ревизию 100, но вы хотите воспроизвести состояние, которое она имела в ревизии 50. Тогда просто обновите до ревизии 50.

В том же диалоге вы можете выбрать *глубину*, до которой обновить текущую папку. Используемые термины описаны в xref linkend="tsvn-dug-checkout-depth"/>. Глубина по умолчанию — это Рабочая копия, которая хранит настройки существующей глубины. Вы также можете установить глубину `постоянной`, это означает что последующие обновления будут использовать эту новую глубину, т. е. эта глубина используется по умолчанию.

Чтобы упростить включение и исключение определенных элементов из фиксации нажмите на кнопку Выберите элементы.... Это откроет новый диалог, в котором вы сможете отметить все элементы нужные вам в рабочей копии и снять отметку с ненужных.

Вы также можете выбрать игнорировать ли любые внешние проекты при обновлении (т. е. проекты использованные с помощью `svn:externals`).

Внимание

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.

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 в файле происходит, когда несколько разработчиков внесли изменения в одни и те же строки. Subversion не знает ничего о вашем проекте, поэтому разрешение конфликтов ложится на плечи разработчиков. Зона конфликта в текстовом файле отмечена следующим образом:

```
<<<<<<< имя файла

ваши изменения

=======

результат автоматического мержа с репозиторием

>>>>>>> ревизия
```

Также, для каждого файла, имеющего конфликты, Subversion создает в том же каталоге 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 текстовый редактор для ручного улаживания

конфликта. Вы должны решить, как должен выглядеть код, сделать необходимые изменения и сохранить файл. Использование инструмента слияния, например, TortoiseMerge или другого популярного инструмента, обычно является более простым вариантом, т.к. в большинстве случаев они отображают файлы в 3-х панелях и вам не нужно беспокоиться о маркерах конфликта. Если же вы используете текстовый редактор, то вы должны найти строки начинающиеся с символов <<<<<<<.

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. Конфликт свойств

Конфликт свойств возникает когда два или более разработчиков изменили одно и то же свойство. Как и содержимое файла, разрешение конфликта может быть выполнено только разработчиками.

Если одно из изменений должно перекрыть другое, то выберите вариант Уладить, применив локальное свойство или Resolve using remote property. Если изменения должны быть слиты, выберите Редактировать свойство вручную, решите каким должно быть значение свойства и отметьте как разрешенный.

4.6.3. 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 различных 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.

4.6.3.1. Локальное удаление, поступающее при обновлении редактирование

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

2. Одновременно разработчик 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.

The conflict edit dialog offers to merge changes if it can find the original file of the renamed `Bar.c`. If there are multiple files that are possible move sources, then a button for each of these files is shown which allow you to chose the correct file.

4.6.3.2. 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 добавленный (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.

To merge her local changes with the reshuffle, Developer B must first find out to what filename the conflicted file `Foo.c` was renamed/moved in the repository. This can be done by using the log dialog. Then use the button which shows the correct source file to resolve the conflict.

If Developer B decides that A's changes were wrong then she must choose the Mark as resolved button in the conflict editor dialog. This marks the conflicted file/folder as resolved, but Developer A's changes need to be removed by hand. Again the log dialog helps to track down what was moved.

4.6.3.3. 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`составить - 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.

4.6.3.4. Локально отсутствующее, поступающее при обновлении редактирование

1. Разработчик 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 отсутствующий файл из хранилища и после этого пометите конфликт как улаженный, то ваша копия будет снова удалена. Сначала вы должны уладить конфликт.

4.6.3.5. Локальное редактирование, поступающее при слиянии удаление

1. Разработчик А работающий в trunk перемещает `Foo.c` в `Bar.c` и фиксирует это в хранилище.

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

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.

To merge her local changes with the reshuffle, Developer B must first find out to what filename the conflicted file `Foo.c` was renamed/moved in the repository. This can be done by using the log dialog for the merge source. The conflict editor only shows the log for the working copy as it does not know which path was used in the merge, so you will have to find that yourself. The changes must then be merged by hand as there is currently no way to automate or even simplify this process. Once the changes have been ported across, the conflicted path is redundant and can be deleted.

If Developer B decides that A's changes were wrong then she must choose the Mark as resolved button in the conflict editor dialog. This marks the conflicted file/folder as resolved, but Developer A's changes need to be removed by hand. Again the log dialog for the merge source helps to track down what was moved.

4.6.3.6. Локальное удаление, поступающее при слиянии удаление

1. Разработчик A работающий в trunk перемещает `Foo.c` в `Bar.c` и фиксирует это в хранилище.

2. Разработчик B работающий в ответвлении перемещает `Foo.c` в `Bix.c` и фиксирует это в хранилище.

Слияние изменений разработчика A 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.

To resolve this conflict, Developer B has to find out to what filename the conflicted file `Foo.c` was renamed/moved in the repository. This can be done by using the log dialog for the merge source.

Затем разработчик Б должен решить, какое из новых имён файла `Foo.c`составить - то, которое дал разработчик A 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.

4.6.3.7. Другие конфликты деревьев

Существуют и другие случаи помечаемые как конфликты деревьев из-за того, что конфликт затрагивает скорее папку, а не файл. Например, если вы добавите папку с одинаковым именем в ствол и отвлетвление и затем попытаетесь выполнить слияние, то у вас возникнет конфликт деревьев. Если вы хотите исключить созданную папку из слияния, то отметьте конфликт как улаженный. Если же вы хотите использовать папку из источника слияния, то вам необходимо удалить эту папку из назначения и выполнить слияние снова. Если вы столкнулись с чем-то более сложным, то улаживайте конфликт вручную.

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

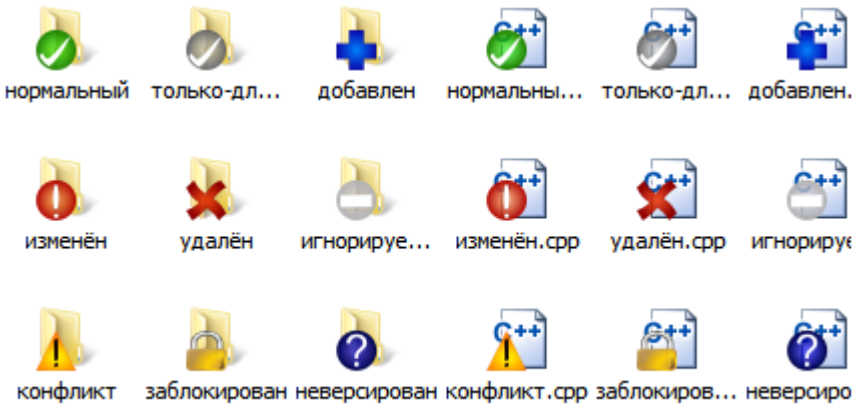


Рисунок 4.12. Проводник с пометками на значках

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.

Сейчас когда есть много Tortoise клиентов кругом (TortoiseCVS, TortoiseHg, ...) ограничение значков стало настоящей проблемой. Чтобы справиться с этим проект TortoiseSVN предоставил общеиспользуемый набор значков, загружаемый как DLL, который может быть использован всеми клиентами Tortoise. Проверьте поставщика вашего клиента, использует ли он это :-)

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

4.7.2. Подробный статус

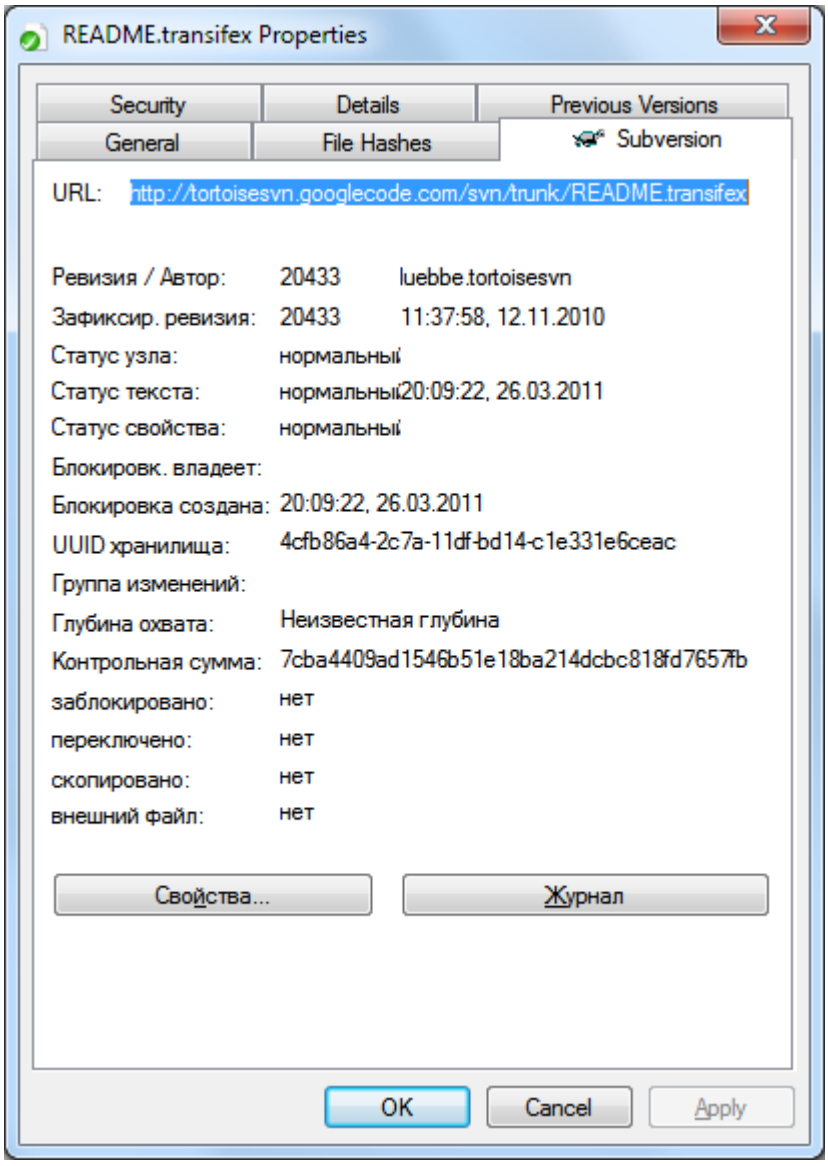


Рисунок 4.13. Страница свойств Проводника, вкладка Subversion

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.

4.7.3. Локальный и удалённый статус

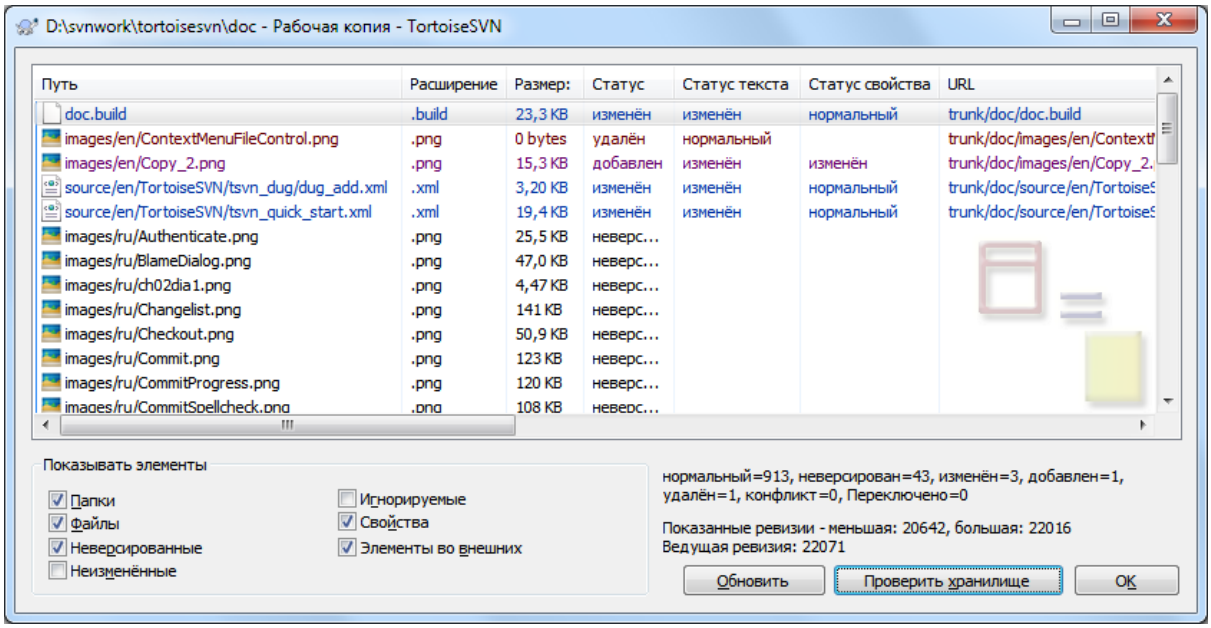


Рисунок 4.14. Проверка на наличие изменений

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.

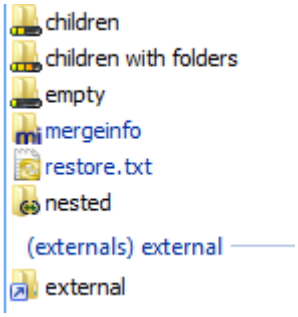
Зеленый
: Элементы, изменённые локально и в хранилище. Изменения будут объединены при обновлении. Это *может* привести к конфликту при обновлении.

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 4.31.1.5, «Настройки цветов в TortoiseSVN».

Оверелейные значки также используются для обозначения других состояний. Нижеприведённый скриншот демонстрирует все возможные оверлеи, которые отображаются при необходимости.



Оверелеи показаны для следующих состояний:

- Глубина извлечения `empty` обозначает только сам элемент.
- Глубина извлечения `files` обозначает только сам элемент и все дочерние файлы без дочерних папок.
- Глубина извлечения `immediates` обозначает только сам элемент и все дочерние файлы и папки, но без элементов дочерних папок.
- Вложенные элементы, т.е. рабочие копии внутри рабочей копии.
- Внешние элементы, т.е. все элементы добавленные с использованием свойства `svn:externals`.
- Элементы, которые были восстановлены после фиксации. Более подробно см. Раздел 4.4.3, «Фиксировать только части файлов».
- Элементы, у которых изменены свойства, но только свойство `svn:mergeinfo`. Если изменено любое другое свойство, то оверлей не используется.

Элементы, которые были переключены на другой путь хранилища, отмечены маркером `(s)`. Может быть вы переключили что-то пока работали в ответвлении и забыли переключить обрано на trunk. Это предупреждающий знак для вас! Контекстное меню позволяется вам переключить это обратно на нормальной путь.

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 хотите проверить файл подробно, то вы можете перетянуть его отсюда в другое приложение, такое как текстовый редактор или IDE. Или можете сохранить копию просто перетянув его в папку в проводнике.

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.

Исправление внешних копирований

Если вы скопировали файл, но забыли, что это надо было сделать при помощи команды Subversion, вы можете исправить такое копирование, чтобы новый файл не потерял свою историю. Просто выберите оба файла: и со старым именем (нормальный или изменённый), и с новым именем (неверсированный) и используйте Контекстное меню → Поправить копирование, чтобы обозначить эту пару файлов как копирование.

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.

Вы можете встретить группы изменений в нескольких местах, но наиболее важные из них - это диалог фиксации и диалог проверки наличия 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 OK. 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 наличия изменений, но для этого придётся включить отображение всех неизменённых файлов.

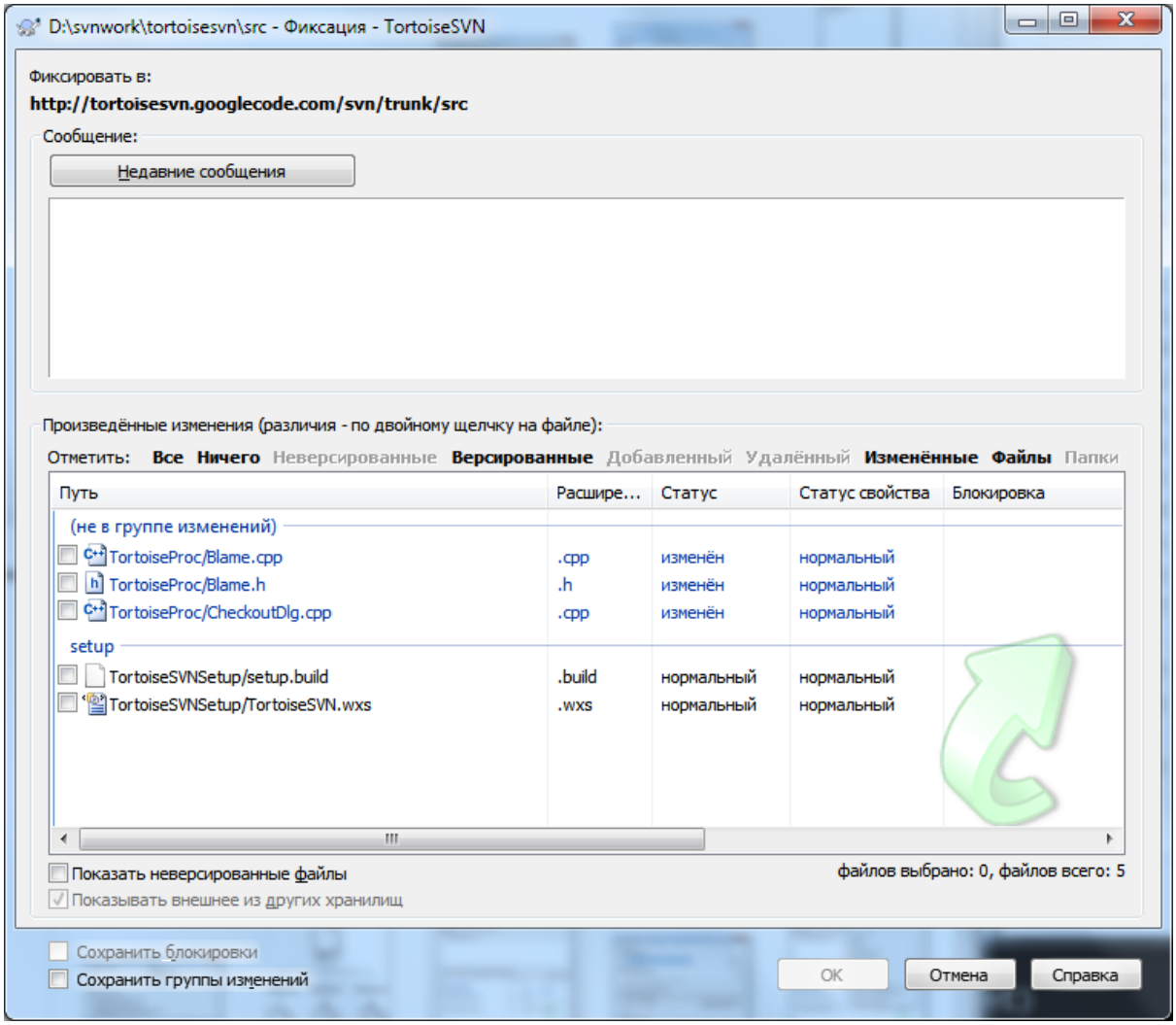


Рисунок 4.15. Диалог фиксации с группами изменений.

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

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 4.4.4, «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.

Предупреждение

Обратите внимание, что если вы используете списки изменений, то внешние включения больше не отображаются в своей отдельной группе. Раз уж есть списки изменений, то файлы и папки группируются по спискам, а не по внешним включениям.

4.9. Shelving

More often than wanted, it's necessary to stop what you were working on and work on something else. For example a serious problem needs immediate dealing with and you have to stop working on the new feature. If possible, you should commit the changes you have done so far and then start working on the urgent issue, but often those changes would break the build or are just not ready for committing yet.

So if you can't commit your local changes yet, you have to put them aside while you're working on the urgent issue. The *shelving* feature helps you do exactly that: you can store your local changes on a shelve, get your working copy in a clean state again and work on the issue. After you're finished with the urgent issue and you've committed those changes, you can *unshelve* your shelved work and continue working on your previous task again.

Two new commands are implemented for this. One for shelving and one for unshelving.

To shelve your local changes, select your working copy and use Context Menu → Shelve The following dialog allows you to select the files you want to shelve and give a name under which you want to store them.

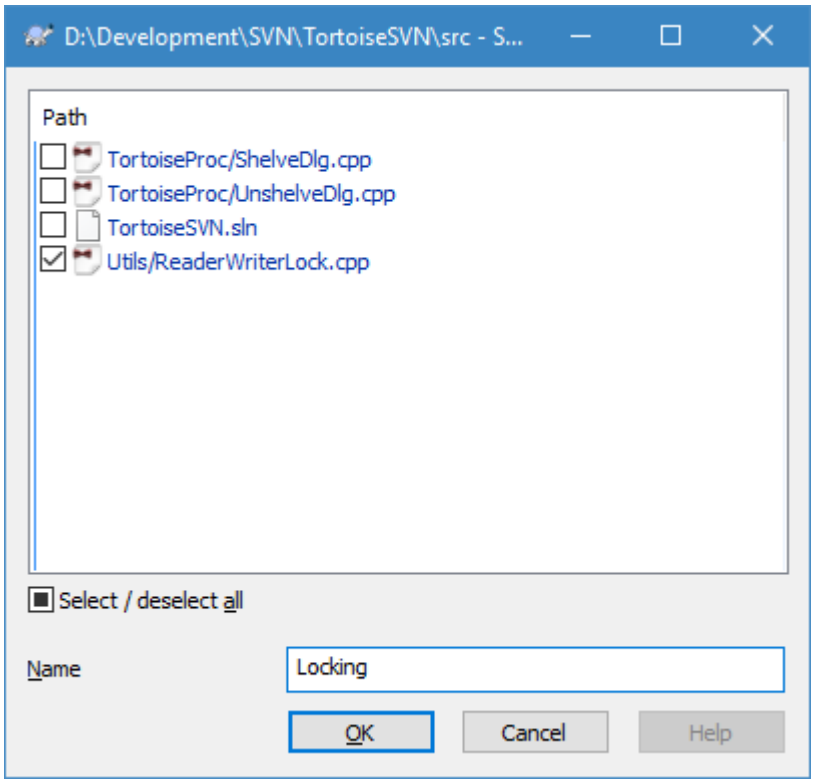


Рисунок 4.16. Shelve dialog

Then click OK and the changes are stored under the name you specified, and your working copy is reset to a clean state.

To unshelve your changes, use Context Menu → Unshelve to get the unshelve dialog. This dialog shows you a list of all shelved items. Select the shelved item you want to apply back to your working copy and click OK.

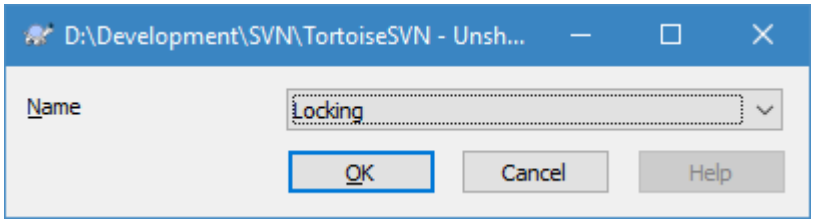


Рисунок 4.17. Unshelve dialog

Подсказка

Shelves are purely a local client feature. Creating and removing Shelves will not affect the repository, nor anyone else's working copy.

Предупреждение

Currently only text files can be shelved and unshelved. Binary files and text files encoded in utf-16 (unicode) can not be shelved.

4.10. 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.

4.10.1. Вызов диалога журнала ревизий

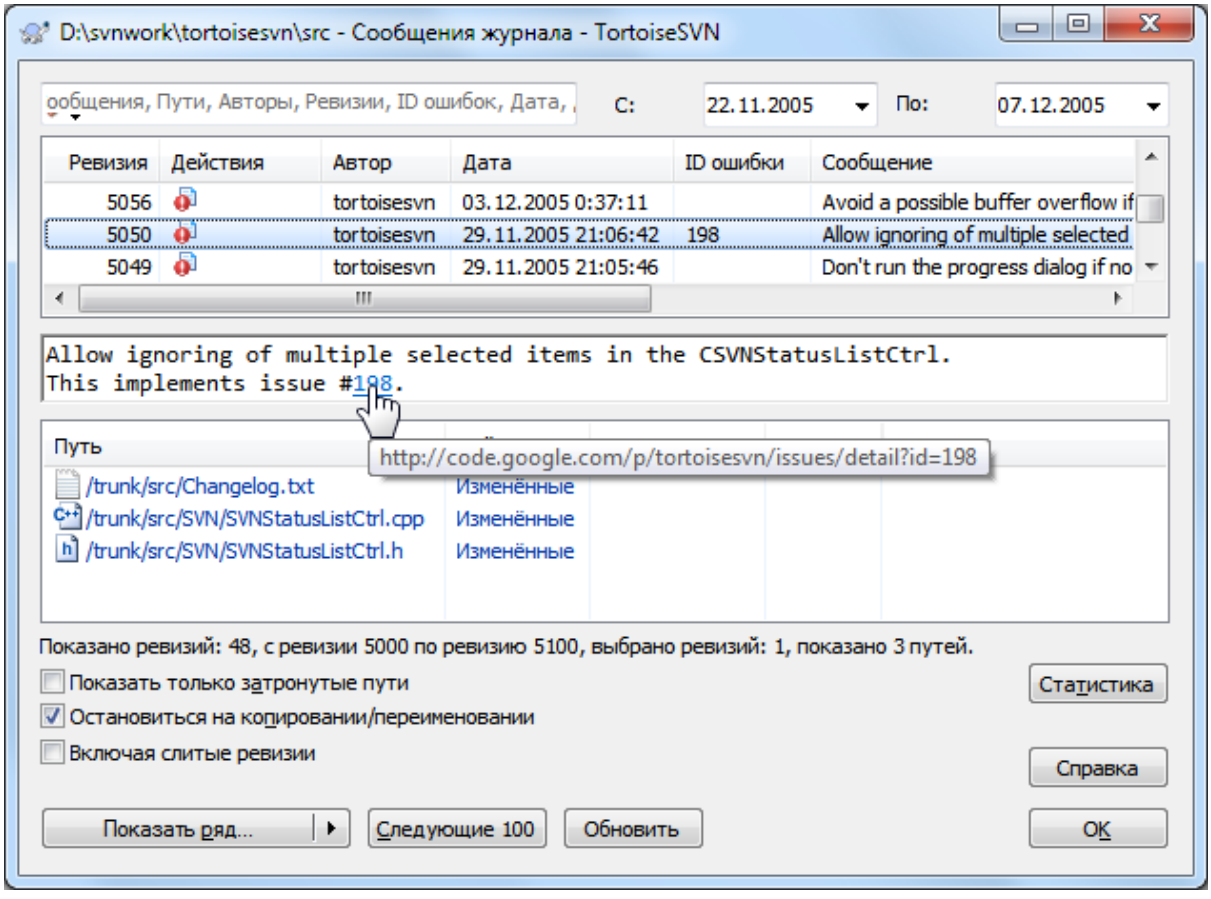


Рисунок 4.18. Диалоговое окно журнала ревизий

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
- Из обозревателя хранилища

Если хранилище недоступно, вы увидите диалог Перейти в автономный режим?, который описывает Раздел 4.10.10, «Автономный режим».

4.10.2. 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*.

Если ревизия переместила или переименовала файл или директорию, то в четвертой колонке отображается значок *перемещен*.

Если ревизия заменила файл или директорию при перемещении/переименовании, то в четвертой колонке отображается значок *перемещено с заменой*.

Если ревизия объединила файл или директорию, то в четвертой колонке отображается иконка *слито*.

Если в ревизии есть обратное слияние файла или директории, то в четвертой колонке отображается значок *обратное слияние*.

4.10.3. Получение дополнительной информации

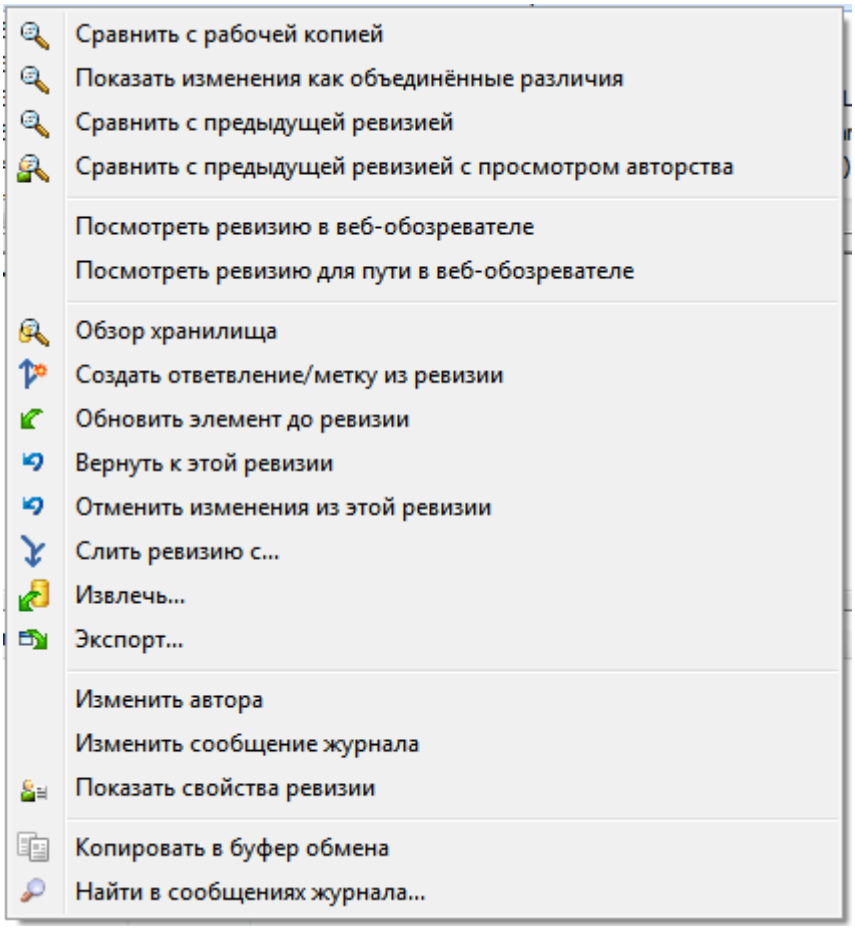


Рисунок 4.19. Контекстное меню верхней панели диалогового окна журнала ревизий

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 файла рабочей ревизии BASE, и и сравнивает полученные результаты с применением визуального средства просмотра различий. Прочтите Раздел 4.24.2, «Авторство различий» для дополнительной информации. (только файлы)

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.

  Если вы удерживаете клавишу **Shift** при выборе пункта меню, то сначала появится диалог, в котором вы можете установить параметры для объединенных различий. Эти параметры включают возможность игнорировать изменения в окончаниях строк и пробельных символах.

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 измененных файлов позволяющий вам выбрать файлы. Получает авторство для выбранной и предыдущей ревизий и сравнивает полученные результаты с применением визуального средства просмотра различий. (только папки)

Сохранить ревизию в...
: Сохраняет выбранную ревизию в файл и у вас есть старая ревизия этого файла. (только файлы)

Открыть / Открыть с помощью...
: Открывает выбранный файл либо с помощью приложения по умолчанию для этого типа файла, либо с помощью выбранного вами приложения. (только файлы)

Авторство...
: Получает авторство файла для выбранной ревизии. (только файлы)

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 4.10.7, «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 4.10.7, «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.

Создать рецензию code collaborator
: Это меню отображается, если установлен инструментарий SmartBear code collaborator. При первом запуске появляется диалог, в котором пользователю предлагается ввести свои учетные данные для code collaborator и SVN. Когда настройки сохранены, диалог больше не появляется при вызове меню, за исключением ситуации, когда пользователь удерживает клавишу **Ctrl** при выполнении пункта меню. Конфигурация и выбранные ревизии используются для запуска графического интерфейса пользователя клиента code collaborator, который создает новую рецензию для выбранных ревизий.

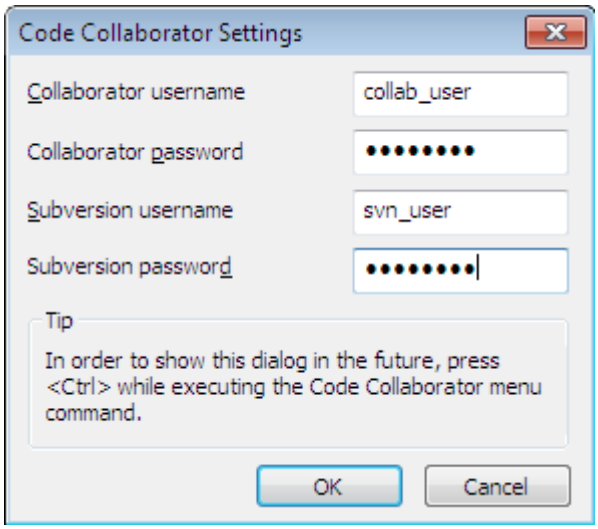


Рисунок 4.20. Диалог настроек Code Collaborator

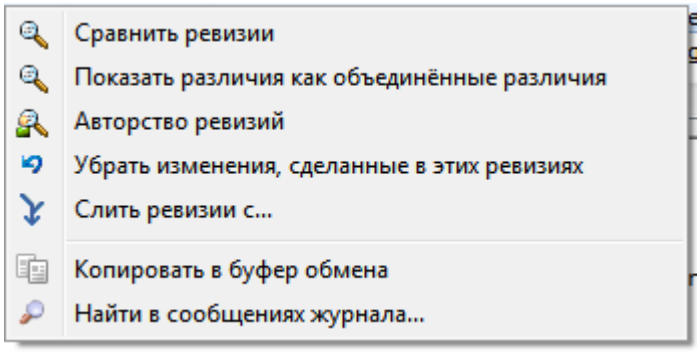


Рисунок 4.21. Контекстное меню верхней панели для двух выбранных ревизий

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 4.11.3, «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 4.24.2, «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.

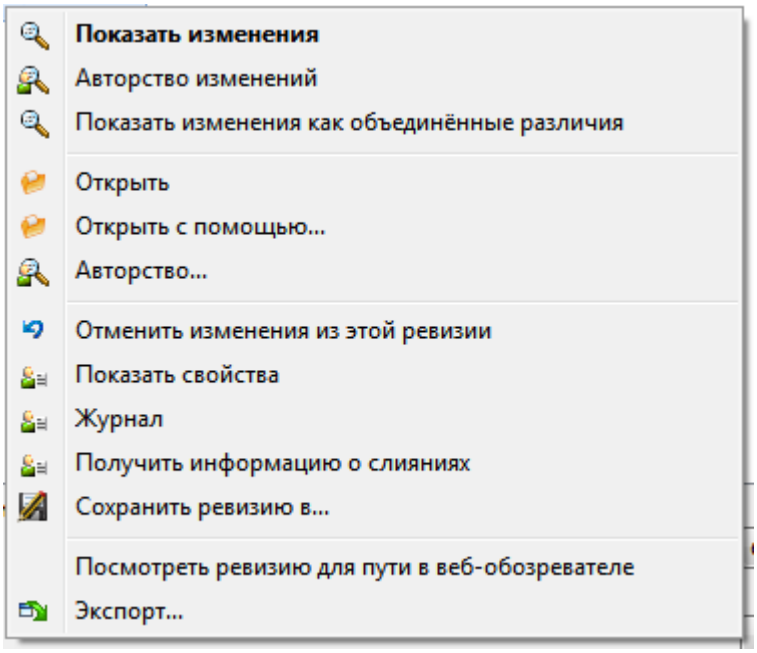


Рисунок 4.22. Контекстное меню нижней панели окна журнала

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

Показать изменения
: Показать изменения в выбранной ревизии для выбранного файла.

Авторство изменений
: Получает авторство для выбранной и предыдущей ревизий выделенного файла и сравнивает полученные результаты с применением визуального средства просмотра различий. Прочтите Раздел 4.24.2, «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 4.10.6, «Возможности по отслеживанию слияний».

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

Экспорт...
: Экспортирует выбранные элементы в этой ревизии в папку сохраняя иерархию файлов.

Когда выбрано несколько файлов в нижней панели окна журнала контекстное меню изменяется на следующее:

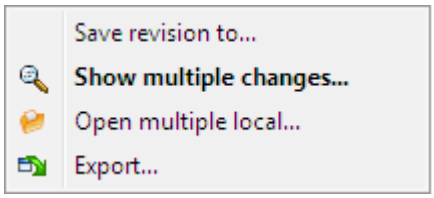


Рисунок 4.23. Нижняя панель диалога журнала с контекстным меню при нескольких выбранных файлах.

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

Показать множество изменений...
: Показать изменения, сделанные в выбранных файлах выбранных ревизий. Обратите внимание, что функциональность показа изменений вызывается несколько раз, что может запустить несколько копий вашей утилиты показа различий, или только добавить новую вкладку сравнения в утилите. Если вы выбрали более 15 файлов, то вам будет предложено подтвердить действие.

Открыть множество локальных файлов...
: Откроются файлы локальной рабочей копии, которые соответствуют выбранным файлам с помощью приложения, которое зарегистрированно для расширения файла. [Такое же поведение вы получили бы кликнув дважды на файле(ах) рабочей копии в проводнике Windows]. Это может быть медленной операцией в зависимости от того, как расширение файла ассоциировано с приложением, и возможностей приложения. В худшем случае для каждого выбранного файла могут быть запущены новые экземпляры приложения.

  Если вы удерживаете клавишу **Ctrl** при вызове команды, файлы рабочей копии всегда загружаются в Visual Studio. Это работает только при следующих условиях: Visual Studio должно быть запущено в том же контексте пользователя и иметь тот же уровень целостности процесса (process integrity level) [выполняемый как админ или нет] как и TortoiseProc.exe. Желательно иметь решение (solution) с загруженными изменёнными файлами, хотя это и не обязательно. Будут загружены только файлы существующие на диске с расширениями [.cpp, .h, .cs, .rc, .resx, .xaml, .js, .html, .htm, .asp, .aspx, .php, .css and .xml]. В Visual Studio может быть загружено максимум 100 файлов за раз, и файлы всегда загружаются в новые вкладки запущенного экземпляра Visual Studio. Преимущество просмотра изменений кода в Visual Studio заключается в том, что вы можете использовать встроенную навигацию по коду, поиск ссылок, статический анализ кода и другие инструменты встроенные в Visual Studio.

Экспорт...
: Экспортировать выбранные файлы/папки выбранной ревизии. Вызывает диалог для подтверждения адреса URL и ревизии, и выбора местоположения для экспорта.

Подсказка

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

Subversion использует номера ревизий подразумевая 2 разные вещи. Ревизия вообще представляет состояние хранилища в точке времени, но также она может использована для представления набора изменений, созданного этой ревизией. Например, «Done in r1234» означает, что изменения зафиксированные в ревизии r1234 реализуют функциональность X. Чтобы было понятно, какой смысл используется, мы используем два разных 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*.

4.10.4. 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 4.31.1.2, «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.

Чтобы обновить последнюю ревизию в случае были другие фиксации пока диалог журнала был открыт нажмите клавишу **F5**.

Чтобы обновить кэш журнала, нажмите **Ctrl-F5** ключи.

4.10.5. 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.

Когда вы запускаете просмотр журнала для папки, происходит выделение наибольшей ревизии, обнаруженной где-либо в этой папке, и это требует обхода рабочей копии. Обход производится в отдельном потоке, чтобы не задерживать отображение журнала, но в результате выделение для папок может появляться не сразу.

4.10.6. 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.

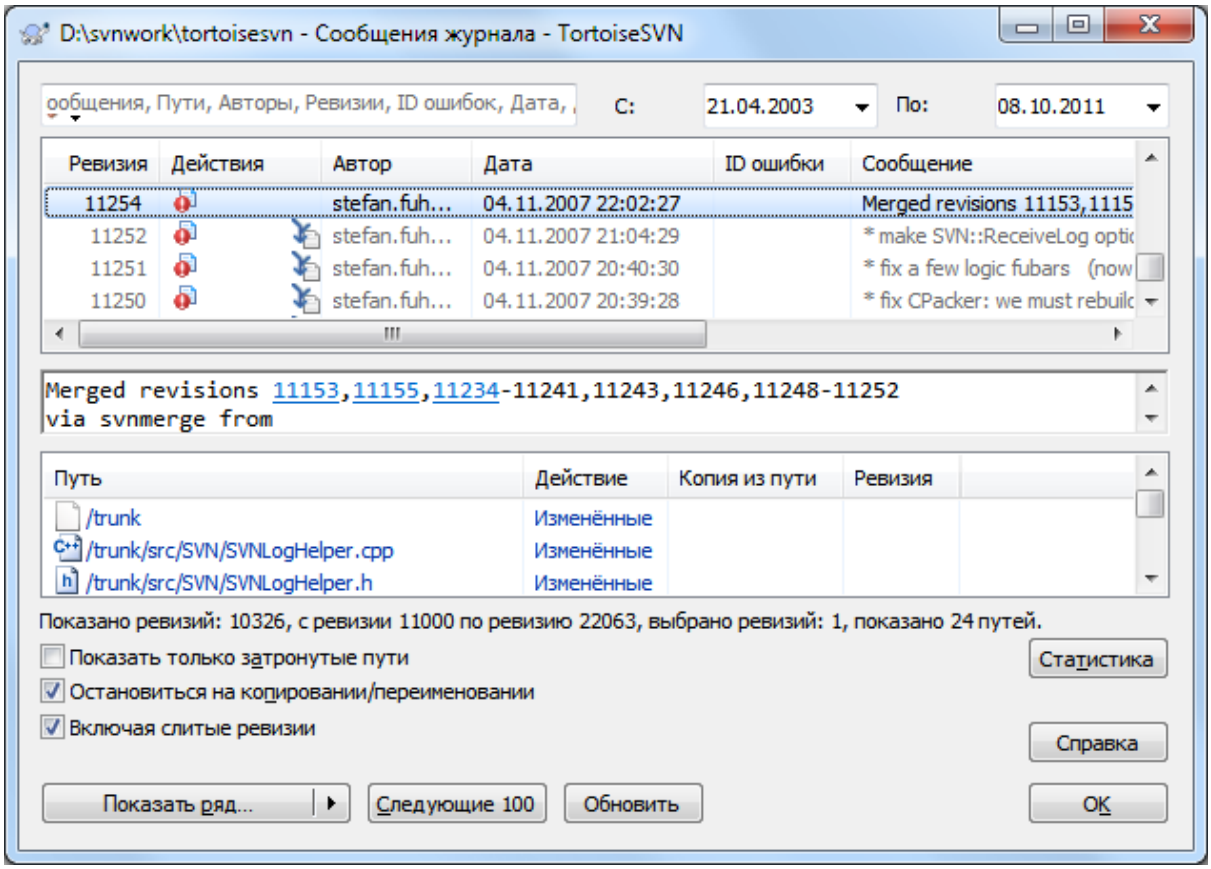


Рисунок 4.24. Диалог журнала, показывающий ревизии с отслеженными слияниями

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.

4.10.7. 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/1.8/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.

Предупреждение

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*.

Важно

С тех пор как TortoiseSVN хранит кэш всей информации журнала, правки автора и сообщений журнала будут показаны только локально у вас. Другие пользователи TortoiseSVN будут видеть кэшированных авторов и сообщения журнала пока не обновят кэш журнала. Смотрите Раздел 4.10.11, «Обновление вида»

4.10.8. 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.

Щелкните по иконке поиска для выбора направления поиска и выберите режим *Регулярное выражение*. Обычно, достаточно простой подстрокового поиска, но если необходимы более гибкие условия поиска, то используйте регулярные выражения. Если задержать курсор над строкой поиска, то появится

всплывающая подсказка как использовать регулярные выражения или подстроковые функции. Фильтры просто проверяют соответствует ли ваша строка фильтра записям в журнале и только затем с записями, которые *соответствуют* строке фильтра.

Простой поиск подстроки работает также как и поисковая система. Строки для поиска разделены пробелами и все строки должны совпадать. Вы можете использовать лидирующий − для указания того, что определенная подстрока не найдена (инверсное совпадение для этого условия), и вы можете использовать ! в начале выражения для инвертирования всего выражения. Вы можете использовать лидирующий + для указания того, что подстрока должна быть включена, даже если перед этим была исключена с помощью −. Обратите внимание, что порядок включения/исключения имеет важное значение. Вы можете использовать кавычки для обозначения строки содержащей пробелы, и если хотите найти именно кавычки, то используйте две кавычки вместе как самоэкранирующую последовательность. Обратите внимание, что символ обратного слэша *не* используется как экранирующий символ и имеет особого значения при поиске подстроки. Следующие примеры пояснят это:

```
Alice Bob -Eve
```

ищет строки содержащие Alice и Bob, но не Eve

```
Alice -Bob +Eve
```

ищет строки содержащие Alice, но не Bob, или строки содержащие Eve.

```
-Case +SpecialCase
```

ищет строки не содержащие Case, но включающие строки содержащие SpecialCase.

```
!Alice Bob
```

ищет строки не содержащие Alice и Bob

```
!-Alice -Bob
```

вы помните правила Де Моргана? NOT(NOT Alice AND NOT Bob) сокращается до (Alice OR Bob).

```
"Alice and Bob"
```

ищет дословное выражение «Alice and Bob»

```
""
```

ищет двойные кавычки везде в тексте

```
"Alice says ""hi"" to Bob"
```

ищет дословное выражение «Alice says "hi" to Bob».

Описание использования регулярных выражений в поиске выходит за рамки данного руководства, но вы можете найти онлайн документацию и обучающую программу по данному адресу http://www.regular-expressions.info/.

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

Вы можете также отфильтровать имена путей в нижней панели, используя флажок Скрыть несвязанные измененённые пути . Изменённые пути это те пути, которые содержат путь, показываемый в журнале. Если вы забрали журнал для какой-то папки, то это означает что только в той папке и ниже. Для файла означает только этот файл. Обычно список путей показывает любые другие пути, которые были затронуты в той же фиксации, но в сером цвете. Если же флажок отмечен, то эти пути не отображаются.

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 4.18.2, «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 возможности в обозревателе хранилища.

4.10.9. 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 ;-)

4.10.9.1. 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.

4.10.9.2. Страница 'Фиксации по автору'

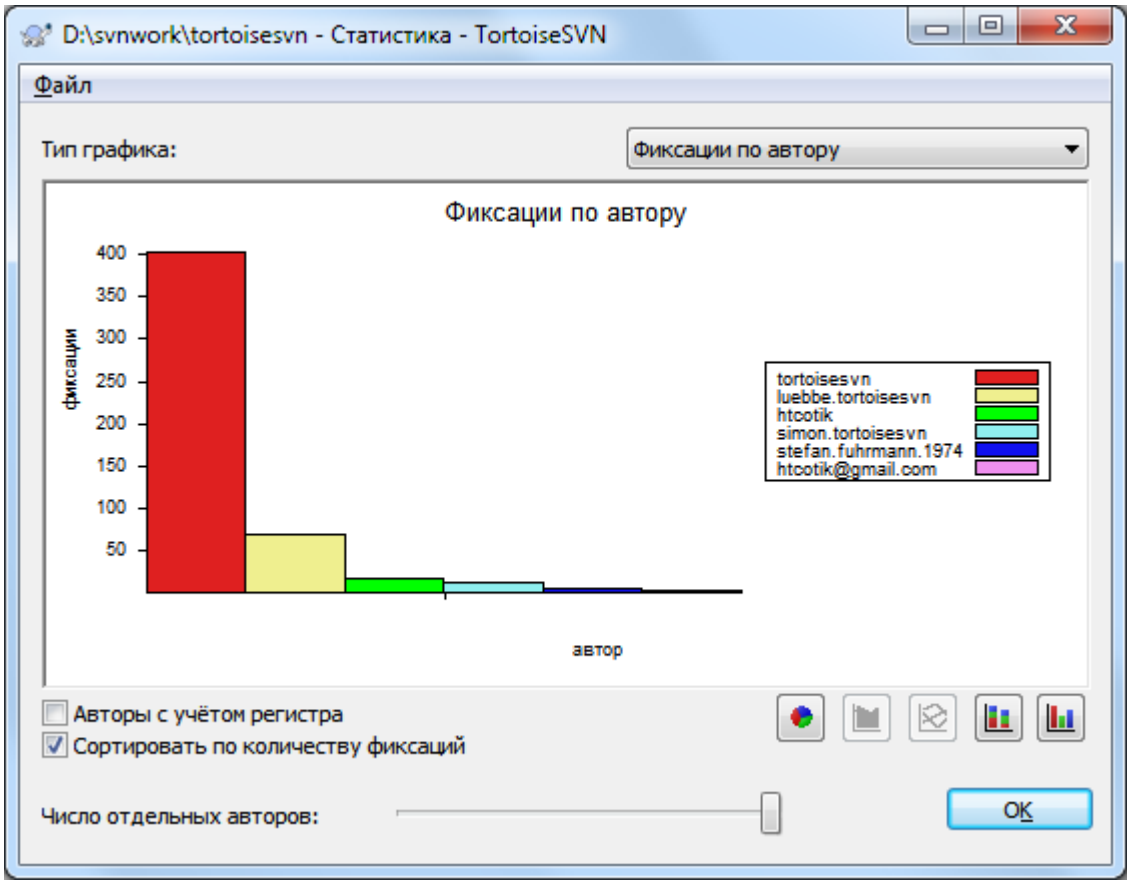


Рисунок 4.25. Гистограмма Фиксации-по-автору

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

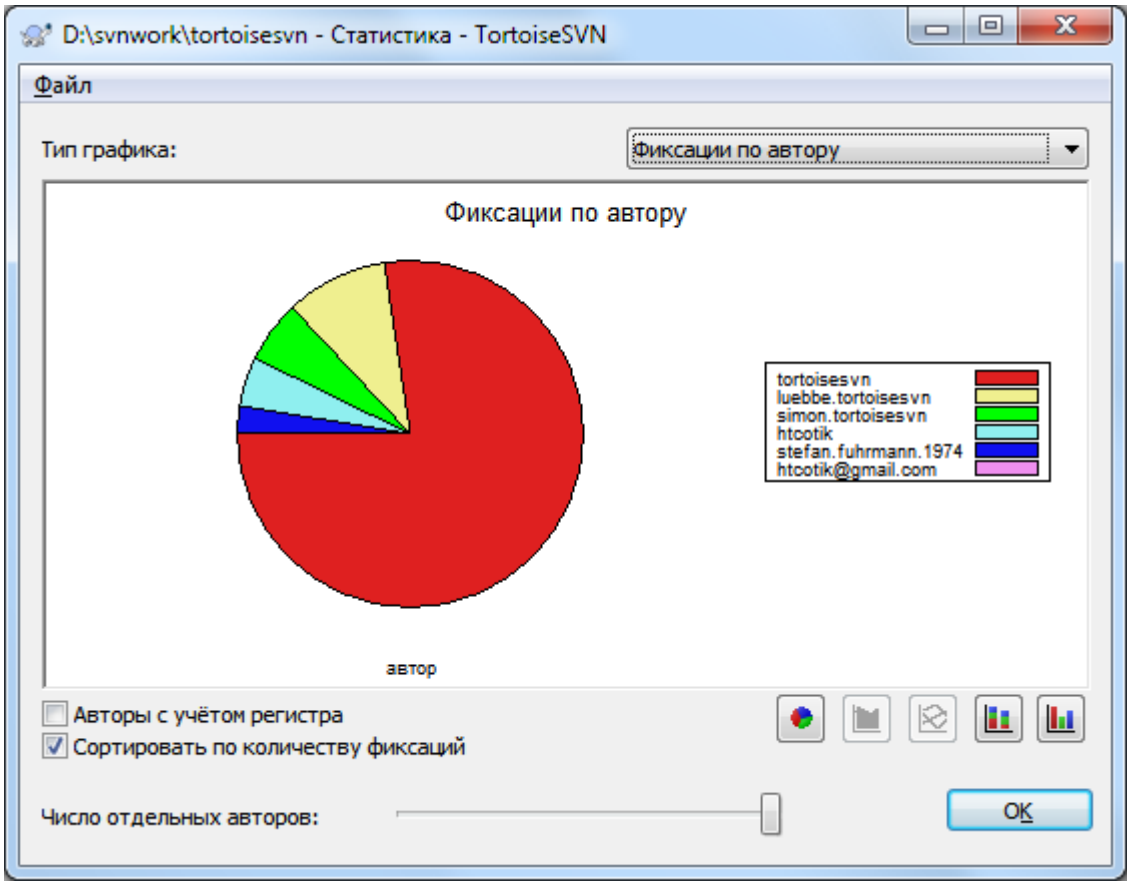


Рисунок 4.26. Секторная диаграмма Фиксации-по-автору

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*.

4.10.9.3. Страница 'Фиксации по датам'

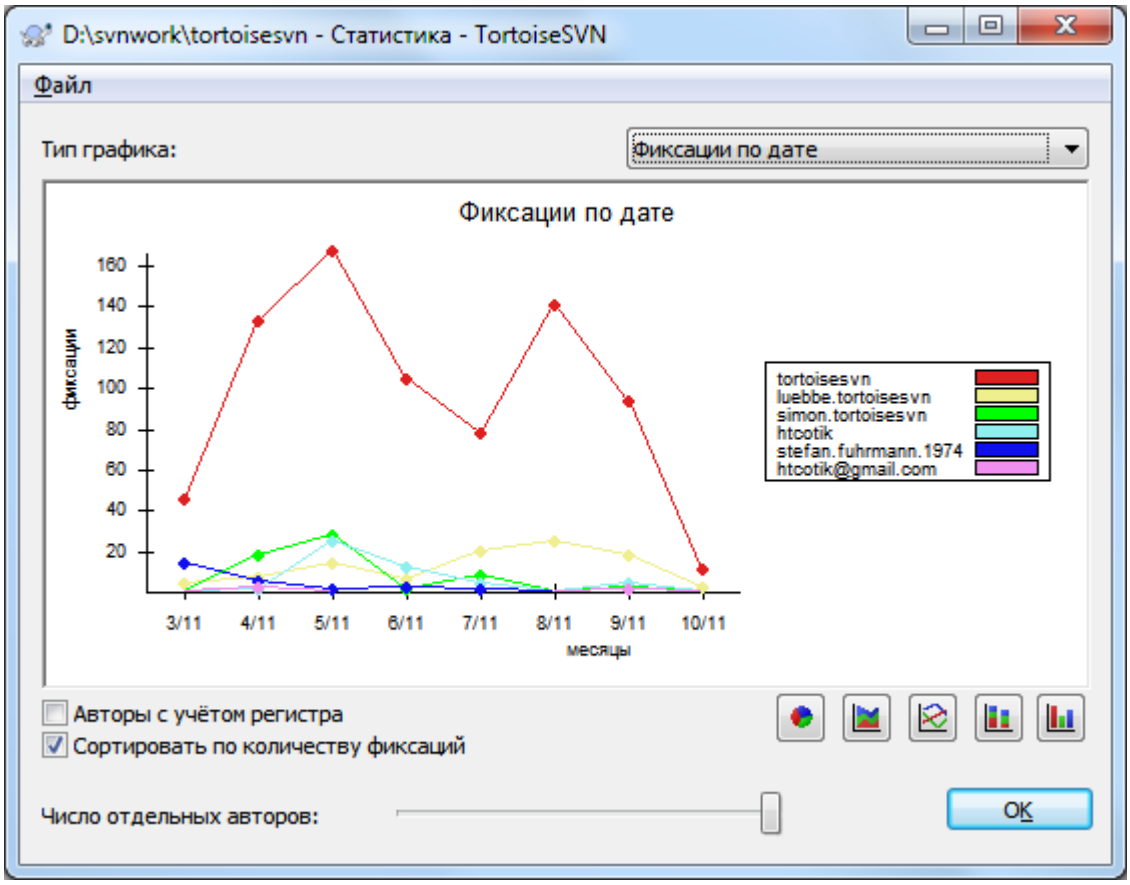


Рисунок 4.27. График Фиксации-по-датам

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.

4.10.10. Автономный режим

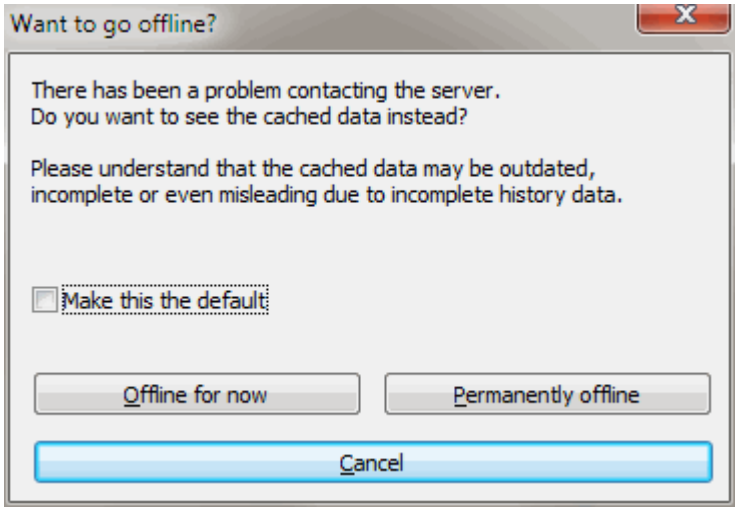


Рисунок 4.28. Диалог перехода в автономный режим

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 4.10.11, «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.

4.10.11. 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.

4.11. 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.

4.11.1. 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 4.11.3, «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 4.11.3, «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.

Если сравниваемые файлы размещены не в одной папке, то используйте команду TortoiseSVN → Сравнить позже, чтобы отметить первый файл для сравнения, затем найдите второй файл и используйте TortoiseSVN → Сравнить с "путь/к/отмеченному/файлу". Чтобы удалить отмеченный файл используйте команду TortoiseSVN → Сравнить позже ещё раз, но удерживайте клавишу **Ctrl** при нажатии.

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 4.24.2, «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 4.11.6, «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 4.31.5, «Настройки внешних программ», чтобы узнать, как настраивать другие инструменты сравнения.

4.11.2. 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 пробельных символов исключает изменения, которые состоят только в изменении количества или типа пробельных символов, т. е. изменение отступов или замена табуляций на пробелы. Добавление пробела там где его раньше не было или удаление пробела всё же отображается как изменение.

Игнорировать все непечатаемые знаки исключает все изменения только пробельных символов.

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

4.11.3. Сравнение папок

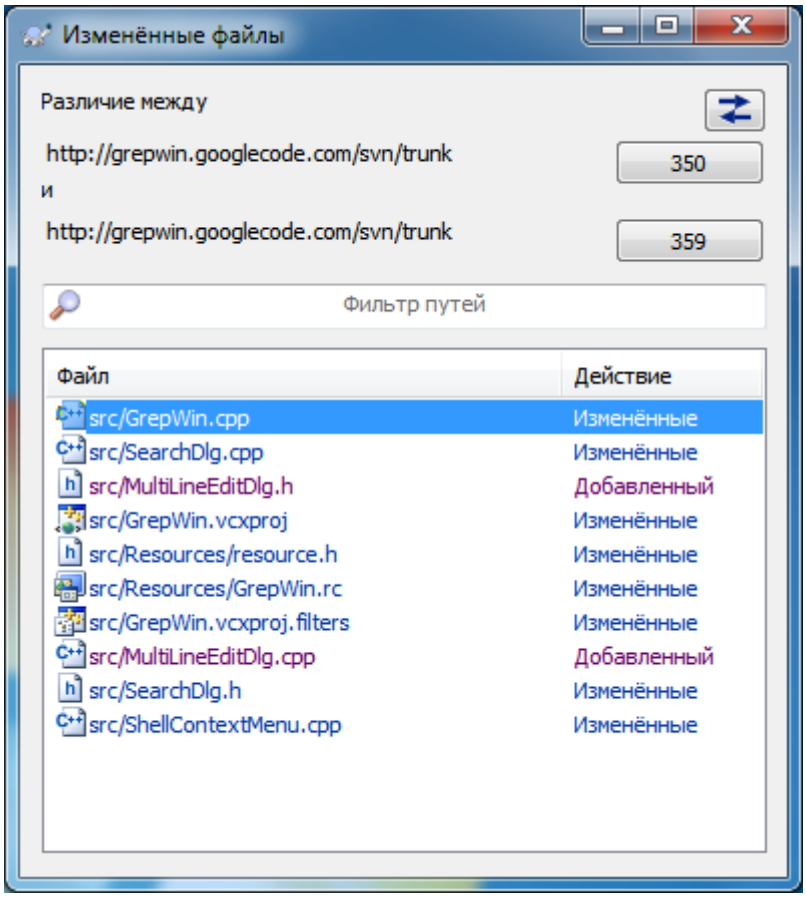


Рисунок 4.29. Диалог сравнения ревизий

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`.

4.11.4. 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.

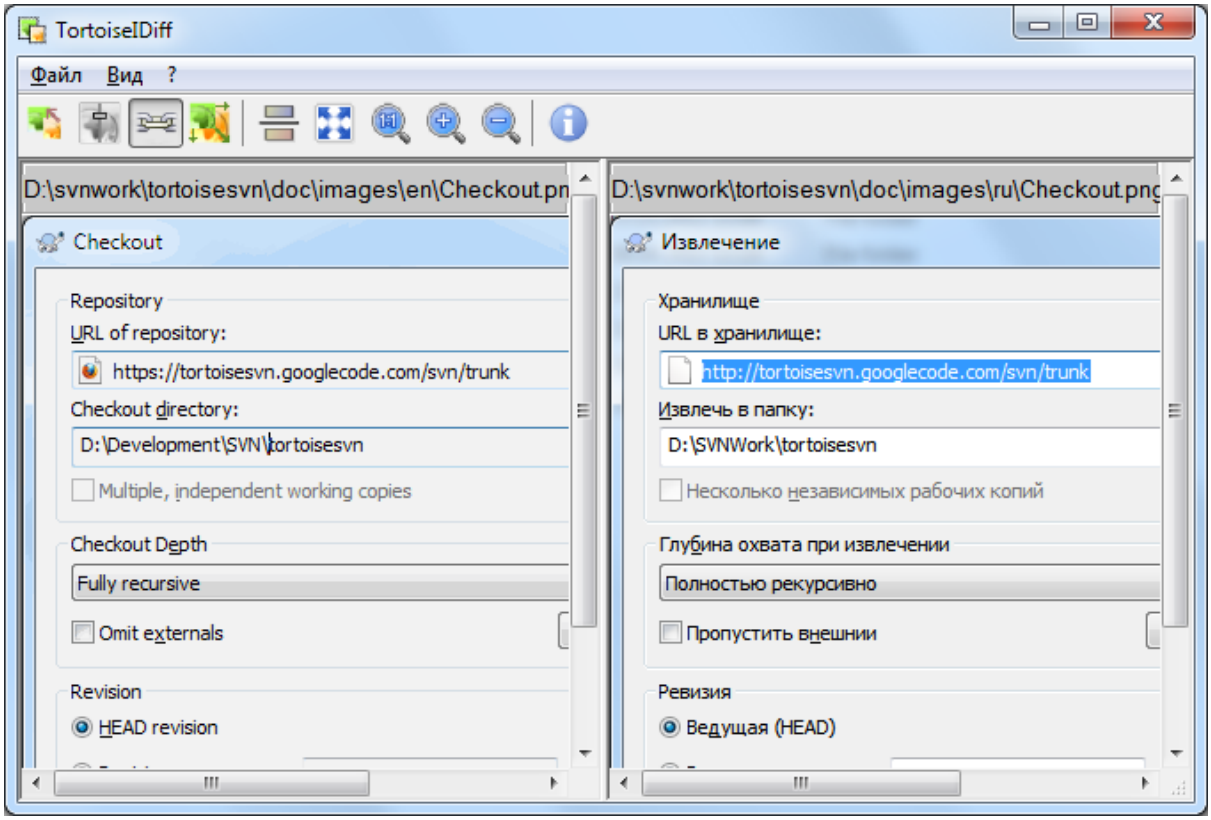


Рисунок 4.30. Программа просмотра различий в картинках

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.

4.11.5. Сравнение докуменов формата Office

Когда вы хотите сравнить нетекстовые документы обычно вам понадобится ПО, с помощью которого вы создали документ, т. к. оно понимает формат этого файла. В общеиспользуемых пакетах Microsoft Office и Open Office есть некоторая возможность просмотра различий, и TortoiseSVN содержит скрипты для запуска с правильными настройками когда вы сравниваете файлы с известными расширениями. Вы можете проверить какие расширения файлов поддерживаются, и добавить свои, открыв TortoiseSVN → Настройки и щелкнув Дополнительно в разделе Внешние программы.

Проблемы с Office 2010

Если вы установили версию *Click-to-Run* пакета Office 2010 и пытаетесь сравнить документы, то можете получить сообщение об ошибке из Windows Script Host наподобие этой: «ActiveX component can't create object: word.Application». Это говорит о том, что вы должны использовать MSI версию пакета Office чтобы заработала функция сравнения.

4.11.6. 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* [https://winmerge.sourceforge.net/] is a great open-source diff tool which can also handle directories.

Perforce Merge
: Perforce is a commercial RCS, but you can download the diff/merge tool for free. Get more information from *Perforce* [https://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/].

SourceGear DiffMerge
: SourceGear Vault is a commercial RCS, but you can download the diff/merge tool for free. Get more information from *SourceGear* [https://www.sourcegear.com/diffmerge/].

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
: Similar to ExamDiff Pro, this is an excellent shareware diff tool which can handle directory diffs and unicode. Download it from *Scooter Software* [https://www.scootersoftware.com/].

Araxis Merge
: Araxis Merge is a useful commercial tool for diff and merging both files and folders. It does three-way comparison in merges and has synchronization links to use if you've changed the order of functions. Download it from *Araxis* [https://www.araxis.com/merge/index.html].

В Раздел 4.31.5, «Настройки внешних программ» описано, как настроить TortoiseSVN для использования этих инструментов.

4.12. Добавление новых файлов и папок

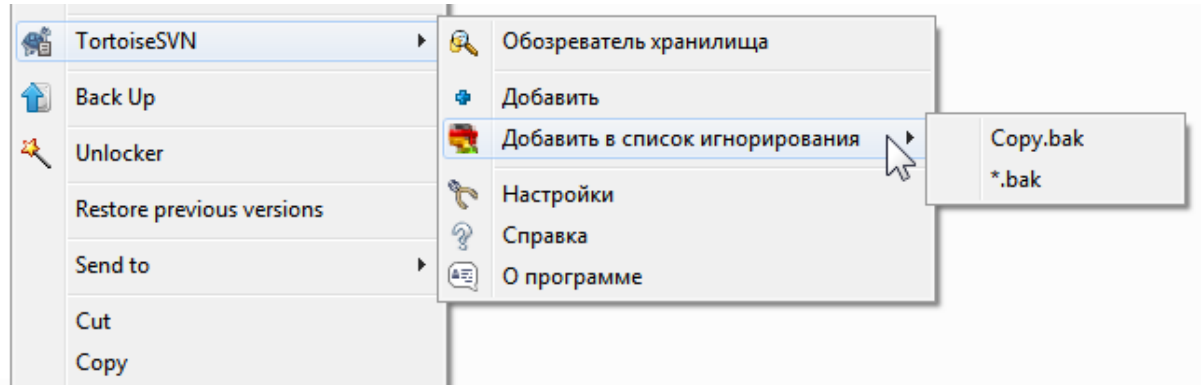


Рисунок 4.31. Контекстное меню Проводника для неверсированных файлов

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....

4.13. 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, 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.

Скопировать файлы и папки из рабочей копии самым простым способом можно с помощью меню открывающегося при перетягивании правой кнопкой мыши. Когда вы перетягиваете правой кнопкой файл или папку из одной рабочей копии в другую или даже в ту же папку, появляется контекстное меню при отпускании кнопки мыши.

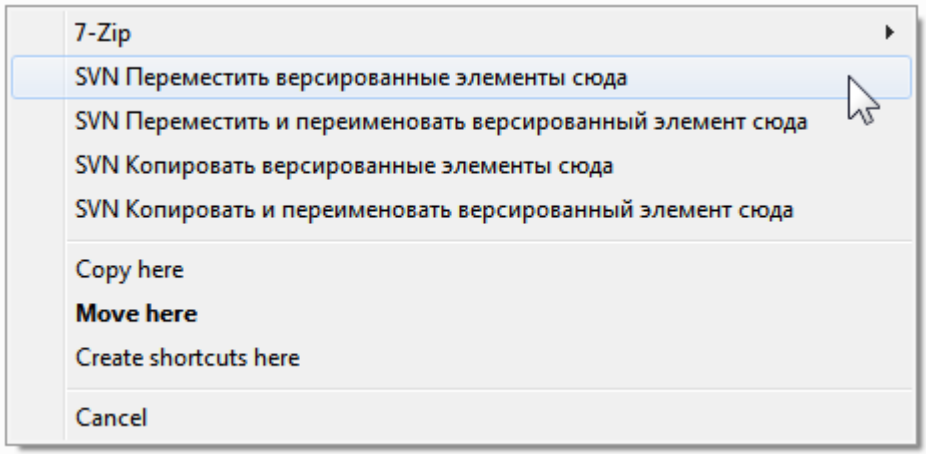


Рисунок 4.32. Меню при перетаскивании правой клавишей мыши для папки под управлением версиями

Теперь вы можете скопировать версированное содержимое в новое место и при этом его переименовать.

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 4.20.1, «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 4.10.3, «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 4.25, «Обозреватель хранилища», чтобы узнать, как это сделать.

Невозможно выполнять копирование между хранилищами

В то время как вы можете копировать или перемещать 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.

4.14. Игнорирование файлов и папок

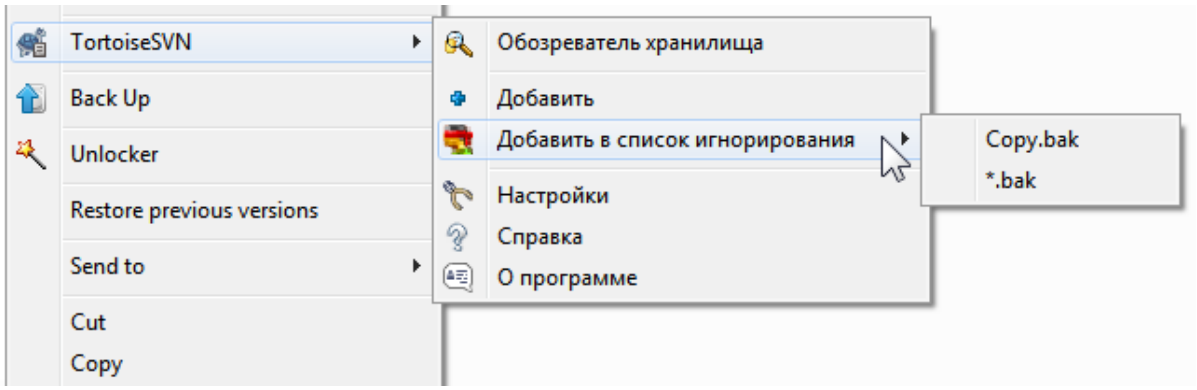


Рисунок 4.33. Контекстное меню Проводника для неверсированных файлов

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 щёлкнете 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. Оба подменю также имеют эквиваленты `(рекурсивно)`. Если вы выберете несколько файлов, подменю не появится, и вы сможете добавить только эти конкретные файлы/папки.

Если вы выберете в контекстном меню игнорирования вариант `(рекурсивно)`, то элемент будет проигнорирован не только в выбранной папке, но также и во всех подпапках. Для этого требуется клиент SVN версии 1.8 или выше.

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 4.18, «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 4.31.1, «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».

4.14.1. 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`.

Соответствие шаблону регистрозависимое, что может привести к проблемам на Windows. Вы можете включить независимость от регистра явным образом удваивая символы, т. е. вместо регистронезависимого шаблона `*.tmp` использовать `*.[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 игнорировать 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 добиться этого при помощи глобальных шаблонов игнорирования.

4.15. Удаление, перемещение и переименование

Subversion позволяет переименовывать и перемещать файлы и папки. Так есть пункты меню для удаления и переименования в подменю TortoiseSVN.

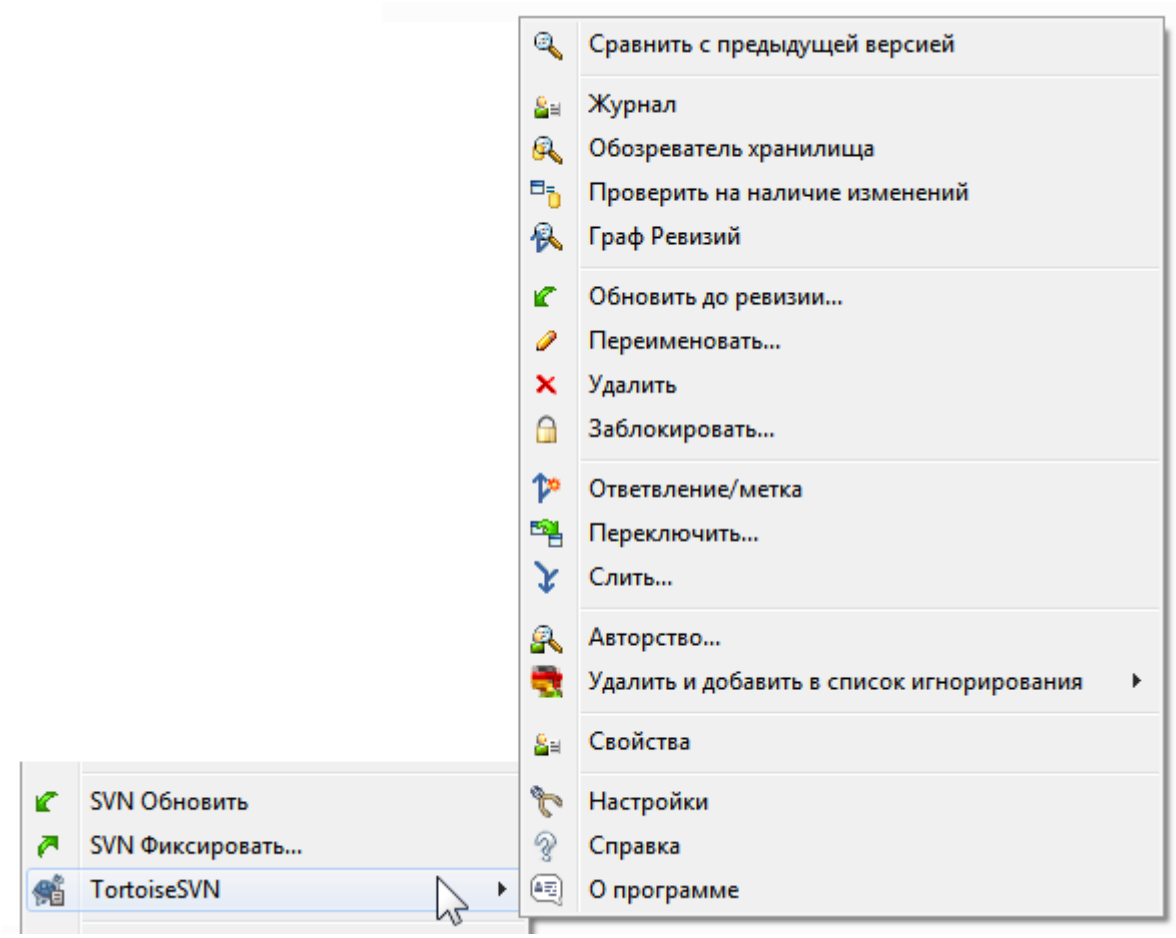


Рисунок 4.34. Контекстное меню Проводника для версированных файлов

4.15.1. Удаление файлов и папок

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

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

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 элемент удаляется в Проводнике, а не при помощи контекстного меню TortoiseSVN, диалог фиксации отобразит такие элементы и позволит вам удалить их также из под управления версиями перед фиксацией. Однако, если вы обновите вашу рабочую копию, Subversion обнаружит отсутствующий элемент 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.

Возвращение назад удалённого файла или папки

Если вы удалили файл или папку и уже зафиксировали эту операцию удаления в хранилище, то обычное TortoiseSVN → Убрать изменения не вернет это назад. Но файл или папка не потеряны навсегда. Если вы знаете ревизию в которой был удален файл или папка (если не знаете — найдите в журнале), то откройте обозреватель хранилища и переключитесь на эту ревизию. Затем выберите удаленный файл или папку, нажмите правую кнопку мыши и выберите Context Menu → Копировать в...

4.15.2. 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 хотите переместить файлы внутри вашей рабочей копии, возможно в другую подпапку, воспользуйтесь обработчиком перетагивания правой кнопкой 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.

Другим способом перемещения или копирования файлов является использования команд Windows копировать/вырезать. Выберите файлы, которые вы хотите скопировать, сделайте правый клик и выберите Context Menu → Копировать в контекстном меню проводника. Затем перейдите в целевую папку, сделайте правый клик и выберите TortoiseSVN → Вставить. Для перемещения файлов выберите Context Menu → Вырезать вместо Context Menu → Копировать.

Можно использовать также обозреватель хранилища для перемещения файлов и папок. Чтобы узнать больше о том, как это сделать, прочтите Раздел 4.25, «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.

4.15.3. 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

There is a server hook script available at: https://svn.apache.org/repos/asf/subversion/trunk/contrib/hook-scripts/ that will prevent checkins which result in case conflicts.

4.15.4. 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.

4.15.5. 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.

4.16. 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 OK.

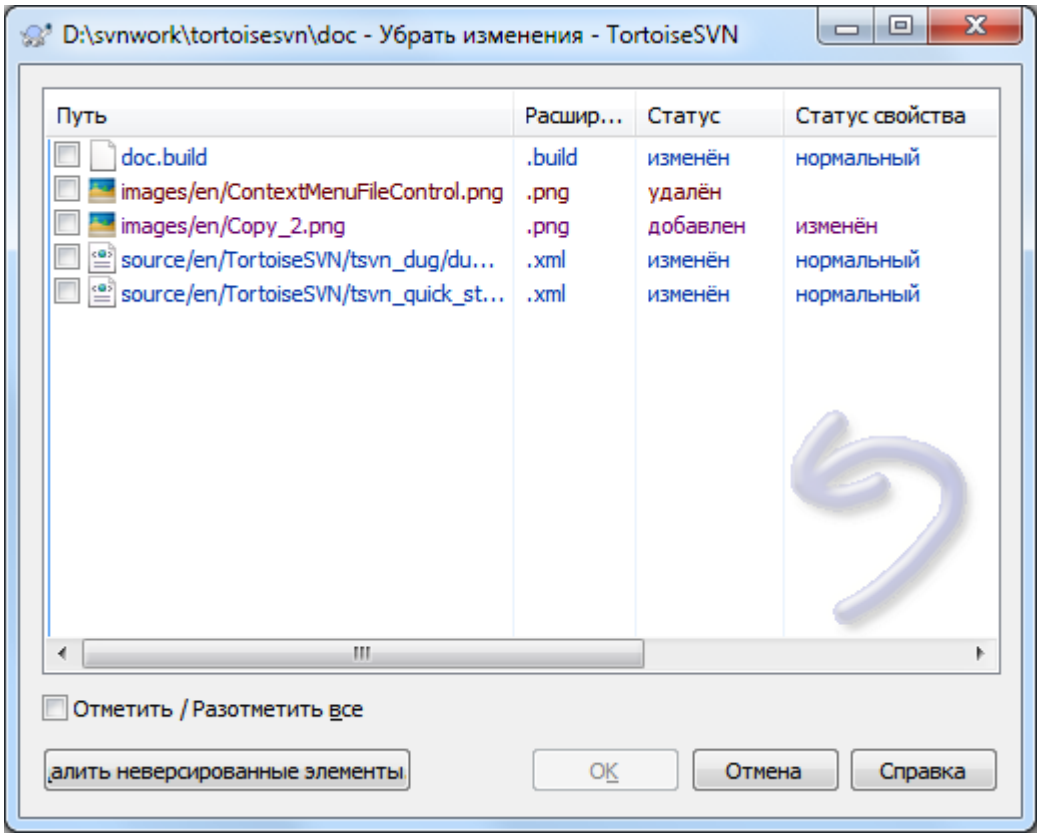


Рисунок 4.35. Диалог 'Убрать изменения'

Если вы также хотите очистить все настроенные списки изменений, то установите флажок в нижней части диалога.

Если вы хотите отменить удаление или переименование, то следует использовать "Убрать изменения" на родительской папке, т. к. удаленный элемент не существует и вы не сделаете на нем правый клик.

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.

Поскольку возврат в прежнее состояние иногда используется для очистки рабочей копии, здесь есть дополнительная кнопка, которая позволяет удалить также и неверсированные элементы. При щелчке на этой кнопке появляется ещё одно окно со списком всех неверсированных элементов, которые вы можете выбрать для последующего удаления.

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 4.10, «Диалоговое окно журнала ревизий» для дополнительной информации.

Отмена 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.

4.17. Очистка

Если команда Subversion не может быть завершена, например, из-за проблем сервера, то ваша рабочая копия может остаться в несогласованном состоянии. В этом случае вам необходимо выполнить команду для папки: TortoiseSVN → Очистка Рекомендуется выполнять эту команду для папки верхнего уровня вашей рабочей копии.

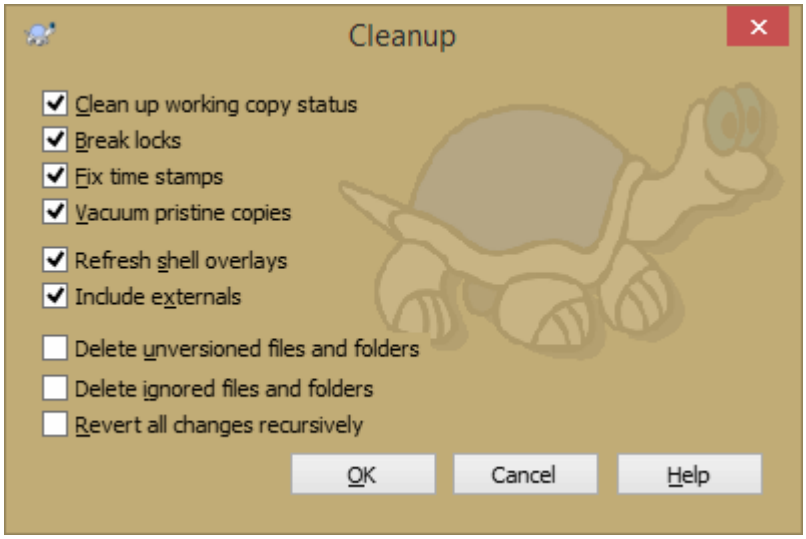


Рисунок 4.36. Диалог Очистки

В диалоге очистки также есть другие полезные варианты для приведения рабочей копии в `чистое` состояние.

Очистить статус рабочей копии
: Как говорилось выше, этот вариант пытается привести рабочую копию из несогласованного в рабочее состояние. Это не затрагивает ваши данные, а только внутреннее состояние базы данных рабочей копии. Фактически это команда `Очистить`, которая знакома вам по предыдущим версиям TortoiseSVN или другим клиентам SVN.

Снять блокировки на запись
: Если опция включена, то из базы данных рабочей копии удаляются все блокировки записи. В большинстве случаев, это необходимо для того, чтобы работала очистка.

  Отключайте эту опцию только в том случае, если рабочая копия используется одновременно другими пользователями/клиентами. Но, если очистка перестанет работать, вам придется включить эту опцию, чтобы очиста заработала.

Исправить временные метки
: Изменения имен всех файлов в момент последней фиксации.

Вакуумные нетронутые копии
: Удаляет неиспользуемые первоначальные копии и сжимает все оставшиеся первоначальные копии файлов рабочей копии.

Обновить оверлеи оболочки
: Иногда оверлеи оболочки, особенно в дереве просмотра в левой части проводника, не отображают текущий статус, или кэш статуса не может определить изменения. В этой ситуации вы можете использовать данную команду для принудительного обновления.

Включая внешние
: Если отмечено, то все действия выполняются для всех файлов и папок включая те, которые со свойством `svn:externals`.

Удалить неверсированные файлы и папки, Удалить игнорируемые файлы и папки
: Это быстрый и простой способ удалить все сгенерированные файлы в вашей рабочей копии. Все неверсированные файлы и папки перемещаются в корзину.

  Примечание: вы можете сделать тоже самое из диалога TortoiseSVN → Убрать изменения. Там вы также получаете список всех неверсированных файлов и папок, чтобы выбрать их для удаления.

Отменить все изменения рекурсивно
: Эта команда отменяет все ваши локальные изменения, которые ещё не зафиксированы.

  Примечание: лучше использовать команду TortoiseSVN → Убрать изменения, потому что вы сначала можете посмотреть и выбрать файлы, в которых хотите убрать изменения.

4.18. Установки проекта

4.18.1. Свойства Subversion

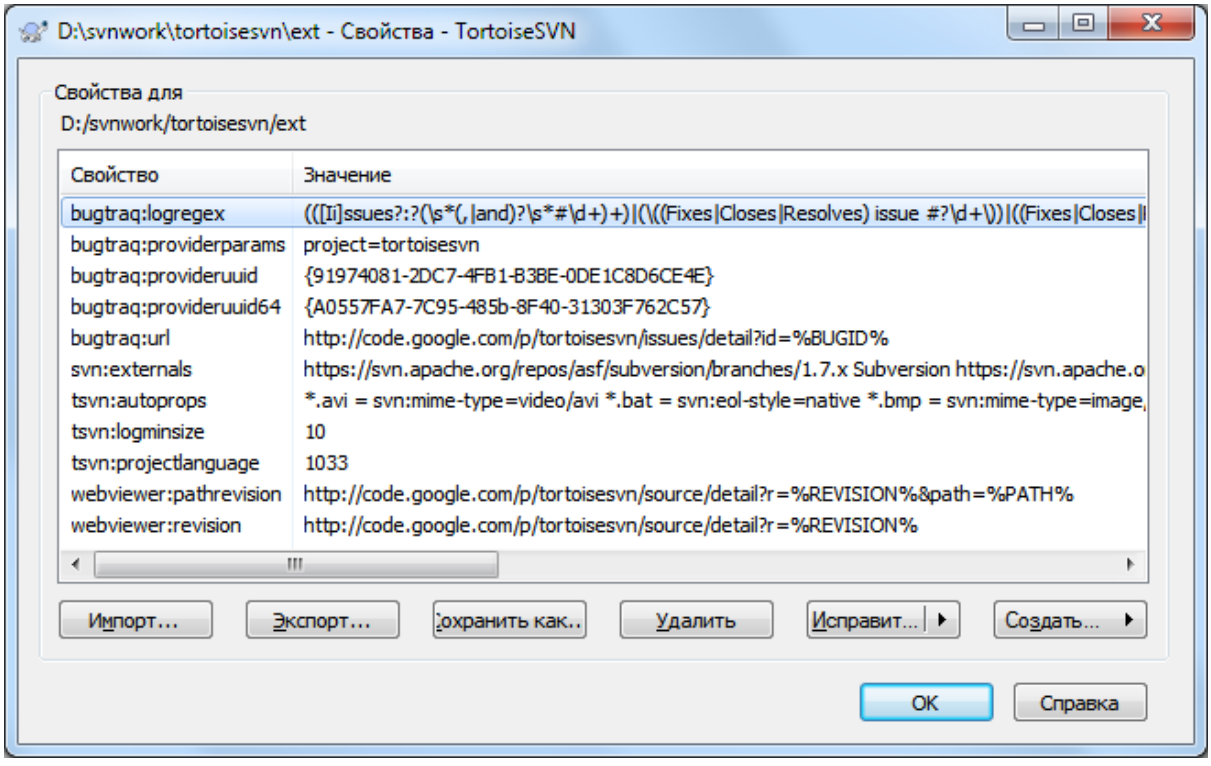


Рисунок 4.37. Страница свойств Subversion

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 4.19, «Внешние включения».

4.18.1.1. 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/1.8/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/1.8/svn.advanced.props.html].

4.18.1.2. Добавление и редактирование свойств

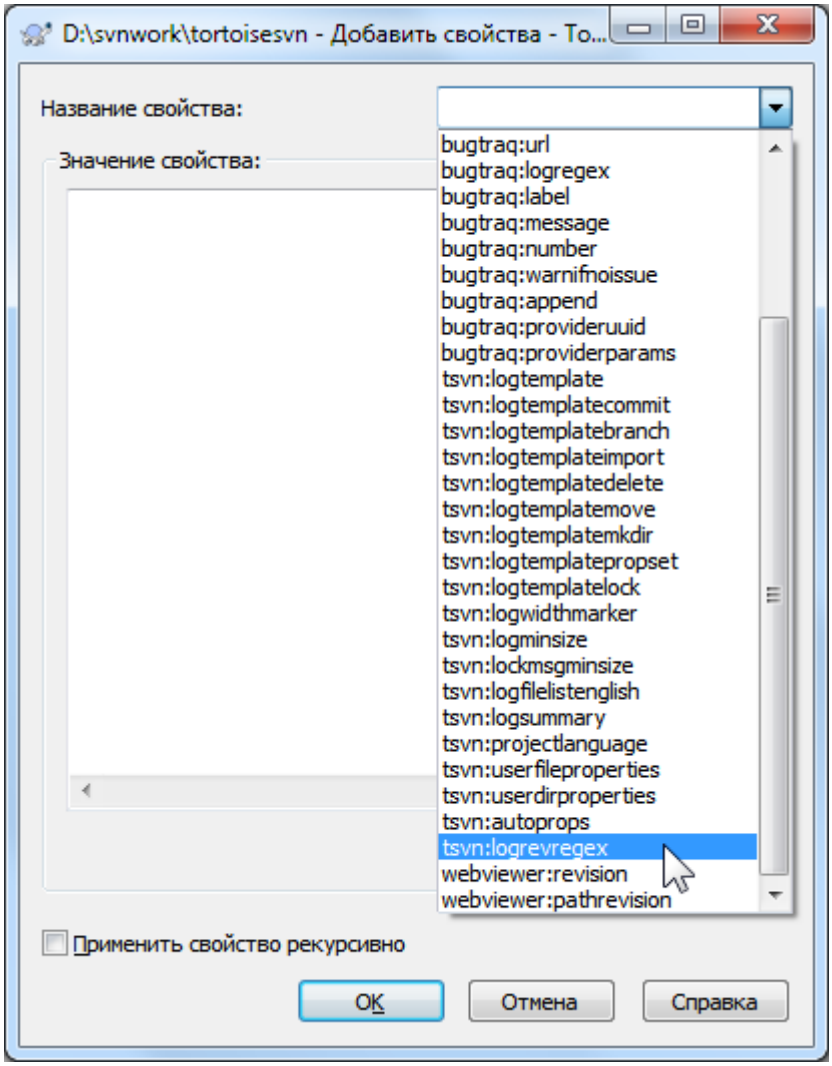


Рисунок 4.38. Добавление свойств

Для добавления нового свойства нажмите Создать.... Выберите требуемое имя свойства из меню, затем введите нужную информацию в диалоге конкретного свойства. Диалоги свойств более подробно описаны в Раздел 4.18.3, «Свойства».

Чтобы добавить свойство, которое не имеет собственного диалога, выберите Дополнительно из меню Создать.... Затем либо выберите существующее свойство из выпадающего списка, либо введите пользовательское имя свойства.

Если хотите применить свойство ко многим элементам за один раз, то выберите файлы/папки в проводнике, потом выберите меню Context menu → свойства.

Если вы желаете применить свойство к *каждому* файлу и папке, расположенному ниже текущей папки в иерархии, выберите флажок Применить свойство рекурсивно.

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 4.19, «Внешние включения».

Редактировать свойства HEAD ревизии

Так как свойства версируются, вы не можете редактировать свойства предыдущих ревизий. Если вы смотрите свойства из диалога журнала или из не-HEAD ревизии в обозревателе хранилища, то вы увидите список свойств и значений без кнопок редактирования.

4.18.1.3. 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.

4.18.1.4. 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.

4.18.1.5. 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.

Начиная с Subversion 1.8, вы можете также установить свойство `svn:auto-props` на корневой папке. Значение свойства автоматически будет унаследовано всеми дочерними элементами.

Какой бы метод вы ни выбрали, имейте ввиду, что auto-props применяется к файлам только в момент добавления в рабочую копию. Auto-props никогда не изменит 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.

4.18.2. 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.

Также есть шаблоны специфичные для действий, которые вы можете использовать вместо `tsvn:logtemplate`. Шаблоны специфичные для действий используются когда они заданы, но если не заданы, то будет использован `tsvn:logtemplate`.

Возможные значения:

- `tsvn:logtemplatecommit` используется для всех фиксаций из рабочей копии.
- `tsvn:logtemplatebranch` используется когда вы создаете ответвление/метку или когда копируете файлы или папки прямо в обозревателе хранилища.
- `tsvn:logtemplateimport` используется для импорта.
- `tsvn:logtemplatedelete` используется при удалении элементов прямо в обозревателе хранилища.
- `tsvn:logtemplatemove` используется при переименовании или перемещении элементов прямо в обозревателе хранилища.
- `tsvn:logtemplatemkdir` используется при создании директорий в обозревателе хранилища.
- `tsvn:logtemplatepropset` используется при изменении свойств в обозревателе хранилища.
- `tsvn:logtemplatelock` используется при получении блокировки.

- 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.

  Начиная с Subversion 1.8, вы должны использовать свойство `svn:auto-props` вместо `tsvn:autoprops`, т.к. оно имеет ту же функциональность, но работает со всеми svn-клиентами, а не только в TortoiseSVN.

- 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 программу проверки орфографии. На Windows 10 используется программа проверки орфографии ОС. На более ранних версиях Windows он может использовать модули программы проверки орфографии, которые также используются OpenOffice и Mozilla. Если они у вас установлены, это свойство будет определять, какую программу проверки орфографии использовать, т.е., на каком языке должны записываться сообщения в журнал вашего проекта. `tsvn:projectlanguage` устанавливает, модуль какого языка должен быть использован при проверке орфографии, когда вы вводите сообщение журнала. Вы можете найти значения для вашего языка на этой странице: *MSDN: Идентификаторы языка* [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` [2].

[2] Для русского значения будут `0x0419` и `1049`, соответственно - прим. переводчика

- 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.

- Свойство `tsvn:logrevregex` задаёт регулярное выражение, соответствующее ссылкам на ревизии в сообщении журнала. Это используется в диалоге журнала, позволяя при щелчке по такой ссылке перейти к указанной ревизии либо в том же окне (если ревизия уже отображается в диалоге журнала, или находится в кэше сообщений), либо открыть новый диалог журнала, показывающий эту ревизию.

  Это регулярное выражение должно соответствовать ссылке целиком, а не только номеру ревизии. Номер ревизии извлекается из обнаруженной строки ссылки автоматически.

  Если это свойство не задано, для создания ссылок на ревизии используется регулярное выражение по умолчанию.

- Существуют несколько свойств для конфигурирования клиентских скриптов обработчиков. Каждое свойство для определенного типа скрипта обработчика.

  Доступные параметры/обработчики

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

  Параметры такие же, как если бы вы сконфигурировали скрипты обработчиков в диалоге настроек. Более подоробно см. Раздел 4.31.8, «Скрипты ловушек, выполняемые на стороне клиента».

  Так как не каждый пользователь имеет свою рабочую копию извлечённую и размещённую в одном и том же месте и с тем же самым именем, то вы можете сконфигурировать выполняемый скрипт/инструмент, который размещён в вашей рабочей копии указав адрес URL в хранилище, используя `%REPOROOT%` как часть URL для указания корня хранилища. Например, если ваш скрипт-обработчик в вашей рабочей копии в `contrib/hook-scripts/client-side/checkyear.js`, то вы должны указать путь к скрипту как `%REPOROOT%/trunk/contrib/hook-scripts/client-side/checkyear.js`. Таким образом, даже если вы переместите ваше хранилище на другой сервер, вам не надо исправлять свойства скрипта-обработчика.

  Вместо переменной `%REPOROOT%` вы также можете указать `%REPOROOT+%`. Символ + используется для вставки любого количества путей к папке, необходимое для поиска скриптом. Это полезно, если вы хотите создать скрипт, который бы находил рабочую копию даже при при создании отдельной ветки и изменения её url. Пример использования: `%REPOROOT+%/contrib/hook-scripts/client-side/checkyear.js`.

  Следующий скриншот демонстрирует как в TortoiseSVN сконфигурирован скрипт для проверки года авторского права в заголовках файлов исходников.

- 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.

  Вы также можете указать, используется ли специальный диалог для добавления/редактирования свойства. TortoiseSVN предлагает четыре различных диалога, в зависимости от типа вашего свойства.

  bool
  : Если ваше свойство может иметь только два состояния, например, истина и ложь, то вы можете сконфигурировать свойство типа `булев`.

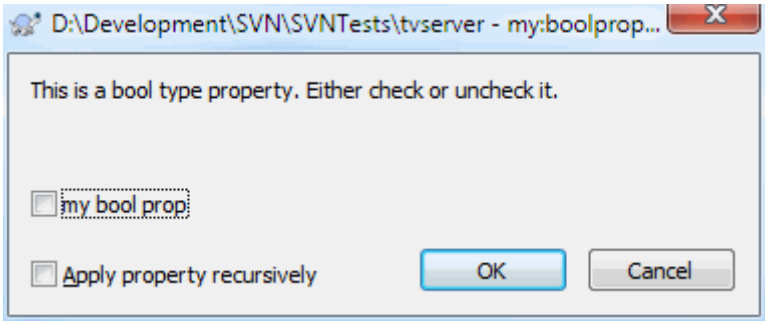


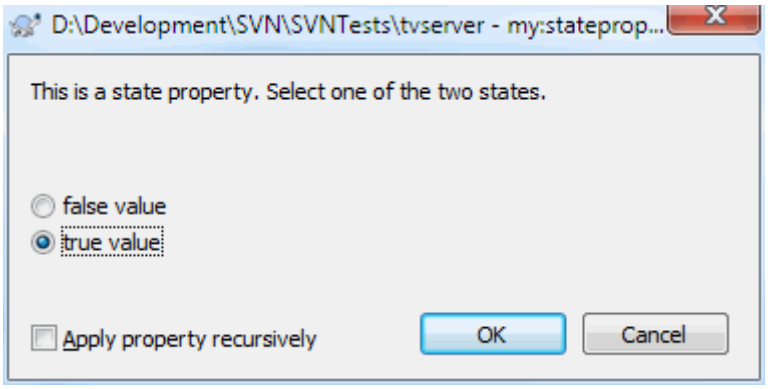
Рисунок 4.39. Диалог свойства пользовательских булевых типов

  Определите ваше свойство вот так:

```
propertyname=bool;labeltext(YESVALUE;NOVALUE;Checkboxtext)
```

  `labeltext` — это текст, отображаемый в диалоге над флажком, в котором вы можете пояснить назначение и использование свойства. Остальные параметры говорят сами за себя.

  state
  : Если ваше свойство представляет одно из возможных значений, например, `да, нет, может быть`, то вы можете сконфигурировать ваше свойство как свойство `состояние`



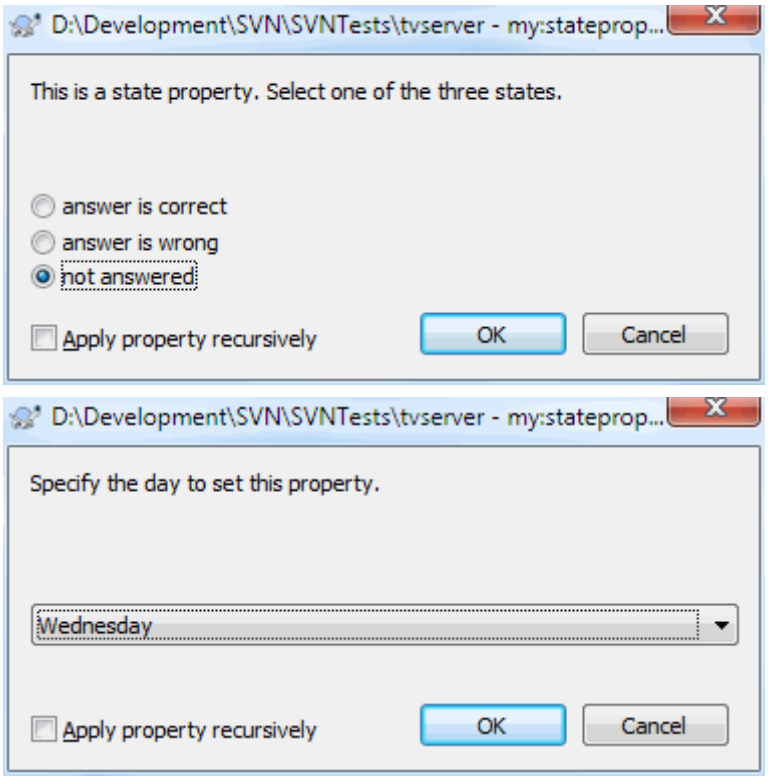


Рисунок 4.40. Диалог свойства пользовательских типов состояния

вот так:

```
propertyname=state;labeltext(DEFVAL;VAL1;TEXT1;VAL2;TEXT2;VAL3;TEXT3;...)
```

Параметры такие же как для свойства `булев`, где `DEFVAL` — значение по умолчанию, используемое если свойство еще не установлено или имеет значение, которое не сконфигурировано.

Для не более трёх разных значений в диалоге отображается до трёх переключателей. Если сконфигурировано больше значений, то используется выпадающий список, из которого пользователь может выбрать нужное состояние.

singleline
: Для свойств, которые состоят из одной строки текста используйте свойство типа `однострочный`:

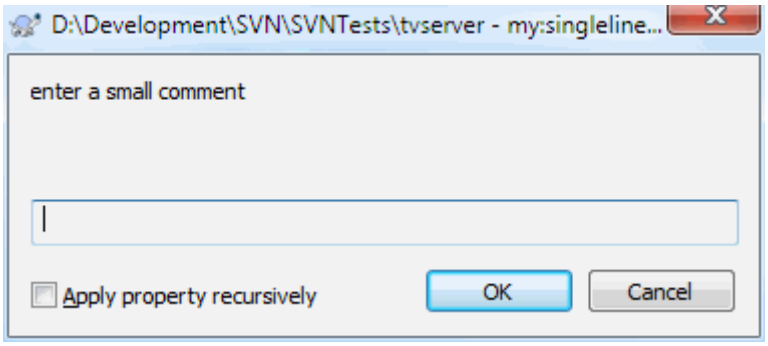


Рисунок 4.41. Диалог свойства пользовательских однострочных типов

```
propertyname=singleline;labeltext(regex)
```

`regex` определяет регулярное выражение, которое используется для проверки соответствия введенного пользователем текста. Если текст не соответствует регулярному выражению, то пользователю показывается ошибка и свойство не устанавливается.

multiline
: Для свойств, которые состоят из множества строк текста, используйте свойство типа `многострочный`:

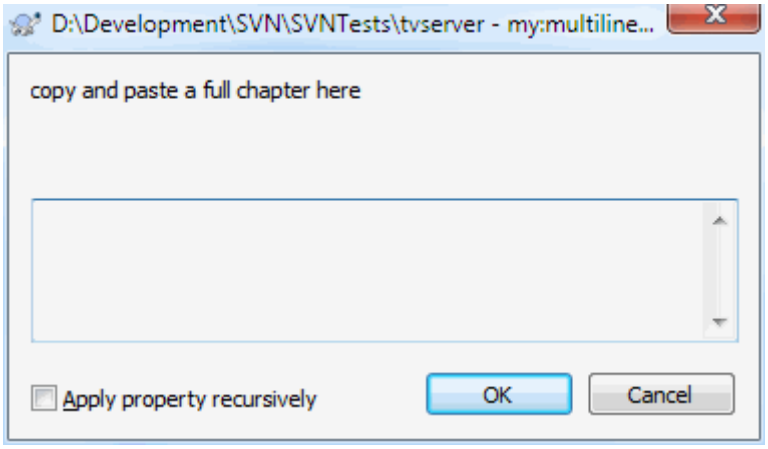


Рисунок 4.42. Диалог свойства пользовательских многострочных типов

```
propertyname=multiline;labeltext(regex)
```

`regex` определяет регулярное выражение, которое используется для проверки соответствия введенного пользователем текста. Не забывайте добавлять символ новой строки (\n) в регулярное выражение!

Приведенные выше скриншоты были сделаны со следующими свойствами `tsvn:userdirproperties`:

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

TortoiseSVN может интегрироваться с некоторыми средствами отслеживания ошибок. При этом используются свойства проекта, начинающиеся с `bugtraq:`. Прочтите Раздел 4.29, «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 4.30, «Интеграция со средствами просмотра хранилища, работающими через веб-интерфейс» для дополнительной информации.

Устанавливайте свойства проекта на папках

Для того чтобы система работата, эти специальные свойства проекта должны быть установлены на *папках*. Когда вы используете команду TortoiseSVN, которая использует эти свойства, свойства читаются из папки на которой вы кликнули. Если там свойства не найдены, то TortoiseSVN будет искать вверх по дереву папок пока не дойдет до неверсированной папки

или корня дерева (например, `C:\`). Если вы можете быть уверены, что каждый пользователь извлекает только из, допустим, `trunk/`, а не какой-то подпапки, тогда достаточно установить свойства для `trunk/`. Если вы не можете быть в этом уверены, то вы должны установить свойства рекурсивно для каждой подпапки. Если вы устанавливаете одно и то же свойство, но используете разные значения на разных уровнях иерархии проекта, то в зависимости от того где кликните в структуре папок вы получите разные результаты.

Только для свойств проекта, т.е. для `tsvn:`, `bugtraq:` и `webviewer:`, вы можете использовать флажок Рекурсивно для установки свойства для всех подпапок в иерархии, без установки его также для всех файлов.

Если вы добавляете новые подпапки в рабочую копию используя TortoiseSVN, то любые свойства проекта, заданные в родительской папке, будут автоматически добавлены в новые дочерние папки.

Ограничения по использованию обозревателя хранилища

Удаленная выборка свойств может занять много времени, поэтому некоторые из свойств описанных выше не будут работать в обозревателе хранилища так, как они работают в рабочей копии.

- Если вы добавляете свойство используя обозреватель хранилища, то только стандартные `svn:` свойства будут доступны на выбор в предопределённом списке. Любое другое имя свойства должно быть добавлено вручную.
- Свойства не могут быть заданы или удалены рекурсивно с помощью обозревателя хранилища.
- Свойства проекта *НЕ* будут автоматически скопированы, если дочерняя папка добавляется посредством обозревателя хранилища.
- Свойство `tsvn:autoprops` *НЕ* задает свойства файлам, добавляемых посредством обозревателя хранилища.

Внимание

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.

4.18.3. Свойства

Некоторые свойства должны использовать определенные значения или должны быть отформатированы особым образом, чтобы быть использованными автоматически. Чтобы упростить получение правильного форматирования некоторых свойств TortoiseSVN предоставляет диалоги редактирования, которые показывают возможные значения или разбивают свойство на отдельные компоненты.

4.18.3.1. Внешнее содержимое

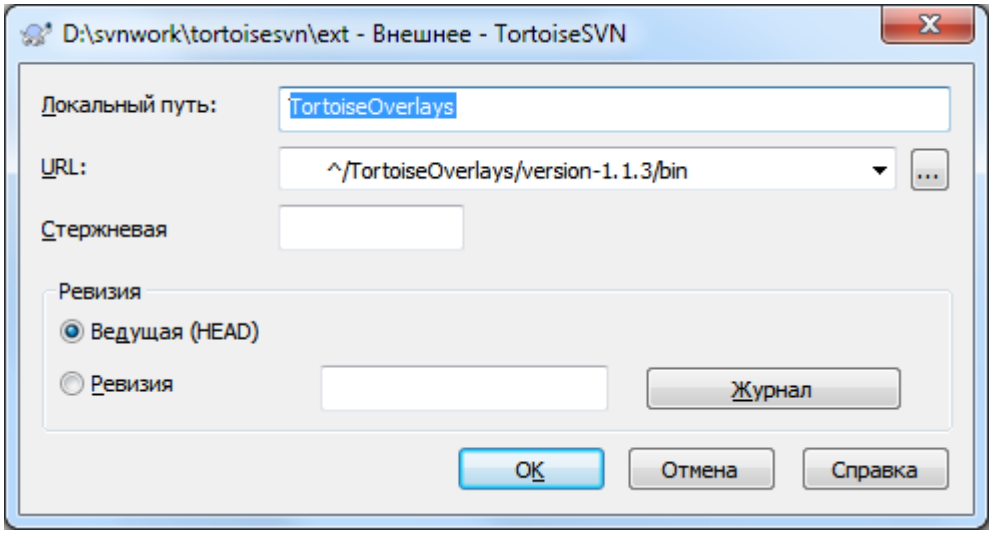


Рисунок 4.43. страница свойств svn:externals

Свойство `svn:externals` может быть использовано для втягивания других проектов из того же хранилища или из совсем другого хранилища, как это описано в Раздел 4.19, «Внешние включения».

Вам необходимо определить имя подпапки, в которую извлекается внешняя папка, и URL-адрес внешнего элемента. Вы можете извлекать внешний элемент с ревизии HEAD, так что при изменении внешнего элемента в хранилище ваша рабочая копия получит эти изменения при обновлении. Однако, если вы хотите, чтобы внешний элемент ссылался на определенную постоянную точку, то вы можете указать определенную ревизию. В этом случае вы возможно захотите указать ту же ревизию как опорную (peg) ревизию. Если внешний элемент переименован в какой-то момент в будущем, то Subversion не сможет обновить этот элемент в вашей рабочей копии. Указывая опорную (peg) ревизию вы сообщаете Subversion, что надо искать элемент по имени в опорной (peg) ревизии, а не в ревизии HEAD.

Кнопка Найти HEAD-ревизию получает ревизию HEAD для каждого URL-адреса внешнего определения и показывает эту HEAD ревизию в самой правой колонке. После того, как ревизия HEAD известна, простой правый клик на внешнем определении даёт возможность установить опорную (peg) ревизию выбранного внешнего определения на явную ревизию HEAD. В случае если ревизия HEAD не известна, то команда по правому клику сначала получит ревизию HEAD.

4.18.3.2. Ключевые слова SVN

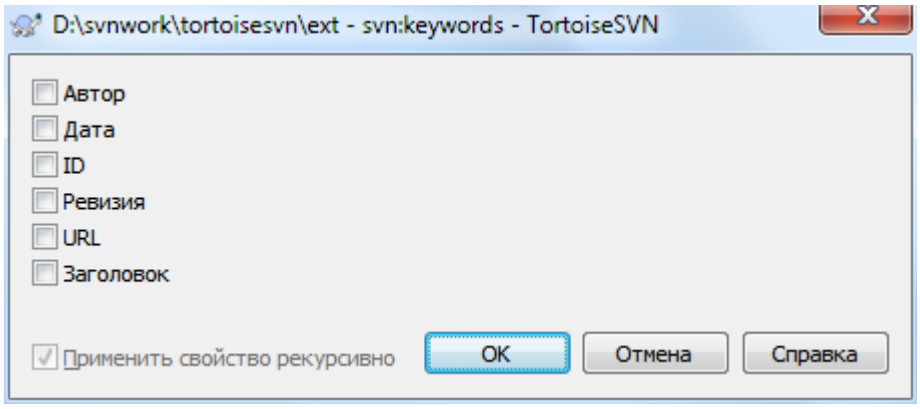


Рисунок 4.44. страница свойств svn:keywords

Выберите ключевые слова, которые должны быть развернуты в вашем файле.

4.18.3.3. Стиль переноса строк

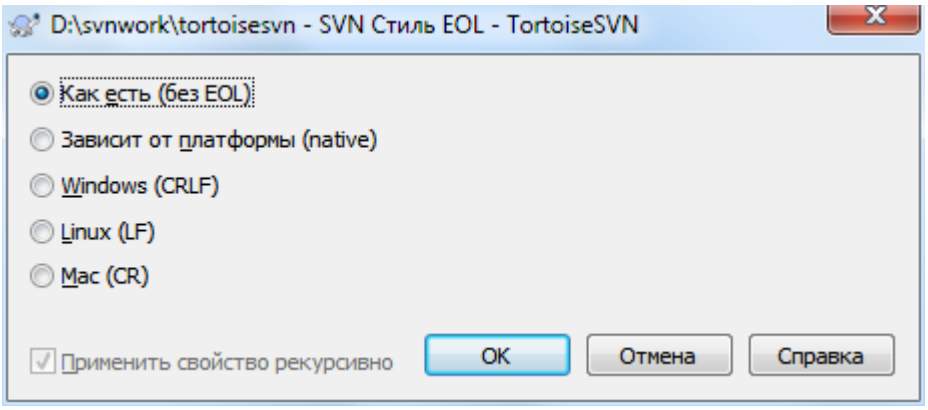


Рисунок 4.45. страница свойств svn:eol-style

Выберите какой стиль окончания строки вы хотите использовать и TortoiseSVN будет использовать правильное значение свойства.

4.18.3.4. Интеграция с системами отслеживания проблем

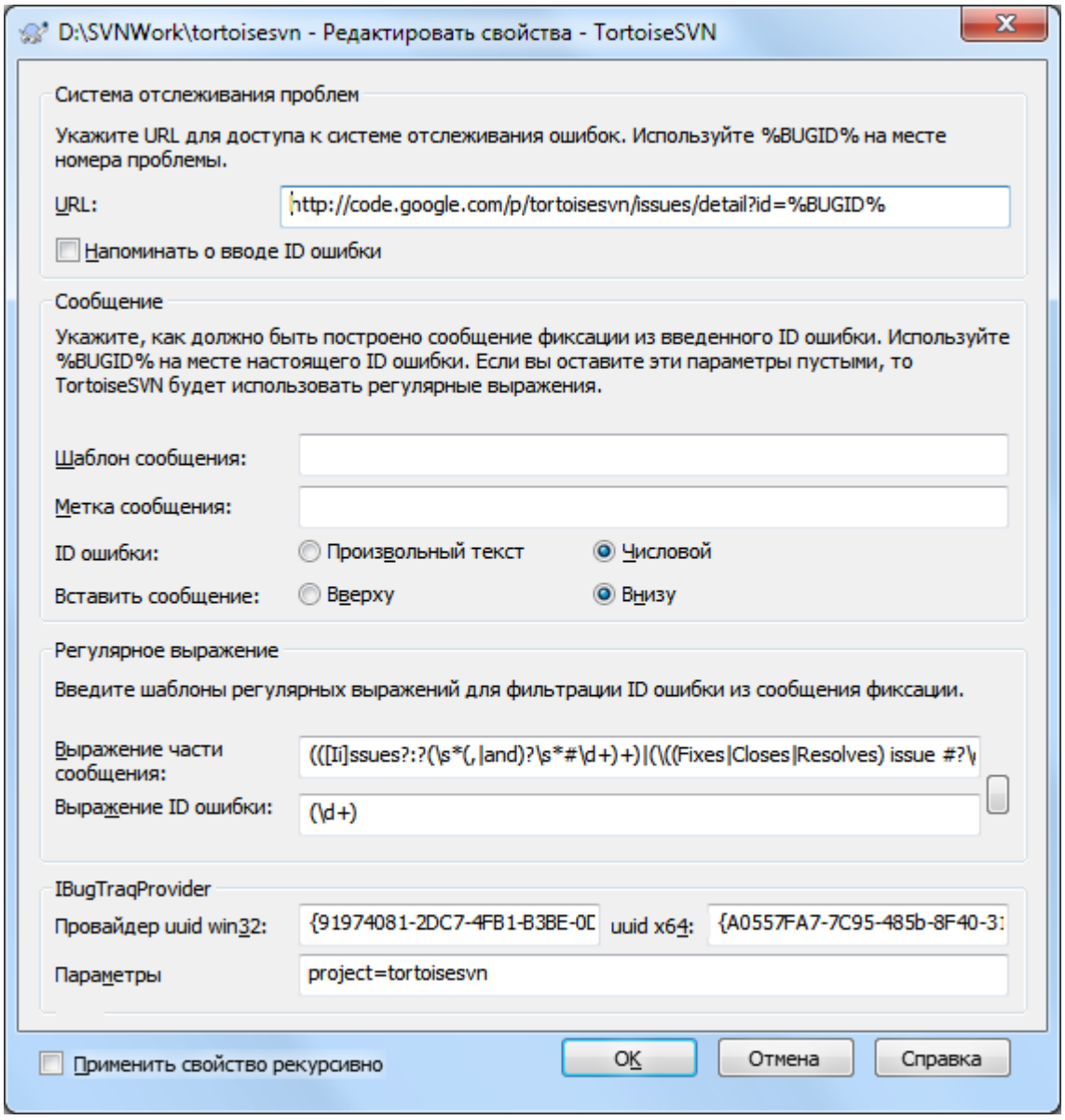


Рисунок 4.46. страница свойств tsvn:bugtraq

4.18.3.5. Размеры журнала сообщений

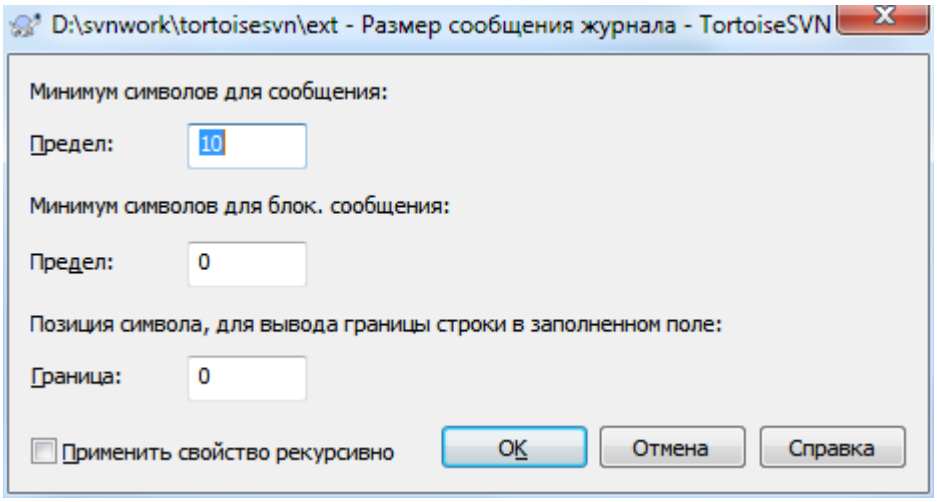


Рисунок 4.47. Страница свойств размер журнала сообщений

Эти 3 свойства управляют форматированием сообщений журнала. Первые 2 отключают OK в диалогах фиксации или блокировки до тех пор, пока сообщение меньше заданной длины. Позиция границы показывает маркер в заданной ширине колонки в качестве руководства для проектов, в которых есть пределы ширины для сообщений журнала. Установка значения в ноль удалит свойство.

4.18.3.6. Язык проекта

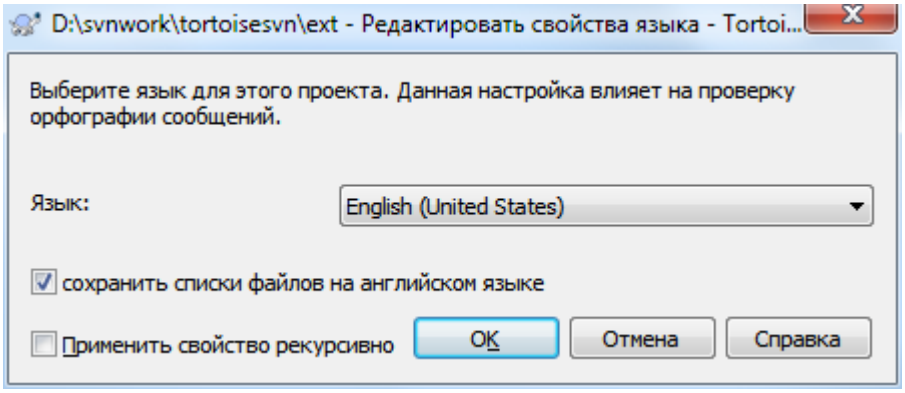


Рисунок 4.48. Страница свойств язык

Выберите язык для проверки орфографии в сообщениях журнала в диалоге фиксации. Флажок списки файлов срабатывает когда вы делаете правый клик в панели сообщения журнала и выбираете Вставить список имен фалов. По умолчанию статус Subversion отображается на локальном языке. Когда этот флажок установлен, то статус всегда будет на английском для проектов, которым требуются неанглийские сообщения журнала.

4.18.3.7. MIME-type

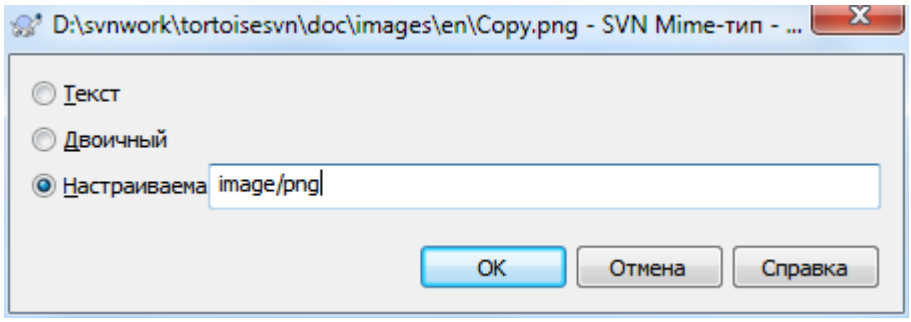


Рисунок 4.49. Страница свойств svn:mime-type

4.18.3.8. svn:needs-lock

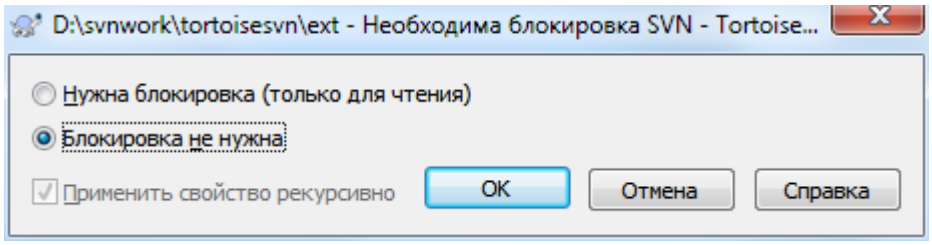


Рисунок 4.50. Страница свойств svn:needs-lock

Это свойство просто определяет будет ли файл извлечен с атрибутом "только чтение" если на него нет блокировки в рабочей копии.

4.18.3.9. svn:executable

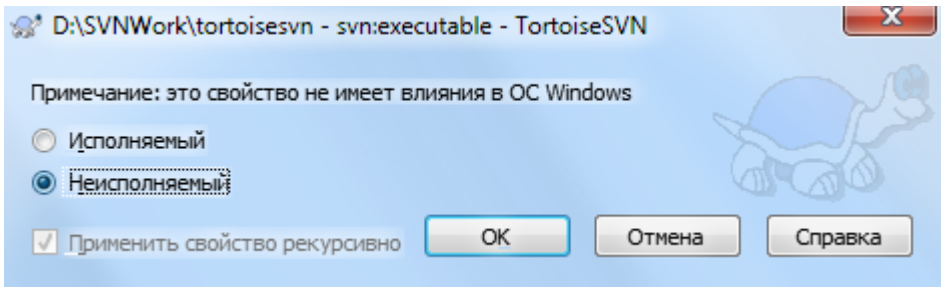


Рисунок 4.51. Страница свойств svn:executable

Это свойство определяет будет ли файл отмечен как исполняемый при извлечении на системах Unix/Linux. Оно не имеет эффекта при извлечении на Windows.

4.18.3.10. Шаблоны сообщения журнала для слияния

Всякий раз, когда ревизии объединяются в рабочей копии, TortoiseSVN создает сообщение журнала из всех объединённых ревизий. Они потом доступны по кнопке Недавние сообщения в диалоге фиксации.

Вы можете настроить это созданное сообщение следующими свойствами:

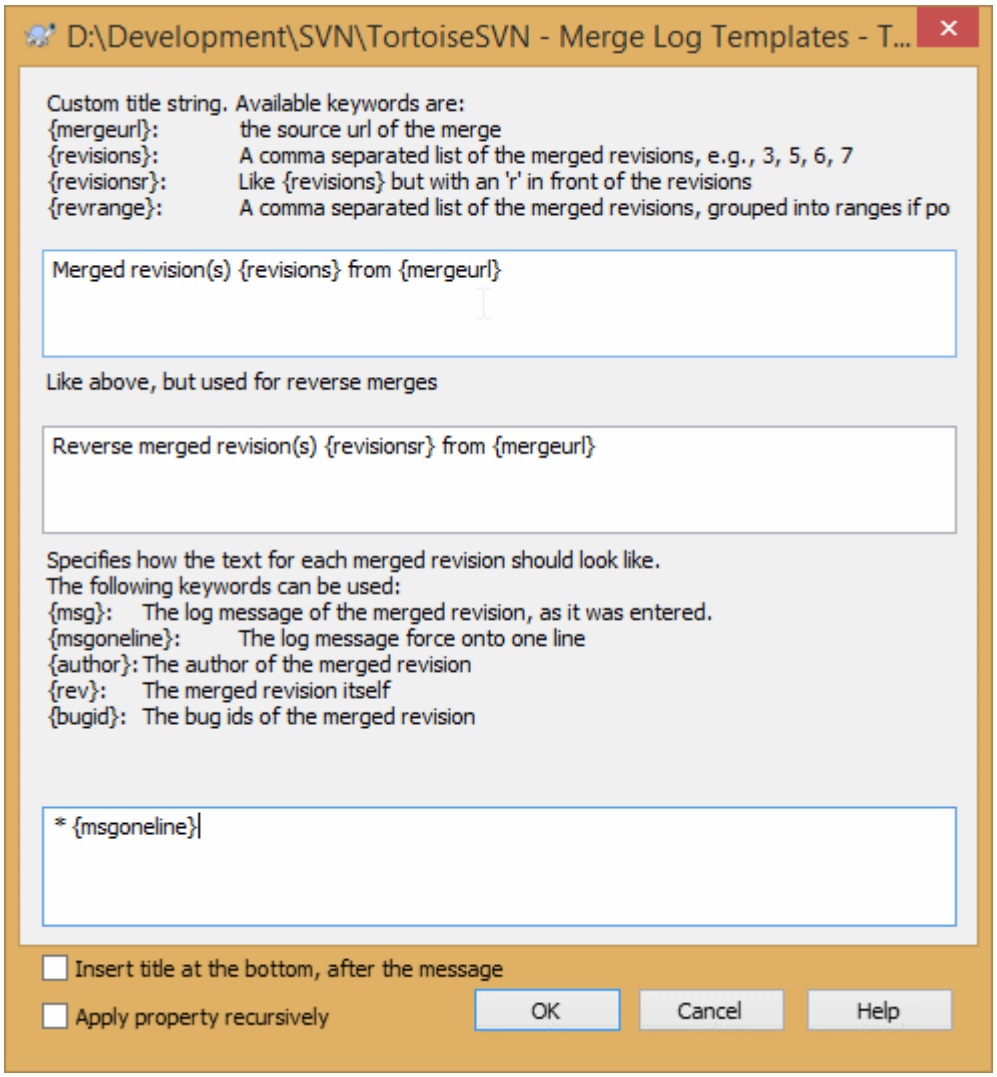


Рисунок 4.52. Диалог свойства шаблонов сообщения журнала для слияния

tsvn:mergelogtemplatetitle, tsvn:mergelogtemplatereversetitle
: Это свойство определяет первую часть созданного сообщения журнала. Можно использовать следующие ключевые слова:

  {revisions}
  : Список объединённых ревизий, разделенный запятыми, например, `3, 5, 6, 7`

  {revisionsr}
  : Также как `{revisions}`, но каждая ревизия начинается с `r`, например, `r3, r5, r6, r7`

  {revrange}
  : Список объединённых ревизий, разделенный запятыми, при возможности объединённый в диапазоны, например, `3, 5-7`

  {mergeurl}
  : Исходный URL-адрес слияния, т.е. откуда объединяются ревизии.

  Значение по умолчанию для этой строки `Объединены ревизии {revrange} из {mergeurl}:` с символом новой строки в конце.

tsvn:mergelogtemplatemsg
: Это свойство определяет как должен выглядеть текст для каждой объединённой ревизии. Можно использовать следующие ключевые слова:

{msg}
Сообщение журнала объединённой ревизии, как оно было введено.

{msgoneline}
Также как `{msg}`, но все символы новой строки заменены на пробелы так, что всё сообщение целиком размещается в одну строку.

{author}
Автор объединённой ревизии.

{rev}
Сама объединённая ревизия.

{bugids}
Идентификаторы (ID) ошибок объединённой ревизии, если таковые есть.

tsvn:mergelogtemplatemsgtitlebottom
Это свойство устанавливает позицию строки заголовка заданного в `tsvn:mergelogtemplatetitle` или `tsvn:mergelogtemplatereversetitle`. Если свойство установлено в `да` или `истина`, то строка заголовка добавляется снизу вместо верха.

Важно

Это работает только, если объединённые ревизии уже в кэше журнала. Если вы отключили кэш журнала или не просматривали журнал перед слиянием, то созданное сообщение не будет содержать информацию об объединённых ревизиях.

4.19. 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.

4.19.1. Внешние папки

Скажем вы извлекаете рабочую копию проекта `/project1` в `D:\dev\project1`. Выберите папку `D:\dev\project1`, сделайте правый клик и выберите меню Windows → Свойства из контекстного меню. Появляется диалог Свойства. Перейдите на вкладку Subversion. Там вы можете установить свойства. Нажмите Свойства.... В диалоге свойств, либо кликните дважды на `svn:externals` если есть, либо нажмите на кнопку Создать... и выберите `внешнее` из меню. Для добавления новой внешней ссылки нажмите на кнопке Создать... и заполните требуемую информацию в появившемся диалоге.

Внимание

Адреса URL должны быть правильно экранированы иначе они не будут работать, т. е. вы должны заменить каждый пробел на `%20`.

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 явных номеров ревизии во всех ваших определениях внешних включений, как описано выше. Применение этого "подхода означает, что вы будете решать, когда скачивать другую копию внешней информации, и какую конкретно копию. 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.

Диалог редактирования свойств `svn:externals` позволяет вам выбрать внешние определения и автоматически установить их явно на ревизию HEAD.

Если внешний проект находится в том же хранилище, то любые изменения в нем будут добавлены в список фиксации, когда вы будете фиксировать ваш главный проект.

Если внешний проект находится в другом хранилище, любые изменения, внесенные во внешний проект будут показаны или указаны, когда Вы фиксируете основной проект, но Вам следует 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 извлечёт `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`.

Если необходимо вы также можете указать опорную (peg) и оперативную ревизию для адреса URL. Чтобы больше узнать об опорных (peg) и оперативных ревизиях прочитайте, пожалуйста, *соответствующую главу* [http://svnbook.red-bean.com/en/1.8/svn.advanced.pegrevs.html] в книге о Subversion.

Важно

Если вы укажете папку назначения для внешних включений как подпапку, как в приведённых выше примерах, убедитесь, что *все* папки межу ними также версионируемые. Так для вышеприведённых примеров папка `common` должна быть версионируемой!

Несмотря на то, что внешние включения будут работать правильно в большинстве случаев, если папки между ними неверсируемые, всё же есть несколько операций, которые не будут работать как вы ожидаете. И статус оверлейных значков также не будет отображаться корректно.

Если вам необходимо больше информации о том, как TortoiseSVN обходится со свойствами, прочтите раздел Раздел 4.18, «Установки проекта».

Для того, чтобы узнать о различных методах доступа к общим подпроектам, прочтите Раздел B.6, «Включить общий подпроект».

4.19.2. 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.

- Путь к внешнему файлу должен быть прямым потомком папки, которая задана в свойстве `svn:externals`.

- URL внешнего файлового включения должен вести в то же хранилище, в которое внешнее файловое включение будет вставлено; между-хранилищные 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`.

4.19.3. Создание внешних включений с помощью перетаскивания (drag-and-drop)

Если у вас уже есть рабочая копия с файлами и папками, которые вы хотите добавить как внешнее включение в другую рабочую копию, то просто добавьте их с помощью перетаскивания из проводника Windows.

Просто перетяните правой кнопкой мыши файл или папку из одной рабочей копии туда, куда вы хотите их добавить как внешнее включение. При отпускании кнопки мыши появится контекстное меню: SVN Добавить сюда как внешнее. Если вы кликнете на этом пункте меню, то автоматически будет добавлено свойство `svn:externals`. Всё что вам нужно сделать после этого — зафиксировать изменения свойства и обновить, чтобы эти внешние включения правильно добавились в рабочую копию.

4.20. 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.

4.20.1. Создание ответвления или метки

Если вы импортировали ваш проект с рекомендованной структурой папок, создать ответвление или метку очень просто:

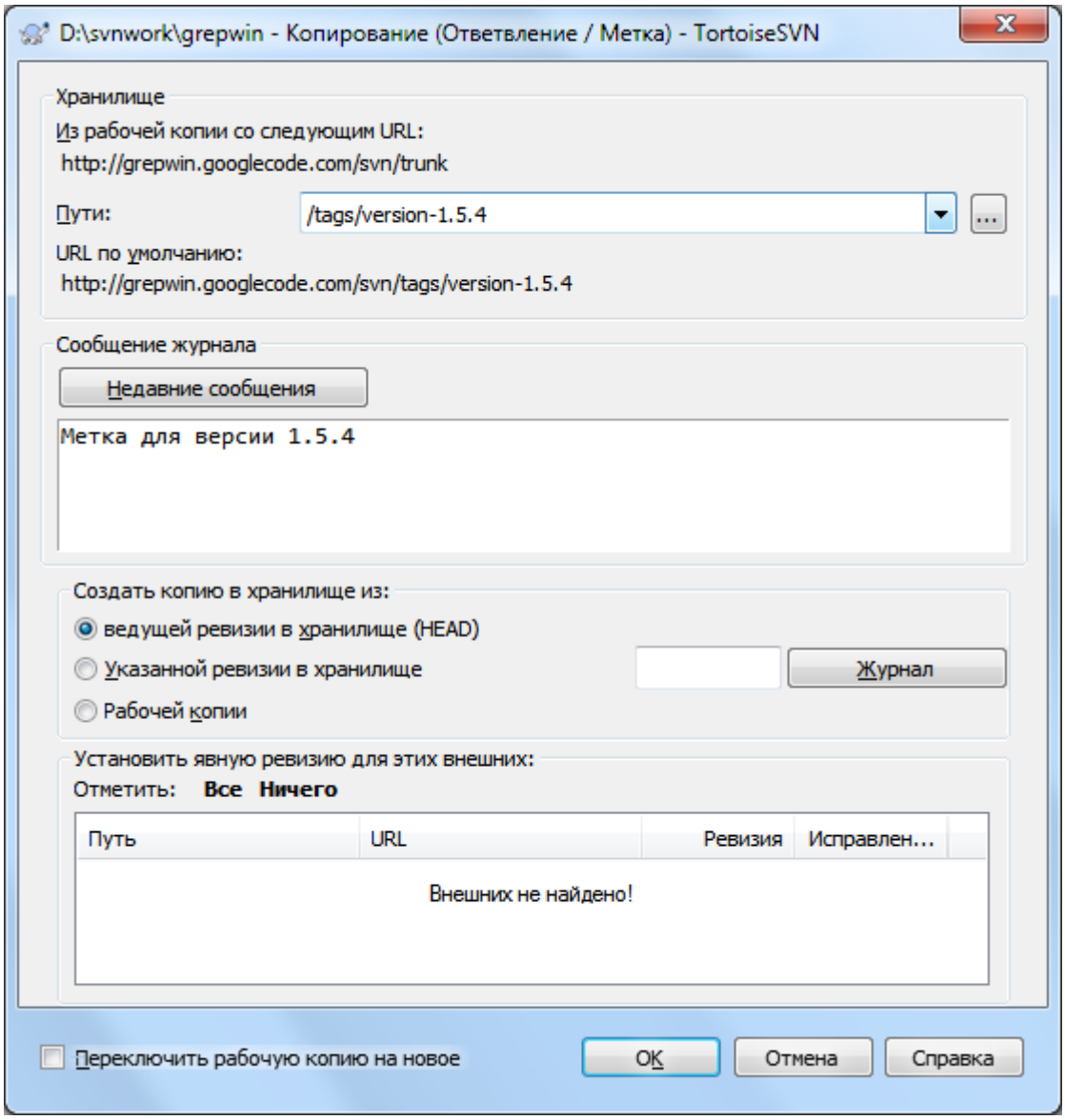


Рисунок 4.53. Диалог создания ответвления/метки

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.

промежуточные папки

Когда вы указываете URL-адрес назначения все папки от первой до самой последней должны существовать, иначе вы получите сообщение об ошибке. В вышеприведенном примере URL-адрес `http://svn.collab.net/repos/ProjectName/tags/` должен существовать, чтобы создать метку `Release_1.10`.

Однако, если вы хотите создать ответвление/метку на URL, в котором есть промежуточные папки, которые ещё не существуют, то вы можете отметить параметр `Создавать промежуточные папки` в нижней части диалога. Если этот параметр включен, то все промежуточные папки создаются автоматически.

Заметьте, что этот параметр по умолчанию отключен во избежание опечаток. Например, если вы набрали URL назначения как `http://svn.collab.net/repos/ProjectName/Tags/Release_1.10` вместо `http://svn.collab.net/repos/ProjectName/tags/Release_1.10`, то вы можете получить ошибку при отключённом параметре, но при включённом параметре папка `Tags` будет создана автоматически, и в результате получится папка `Tags` и папка `tags`.

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.

Если ваша рабочая копия содержит другие проекты добавленные через свойство `svn:externals`, то такие внешние проекты будут перечислены внизу диалога ветки/метки. Для каждого внешнего проекта отображается путь назначения и исходный адрес URL.

Если вы хотите убедиться, что новая метка всегда в согласованном состоянии, то проверьте чтобы все ревизии внешних ссылок были закреплены. Если вы этого не сделаете, и такие ссылки будут указывать на ревизию HEAD, что в будущем может изменится, то извлечение новой метки извлечет ревизию HEAD внешнего проекта и ваша метка может больше не скомпилироваться. Так что явно устанавливать ревизию для внешних проектов при создании метки — это хорошая идея.

Внешние включения автоматически закрепляются на текущей ревизии HEAD или ревизии BASE рабочей копии в зависимости от источника ветки/метки:

Источник копии	Закреплённая ревизия
Ведущая ревизия в хранилище (HEAD)	ревизия HEAD внешнего хранилища
Указанной ревизии в хранилище	ревизия HEAD внешнего хранилища

Источник копии	Закреплённая ревизия
Рабочая копия	ревизия BASE внешней рабочей копии

Таблица 4.1. Закреплённая ревизия

внешние внутри внешних

Если проект, который добавлен как внешний, сам имеет внешние включения, то они не будут помечены тэгом. Только внешние включения, которые являются прямыми потомками, могут быть помечены тэгом.

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.

4.20.2. Другие способы создания ответвления или метки

Вы также можете создать ответвление или метку и не имея рабочей копии. Для этого откройте обозреватель хранилища. В нём вы можете перетаскивать папки в новое место. Вам необходимо удерживать клавишу **Ctrl** при перетаскивании для создания копии, иначе папка будет перемещена, а не скопирована.

Вы также можете перетаскивать папки при помощи правой кнопки мыши. После отпускания кнопки вы сможете выбрать из контекстного меню нужное действие: перемещение или копирование. Конечно же, для создания ответвления или метки вы должны скопировать папку, а не переместить.

Еще один способ — это диалог журнала. Вы можете открыть диалог журнала, например, для trunk, выбрать ревизию (ревизию HEAD на самом верху или ранюю ревизию), сделать правый клик и выбрать Создать ответвление/метку из ревизии....

4.20.3. 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.

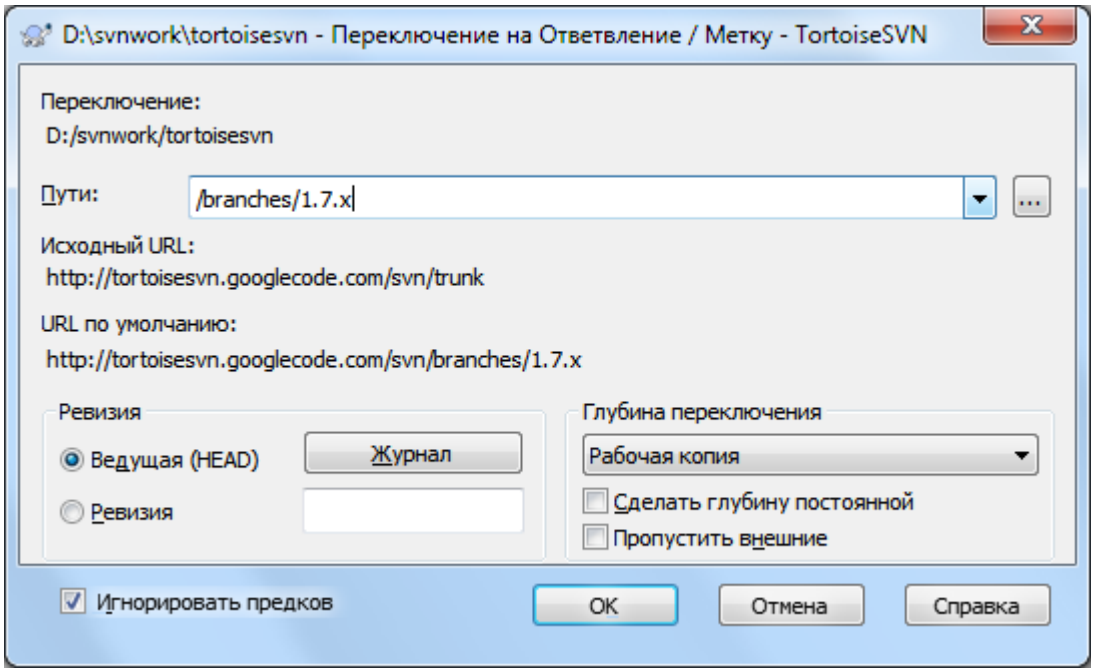


Рисунок 4.54. Диалог переключения

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.

4.21. 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/1.8/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 диапазона 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). »

  Если вы оставили диапазон ревизий пустым, то Subversion использует функции отслеживания слияния для вычисления правильного диапазона ревизий. Это известно как воссоединительное или автоматическое слияние.

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/1.8/svn.advanced.vendorbr.html] в Книге о Subversion.

4.21.1. Слияние с диапазоном ревизий

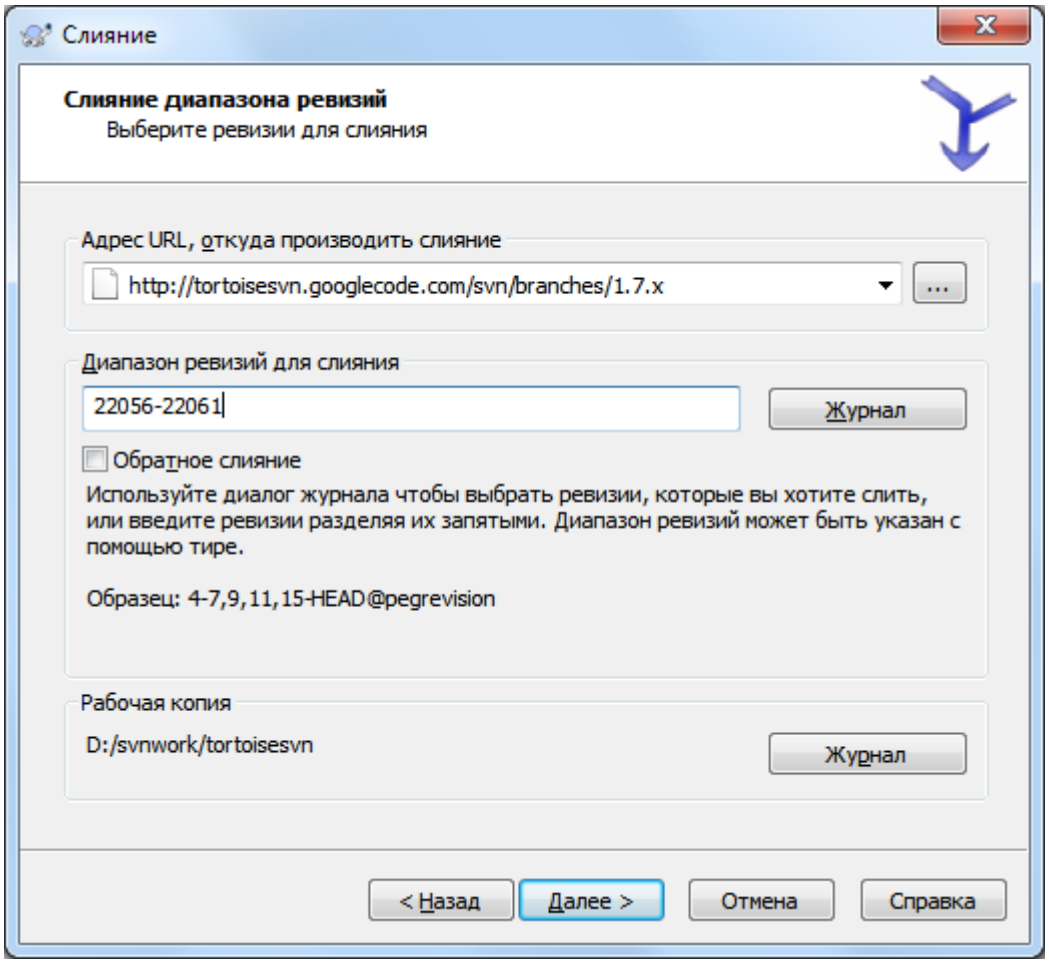


Рисунок 4.55. Мастер слияния - выбор диапазона ревизий

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.

Если вы выполняете слияние из переименованного или удалённого ответвления, то вы вы должны вернуться к ревизии, в которой это ответвление ещё существовало. В этом случае вам также надо указать эту ревизию в качестве опорной (peg) в сливаемом диапазоне ревизий (см. ниже), в противном случае слияние не получится, т.к. путь не будет найден в ревизии HEAD.

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.

Вы должны указать опорную (peg) ревизию для слияния, добавить опорную ревизию в конец ревизий, например, `5-7,10@3`. В вышеприведенном примере ревизии 5,6,7 и 10 будут слиты, при этом ревизия 3 опорная.

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 командной 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 ОК 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 4.21.5, «Отслеживание слияний» для того, чтобы узнать об этом больше.

Когда используется отслеживание слияний диалог журнала покажет предыдущие слитые ревизии и серым цветом ревизии предшествующие общей точке-предку, т. е. перед тем как ветка была скопирована. Флажок Скрыть несливаемые ревизии позволяет вам полностью отфильтровать такие ревизии, так чтобы вы могли видеть только *возможные* для слияния ревизии.

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.

Если вы оставили диапазон ревизий пустым или выбрали переключатель все ревизии, то Subversion сливает все необъединённые ревизии. Это известно как воссоединительное или автоматическое слияние.

Несколько условий воссоединительного слияния. Для начала, сервер должен поддерживать отслеживание слияний. Рабочая копия должна быть с бесконечной глубиной извлечения (без неполных извлечений), и в ней не должно быть локальных изменений, переключенных элементов, или элементов, обновлённых до ревизий, отличных от ведущей. Все изменения, произведённые в стволе за время разработки в ответвлении, должны быть слиты обратно в ответвление (или помечены как слитые). Диапазон ревизий для слияния будет вычислен автоматически.

Нажмите Далее и перейдите к Раздел 4.21.3, «Параметры слияния».

4.21.2. Слияние двух различных деревьев

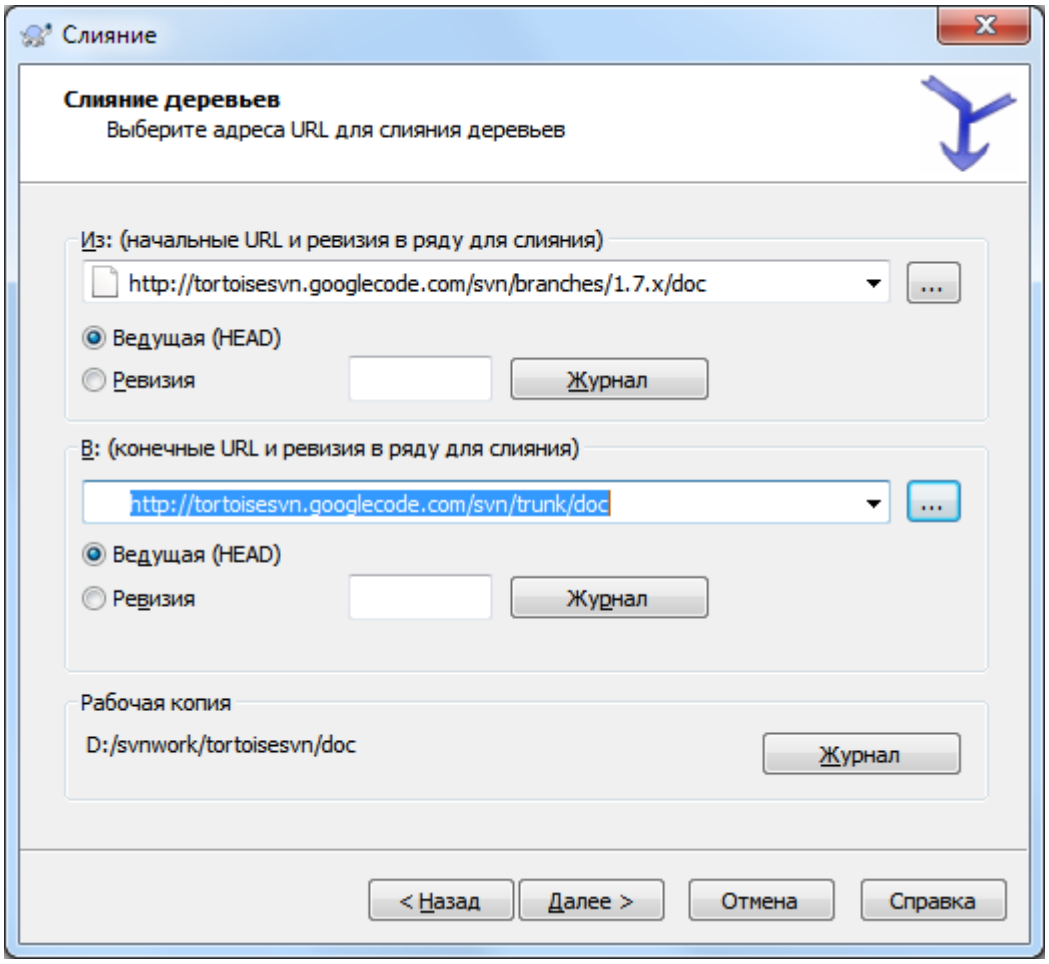


Рисунок 4.56. Мастер слияния - слияние деревьев

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.

4.21.3. 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.

Почти всё время вы хотите сливать с учетом истории файлов, так чтобы изменения по сравнению с общим предком объединяются. Иногда вам может понадобиться слить файлы, которые возможно связаны, но не в вашем хранилище. Например, у вас могут быть импортированы версии 1 и 2 сторонней библиотеки в две разные директории. Несмотря на это, логически они связаны, но Subversion об это не знает, т. к. ему видны только импортированные архивы. Если вы попытаетесь слить различия между этими двумя деревьми, то можете увидеть полное удаление и следующее за ним полное добавление. Чтобы заставить Subverison использовать различия по путям, а не по истории, отметьте флажок Игнорировать предков. Почитайте на эту тему книгу по Subversion *Noticing or Ignoring Ancestry* [http://svnbook.red-bean.com/en/1.8/svn.branchmerge.advanced.html#svn.branchmerge.advanced.ancestry].

Вы можете задать способ обработки изменений завершений строк и пробельных символов. Эти параметры описаны в Раздел 4.11.2, «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.

Флажок Принудительное слияние используется для избежания конфликта деревьев, в котором поступающее удаление затрагивает файл, который или изменён локально, или вообще не является версированным. Если удалить этот файл, то нет способа его восстановить, и поэтому этот флажок по умолчанию выключен.

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.

После задания нужных параметров, вам остается только нажать кнопку Слить. Чтобы выполнить предварительный просмотр результатов - нажмите кнопку Пробное и TortoiseSVN симулирует операцию слияния, но *НЕ* будет делать никаких изменений в рабочей версии. Вы увидите список всех файлов, которые будут изменены при реальном слиянии, а также файлы, в которых *могут* быть конфликты. Поскольку слежение за слиянием еще больше усложнит процесс слияния, то не существует идеального способа проверить завершится ли слияние без конфликтов, так как файлы, отмеченные как конфликтные в пробном слиянии, могут на самом деле слиться без всяких проблем.

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.

4.21.4. 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.

Подсказка

Всякий раз, когда ревизии объединяются в рабочей копии, TortoiseSVN создает сообщение журнала из всех объединённых ревизий. Они потом доступны по кнопке Недавние сообщения в диалоге фиксации.

Для настройки этого созданного сообщения установите соответствующие свойства проекта в вашей рабочей копии. Смотрите Раздел 4.18.3.10, «Шаблоны сообщения журнала для слияния»

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.

4.21.5. 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 информации об отслеживании слияния найдете в документации Subversion *Merge tracking documentation* [http://svn.apache.org/repos/asf/subversion/trunk/notes/merge-tracking/index.html].

4.21.6. Обработка конфликтов после слияния

Важно

The text in the conflict resolver dialogs are provided by the SVN library and might therefore not (yet) be translated as the TortoiseSVN dialogs are. Sorry for that.

Merging does not always go smoothly. Sometimes there is a conflict. TortoiseSVN helps you through this process by showing the *merge conflict* dialog.

Рисунок 4.57. Диалог конфликта слияния

It is likely that some of the changes will have merged smoothly, while other local changes conflict with changes already committed to the repository. All changes which can be merged are merged. The Merge Conflict dialog gives you different ways of handling the lines which are in conflict.

For normal conflicts that happen due to changes in the file content or its properties, the dialog shows buttons which allow you to chose which of the conflicting parts to keep or reject.

Postpone
: Don't deal with the conflict now. Let the merge continue and resolve the conflicts after the merge is done.

Accept base
: This leaves the file as it was, without neither the changes coming from the merge nor the changes you've made in your working copy.

Accept incoming
: This discards all your local changes and uses the file as it arrives from the merge source.

Reject incoming
: This discards all the changes from the merge source and leaves the file with your local edits.

Accept incoming for conflicts
: This discards your local changes where they conflict with the changes from the merge source. But it leaves all your local changes which don't conflict.

Reject conflicts
: This discards changes from the merge source which conflict with your local changes. But it keeps all changes that don't conflict with your local changes.

Пометить как улаженный
: Marks the conflicts as resolved. This button is disabled until you use the button Edit to edit the conflict manually and save those changes back to the file. Once the changes are saved, the button becomes enabled.

Редактировать
: Starts the merge editor so you can resolve the conflicts manually. Don't forget to save the file so the button Mark as resolved becomes enabled.

If there's a tree conflict, please first see Раздел 4.6.3, «Конфликты деревьев» about the various types of tree conflicts and how and why they can happen.

To resolve tree conflicts after a merge, a dialog is shown with various options on how to resolve the conflict:

Рисунок 4.58. The Merge Tree Conflict Dialog

Since there are various possible tree conflict situations, the dialog will show buttons to resolve those depending on the specific conflict. The button texts and labels explain what the option to resolve the conflict does. If you're not sure, either cancel the dialog or use the Postpone button to resolve the conflict later.

4.21.7. Сопровождение ответвления разработки новой возможности

Когда вы разрабатываете новую возможность в отдельном ответвлении, хорошей мыслью является выработка политики реинтеграции этой новой возможности по окончанию её разработки . Если в `стволе` в это же время выполняется другая работа, то вы можете обнаружить, что различия со временем 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.

Другой (быстрый) способ слить все изменения из ствола (trunk) в ветку новой функции это использовать TortoiseSVN → Слить всё... из расширенного контекстного меню (удерживайте клавишу **Shift** при правом клике на файле).

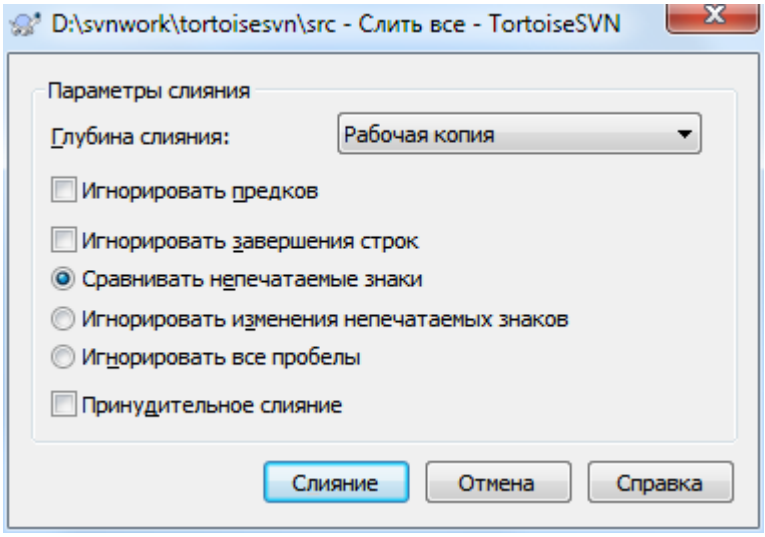


Рисунок 4.59. Диалог 'Слить Всё'

Этот диалог очень простой. Всё что вам надо сделать – это задать опции слияния, как описано в Раздел 4.21.3, «Параметры слияния». Остальное сделает TortoiseSVN автоматически используя механизм отслеживания слияния.

4.22. 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.

Три значения «Блокировки»

В этом разделе, и практически везде в этой книге, слова «блокировка» и «блокирование» описывает механизм взаимного исключения между пользователями, чтобы избежать коллизий при фиксациях. К сожалению, есть два других вида «блокировок» с которыми Subversion, следовательно и эта книга, иногда приходится иметь дело.

Второе — это `блокировки рабочей копии`, используемые внутренне Subversion для предотвращения коллизий между несколькими клиентами Subversion работающими с одной и той же рабочей копией. Обычно вы получаете эти блокировки когда команда, например, обновить/фиксировать/... прервана из-за ошибки. Эти блокировки могут быть удалены выполнением команды очистки в рабочей копии, как описано в Раздел 4.17, «Очистка».

И третье, файлы и папки могут быть заблокированы если используются другим процессом, например, если у вас открыт документ в Word, то этот документ заблокирован и TortoiseSVN не имеет к нему доступа.

В общем случае вы можете забыть об этих видах блокировок пока не случится что-нибудь, что потребует вашего вмешательства. В этой книге «блокировка» означает первый вид, если обратное не указано явно или следует из контекста.

4.22.1. 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.

Важно

Блокировка назначается не на определенного пользователя, а на определенного пользователя и рабочую копию. Иметь блокировку в одной рабочей копии также предохраняет того же пользователя от фиксации блокированного файла из другой рабочей копии.

Например, представьте, что у пользователя Ивана есть рабочая копя на его офисном ПК. Там он начинает работать с изображением и поэтому делает блокировку этого файла. Когда он покидает свой офис он ещё не закончил работать с этим файлом, так что он не снимает эту

> блокировку. Дома у Ивана также есть рабочая копия, и он решает немного поработать над проектом. Но он не может изменить или фиксировать изменения в тот же самый файл, так как блокировка файла сделана в рабочей копии в офисе.

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.

4.22.2. Получение блокировки

Выберите файл(ы) в рабочей копии, которые вы желаете заблокировать, после чего выполните команду TortoiseSVN → Заблокировать....

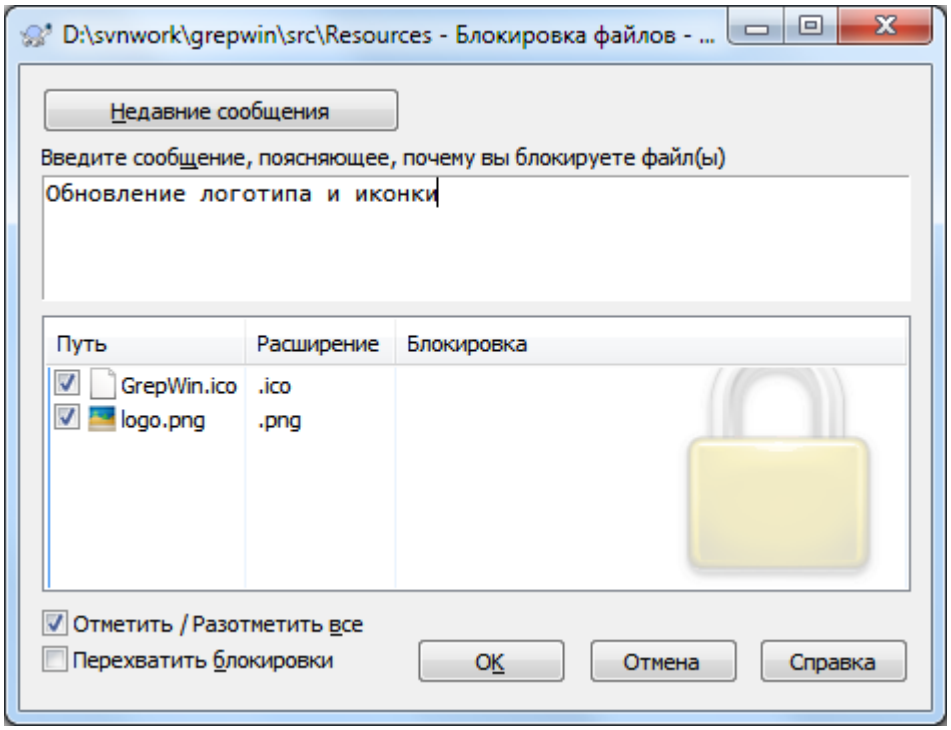


Рисунок 4.60. Диалог блокировки

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.

Вы можете установить свойство проекта `tsvn:logtemplatelock` чтобы предоставить пользователям шаблон сообщения для заполнения в качестве сообщения блокировки. Обратитесь к Раздел 4.18, «Установки проекта» за инструкциями по установке свойств.

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...

4.22.3. 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.

4.22.4. Проверка состояния блокировки

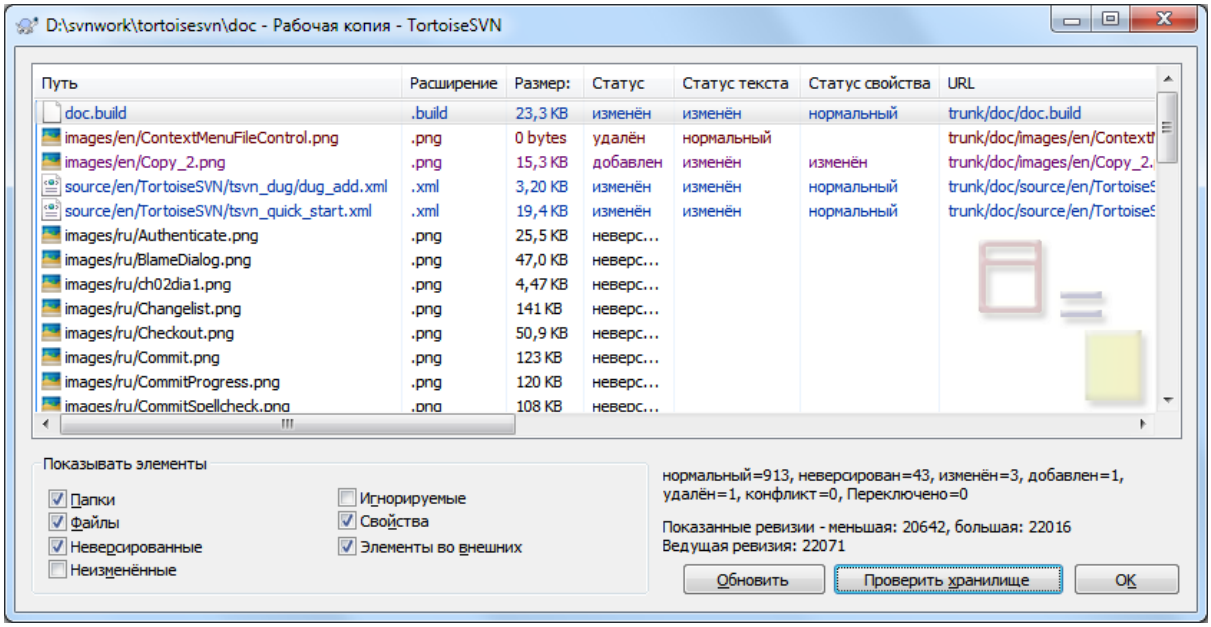


Рисунок 4.61. Диалог проверки на наличие изменений

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.

4.22.5. 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 4.18, «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 4.18.1.5, «Автоматическая установка свойств».

4.22.6. 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.

4.23. 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.

4.23.1. Создание файла заплатки

Сначала вы должны сделать *и проверить* ваши изменения. Затем, вместо использования на родительской папке TortoiseSVN → Фиксировать..., выберите TortoiseSVN → Создать заплатку...

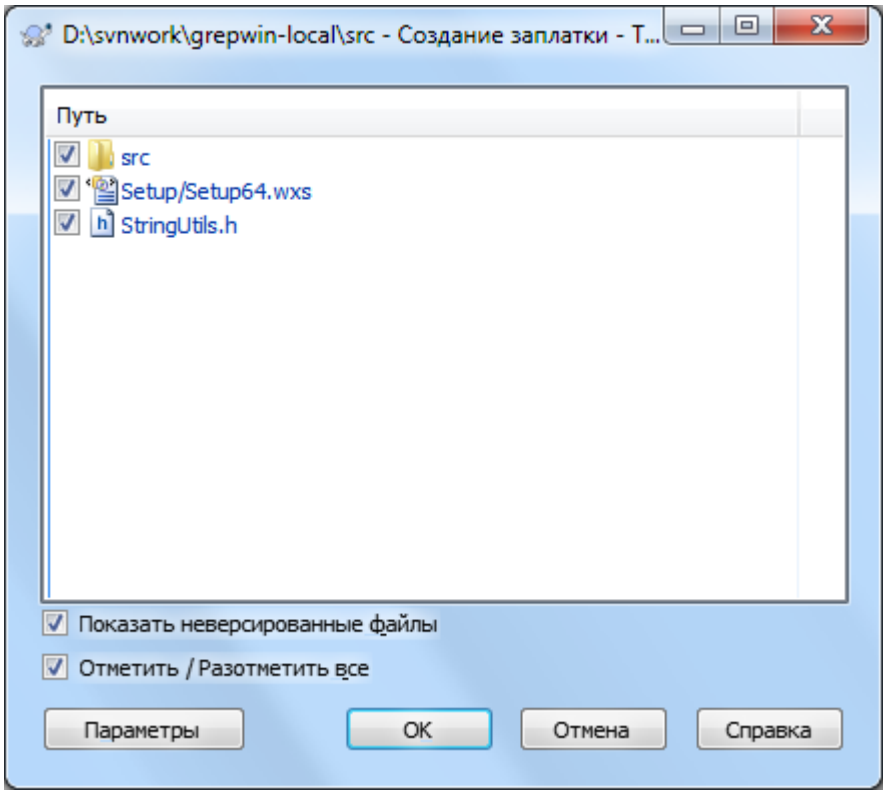


Рисунок 4.62. Диалог создания заплатки

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.

Нажимая на кнопку Параметры вы можете указать как создать заплатку. Например, вы можете указать, что изменения в окончаниях строк или пробельных символах не входят в окончательный файл заплатки.

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.

Подсказка

Не сохраняйте файл заплатки с расширением `.txt` если вы планируете его послать по e-mail кому-то другому. Обычные текстовые файлы часто повреждаются e-mail программами и часто случается, что пробельные символы и символы новых строк автоматически конвертируются и сжимаются. Если такое происходит, то заплатка не будет применена корректно. Поэтому используйте `.patch` или `.diff` в качестве расширения при сохранении файла 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.

Если вам нравится, то можете создать заплатку из диалога Фиксация или Проверить на наличие изменений. Просто выберите файлы и используя контекстное меню создайте заплатку из этих файлов. Если вы хотите увидеть диалог Параметры вы должны удерживать нажатой клавишу **shift** при правом клике.

4.23.2. 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. Имейте ввиду, эта опция появляется только если вы сохранили заплатку в буфер обмена при помощи TortoiseSVN → Создать заплатку.... Копирование заплатки в буфер обмена из другого приложения не приведёт к появлению кнопки.

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.

4.24. 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.

4.24.1. Авторство для файлов

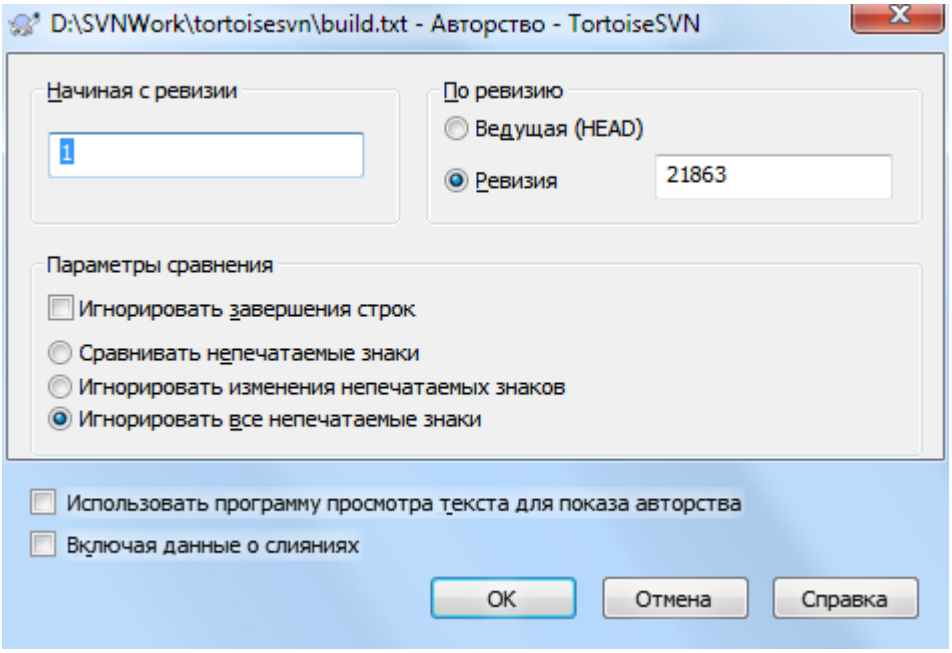


Рисунок 4.63. Диалог авторства/аннотирования

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 просматривается с помощью *TortoiseBlame*, который подсвечивает разные ревизии для более простого чтения. Если вы хотите напечатать или редактировать файл, то 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 4.11.2, «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.

При желании вы также можете добавить информацию о слияния, несмотря на то, что это может занять достаточно продолжительное время для получения с сервера. Когда строки объединены из другого источника информация об авторстве показывает ревизию изменений, которые были сделаны в оригинальном источнике, также когда они были объединены в этот файл.

После того, как вы нажмёте OK, TortoiseSVN начнёт извлекать данные для создания файла авторства. После того, как процесс получения информации об авторстве завершится, результат записывается во временный файл и вы можете его просмотреть.

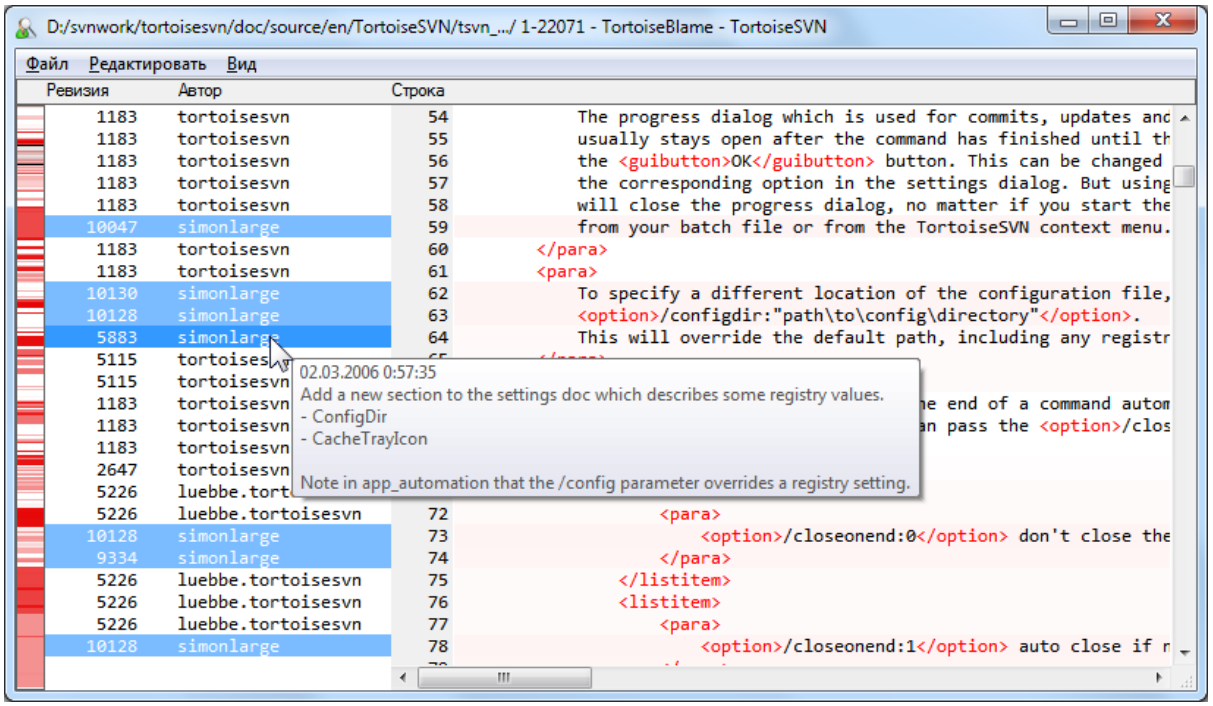


Рисунок 4.64. TortoiseBlame

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 изменяется 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.

Если вы используете отслеживание слияний и вы запросили информацию о слиянии когда запустили просмотр авторства, то объединенные строки показываются немного по-разному. Там где строка изменилась в результате объединения из другого пути TortoiseBlame покажет ревизию и автора последнего изменения в оригинальном файле вместо ревизии, в которой было слияние. Для обозначени таких строки ревизия и автор написаны курсивом. Ревизия, в которой было слияние, показывается отдельно во всплывающей подсказке при наведении курсора мыши на колонку с информацией об авторстве. Если вы не хотите, чтобы объединенные строки отображались таким образом, то выключите флажок Включая данные о слияниях при запуске просмотра авторства.

Если вы хотите видеть пути, участвовавшие в слиянии, то выберите View → Слитые пути. Это покажет путь, в котором строка была изменена последний раз, за исключением изменений в результате слияния.

Ревизия показанная в информации об авторстве предствляет последнюю ревизию, в которой содержимое этой строки было изменено. Если файл был создан путём копирования другого файла, то пока вы изменяете строку, её ревизия авторства будет показывать последнее изменение оригинального файла исходника, а не ревизию, где была копия. Это также применяется к путям, отображаемым в информации о слиянии. Путь показывает расположение в хранилище, где было сделано последнее изменение этой строки.

До настроек TortoiseBlame можно добраться, используя TortoiseSVN → Настройки... на вкладке TortoiseBlame. Подробнее об этом рассказывает Раздел 4.31.9, «Настройки TortoiseBlame».

4.24.2. Авторство различий

Одно из ограничений отчета об авторстве то, что он показывает только файл в определенной ревизии и последнего автора изменившего каждую строку. Иногда вы хотите знать какие изменения были сделаны, а также кто их сделал. Если вы сделаете правый клик на строке в TortoiseBlame, то получите контекстное меню для показа изменений, сделанных в этой ревизии. Но если вы хотите увидеть изменения *и* информацию об автостве одновременно, то вам нужна комбинация отчета о различиях и авторстве.

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.

4.25. 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.

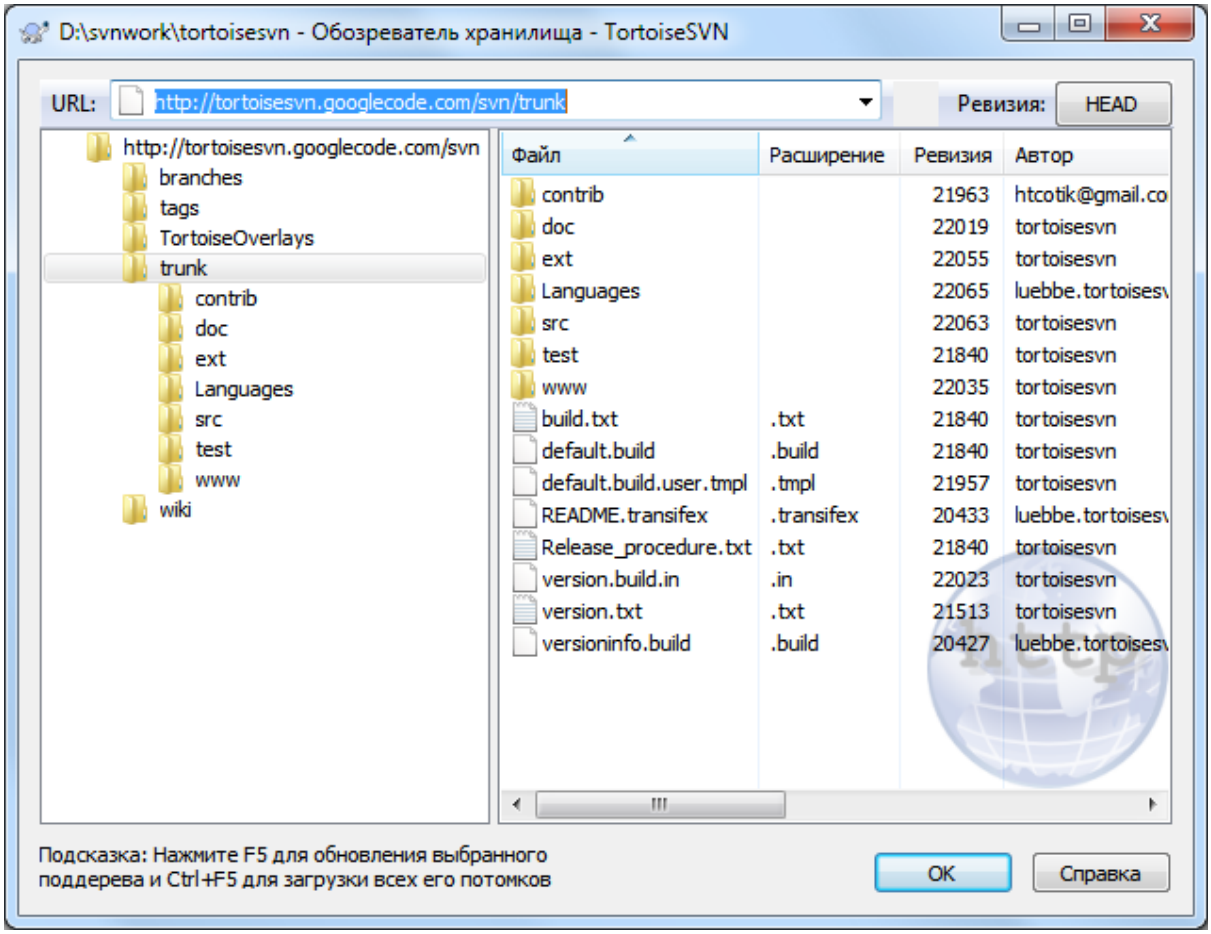


Рисунок 4.65. Обозреватель хранилища

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.

Папки добавленные с помощью свойства `svn:externals` также показаны в обозревателе хранилища. Такие папки отображаются с маленькой стрелкой, которая обозначает, что они не часть структуры хранилища, а всего лишь ссылки.

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.

- Редактировать выбранный файл. Это извлекает временную рабочую копию и запускает редактор по умолчанию для этого типа файла. Когда вы закрываете редактор и сохраняете изменения, тогда появляется диалог фиксации, и вы можете ввести комментарий и зафиксировать изменения.
- 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.
- Извлечь отдельный файл. Это создает «неполную» рабочую копию, которая содержит только один этот файл.
- Удалить или переименовать файл.
- Сохранить неверсированную копию файла на жёсткий диск.
- Скопировать адрес URL из адресной строки в буфер обмена.
- Скопировать файл, либо в другую часть хранилища, либо в рабочую копию, базирующуюся в том же хранилище.
- Посмотреть/отредактировать свойства файла.
- Создать ярлык чтобы вы могли быстро запустить обозреватель хранилища открытый прямо с этого места.

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 неверсированные файлы и папки прямо в хранилище. Фактически это операция Subversion импорт.
- Удалить или переименовать папку.
- Скопировать папку, либо в другую часть хранилища, либо в рабочую копию, базирующуюся в том же хранилище. Это также может быть использовано для создания ответвления/метки, при котором нет необходимости в наличии извлечённой рабочей копии.
- 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.

Внешние элементы (используемые `svn:externals`) также показываются в обозревателе хранилища, и вы даже можете опуститься внутрь содержимого папки. Внешние элементы отмечены красной стрелкой над ними.

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**.

Добавленные в закладки адреса URL/хранилища показаны ниже папок текущего хранилища в левом дереве просмотра. Вы можете добавить туда пункты сделав правый щелчок мышью на любом файле или папке и выбрав Контекстное меню → Добавить в закладки. Правый щелчок на закладке приведёт к просмотру этого хранилища и файла/папки.

4.26. Графы ревизий

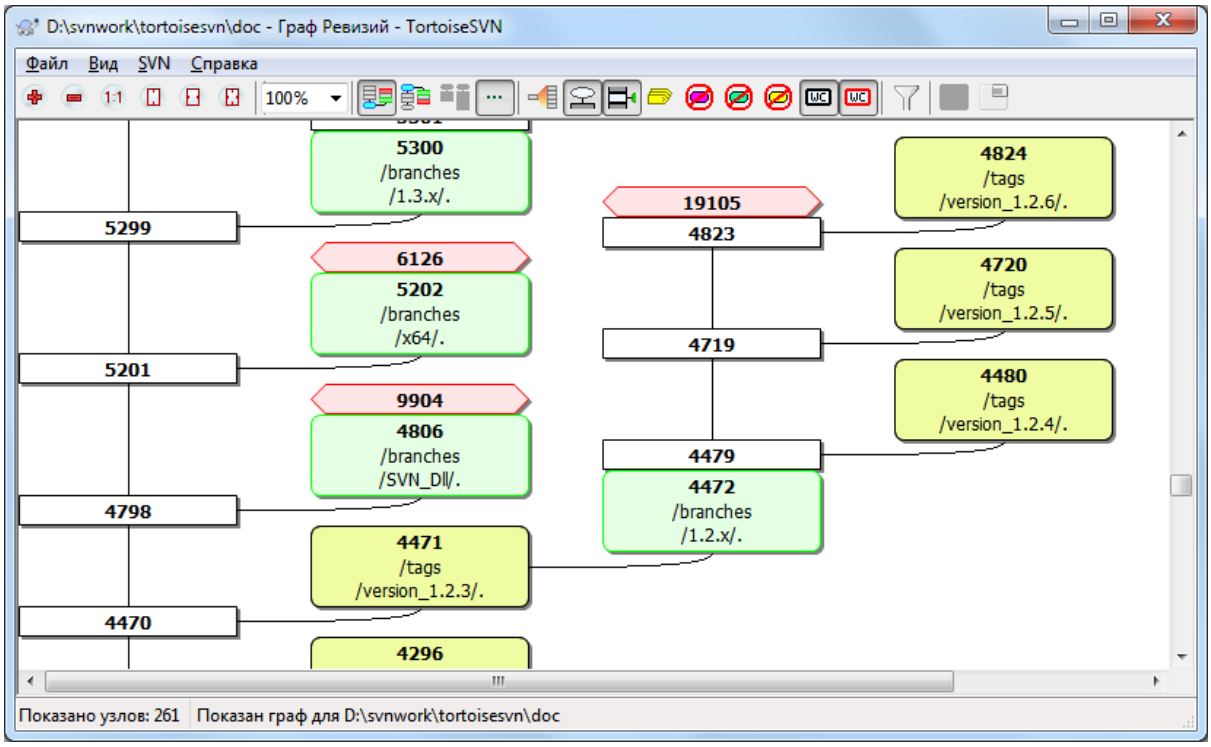


Рисунок 4.66. Граф ревизий

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.

4.26.1. 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 путем копирования другого файла/папки показываются в виде закругленного прямоугольника. Цвет по умолчанию зеленый. Метки и стволы (trunks) рассматриваются как особый случай и используют различные тени в зависимости от TortoiseSVN → Настройки.

Удалённые узлы
: Удаленные элементы, например, больше ненужные ответвления, показываются в виде восьмиугольника (прямоугольник со срезанными углами). По умолчанию красного цвета.

Переименованные узлы
: Переименованные 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.

4.26.2. 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 веткам, так что нет глобального порядка ревизий: последовательные ревизии в ветке будут показаны (часто) в упорядоченных строках. Вложенные ветки, тем не менее, упорядочены таким образом, что поздние ветки будут показаны в той же колонке над ранними ветками чтобы сохранить граф стройным. В результате данная строка может содержать изменения из разных 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 наоборот, сверху вниз.

Снизить число пересечений
: Эта настройка обычно включена и предотвращает граф от показа со многими пересекающимися линиями. Тем не менее, это может сделать размещение колонок менее логичным, например, по диагонали нежели в колонку, и графу потребуется больше места для отрисовки. Если это для вас проблема, то это можно отключить в меню Вид.

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
: Когда в проекте много меток, то отображение каждой метки как отдельный узел в графе занимает много места и запутывает более интересную структуру веток разработки. В то же время вам может понадобится просто найти содержание метки, чтобы сравнить ревизии. Эта настройка прячет узлы для меток и вместо этого показывает их во всплывающих подсказках для узла откуда они были скопированы. Значок метки на правой стороне исходного узла обозначет что были сделаны метки. Это очень упрощает обзор.

  Обратите внимание, что если сама метка использовалась как источник для копии, возможно, новая ветка на основе метки, то такая метка будет показана отдельным узлом, а не свернутым.

Скрыть удалённые пути
: Скрывает пути, которых больше нет в ведущей ревизии хранилища, например, удалённые ответвления.

  Если вы выбрали настройку Свернуть метки и затем удаленная ветка, от которой была созданы метки, всё ещё показана, иначе метки исчезнут тоже. Последняя ревизия, для которой была создана метка, будет показана цветом используемым для удаленных узлов, вместо отображения отдельной ревизии удаления.

  Если вы выбрали настройку Спрятать метки, эти ветки исчезнут опять, т. к. им не нужно показывать метки.

Спрятать неиспользованные ветки
: Скрывает ответвления, в которых не было зафиксировано изменений в соответствующем 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.

  Если вы скрываете определенный путь и этот узел имеет узлы-потомки, потомки будут показаны отдельным деревом. Если вы хотите также скрыть всех потомков, то используйте флажок Убирать поддеревья полностью.

Деревья в полосах
: Когда граф содержит несколько 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 бесполезным из-за чрезмерной степени увеличения и поэтому в этом случае оно показано не будет.

4.26.3. 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 которой 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.

4.26.4. 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.

4.26.5. Подрезка деревьев

В большом дереве может быть трудно ориентироваться, 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, контекстное меню не показывается.

4.27. Экспорт рабочей копии Subversion

Иногда вам может понадобится чистая копия вашего рабочего дерева без папки `.svn`, например, для создания файла архива ваших исходных кодов, или для экспорта на веб-сервер. Вместо того, чтобы выполнить копирование и затем вручную удалять папку `.svn`, TortoiseSVN предлагает команду TortoiseSVN → Экспорт.... Экспорт из источника, заданного адресом URL, и экспорт из рабочей копии, обрабатывается немного по-разному.

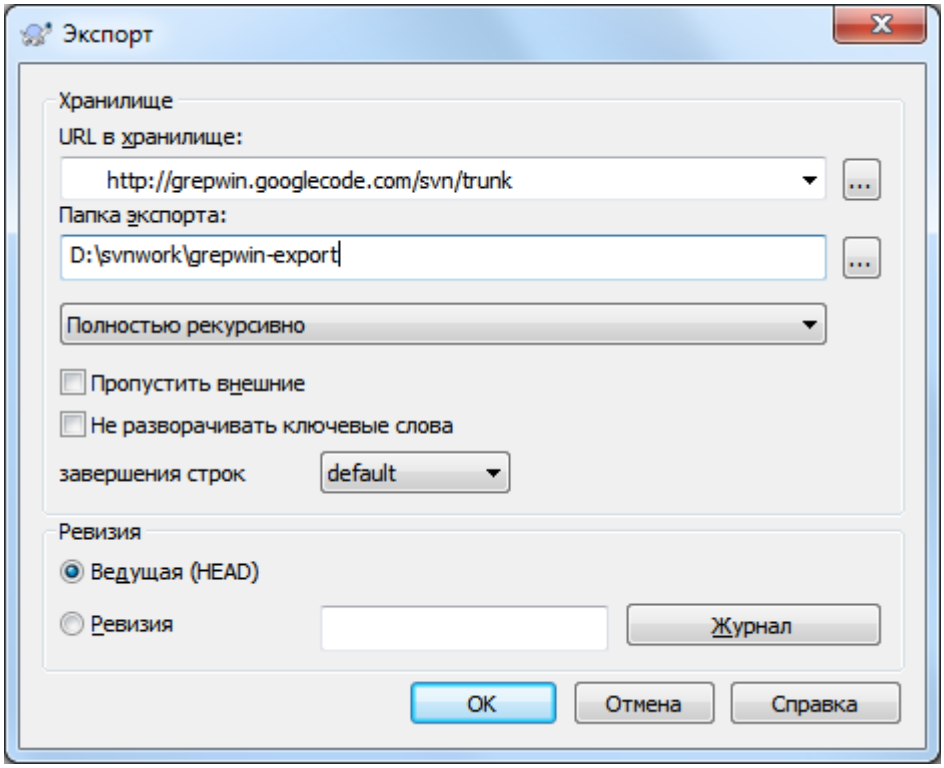


Рисунок 4.67. Диалог Экспорт-из-URL

Если вы выполняете эту команду на неверсированной папке, TortoiseSVN 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 переопределить 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 папки `.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 другое место и выбора Контекстное меню → SVN Экспортировать версированные элементы сюда или Контекстное меню → SVN Экспортировать все элементы сюда или Контекстное меню → SVN Экспортировать изменённые элементы сюда. Второй пункт включает также и неверсированные файлы. Третий пункт экспортирует только изменённые файлы, но сохраняет структуру папок.

При экспорте из рабочей копии если в папке назначения уже есть папка с таким же названием, то вам предоставляется выбор перезаписать существующее содержимое или создать новую папку с автоматически сгенерированным названием, например, `Target (1)`.

Экспортирование отдельных файлов

Диалог экспорта не позволяет экспортировать отдельные файлы, несмотря на то, что Subversion это может.

Для экспорта отдельных файлов в TortoiseSVN вы должны использовать обозреватель хранилища (Раздел 4.25, «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 4.11.3, «Сравнение папок».

Если вы хотите экспортировать дерево структуры вашей рабочей копии, но содержащую только локально изменённые файлы, то см. выше SVN Экспортировать изменённые элементы сюда.

4.27.1. Выведение рабочей копии из-под управления версиями

Иногда возникает задача преобразовать рабочую копию обратно в нормальную папку без директории `.svn`. Всё что вам надо сделать — это удалить директорию `.svn` в корне рабочей копии.

Как альтернатива вы можете экспортировать папку саму в себя. В Проводнике Windows перетащите с помощью правой кнопки мыши корневую папку рабочей копии из файловой панели в саму себя на панели папок. TortoiseSVN обнаруживает такие особые случаи и спрашивает вас хотители вы сделать рабочую копию неверсированной. При ответе *да* управляющая папка будут удалена, и у вас будет простое неверсированное дерево папок.

4.28. Перебазирование рабочей копии

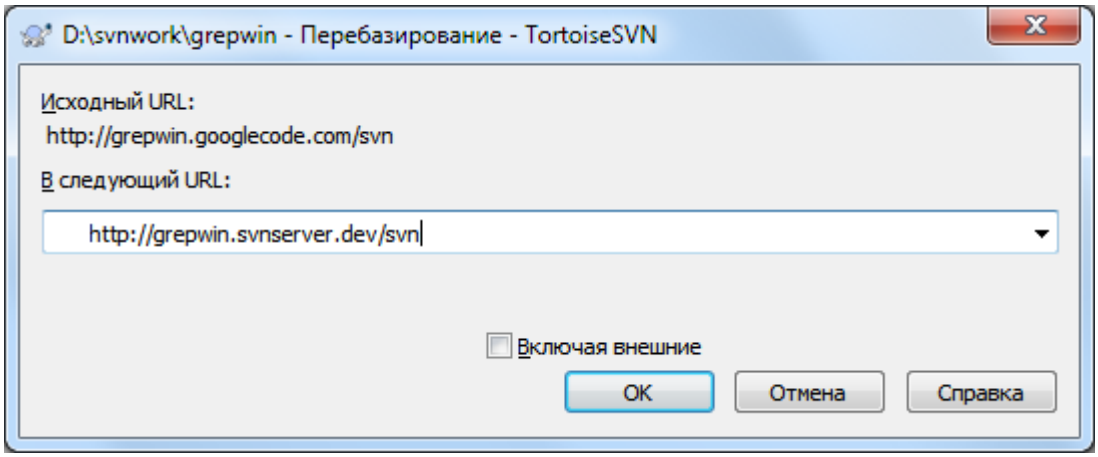


Рисунок 4.68. Диалог перебазирования

Если у вашего хранилища по каким-то причинам изменилось размещение (IP/URL). Возможно у вас застопорилась работа и не можете пока фиксировать, и при этом не хотите извлекать рабочую копию заново из нового размещения и затем перемещать все изменения в новую рабочую копию, команда TortoiseSVN → Перебазировать то что вам нужно. Она делает очень простую вещь: переписывает все адреса URL для каждого файла и папки новым адресом URL.

Примечание

Эта операция работает только в *корне* рабочей копии. Так что пункт в контекстном меню показывается только в корнях рабочих копий.

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.

Предупреждение

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 4.20.3, «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.

4.29. 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.

4.29.1. 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 4.18, «Установки проекта»).

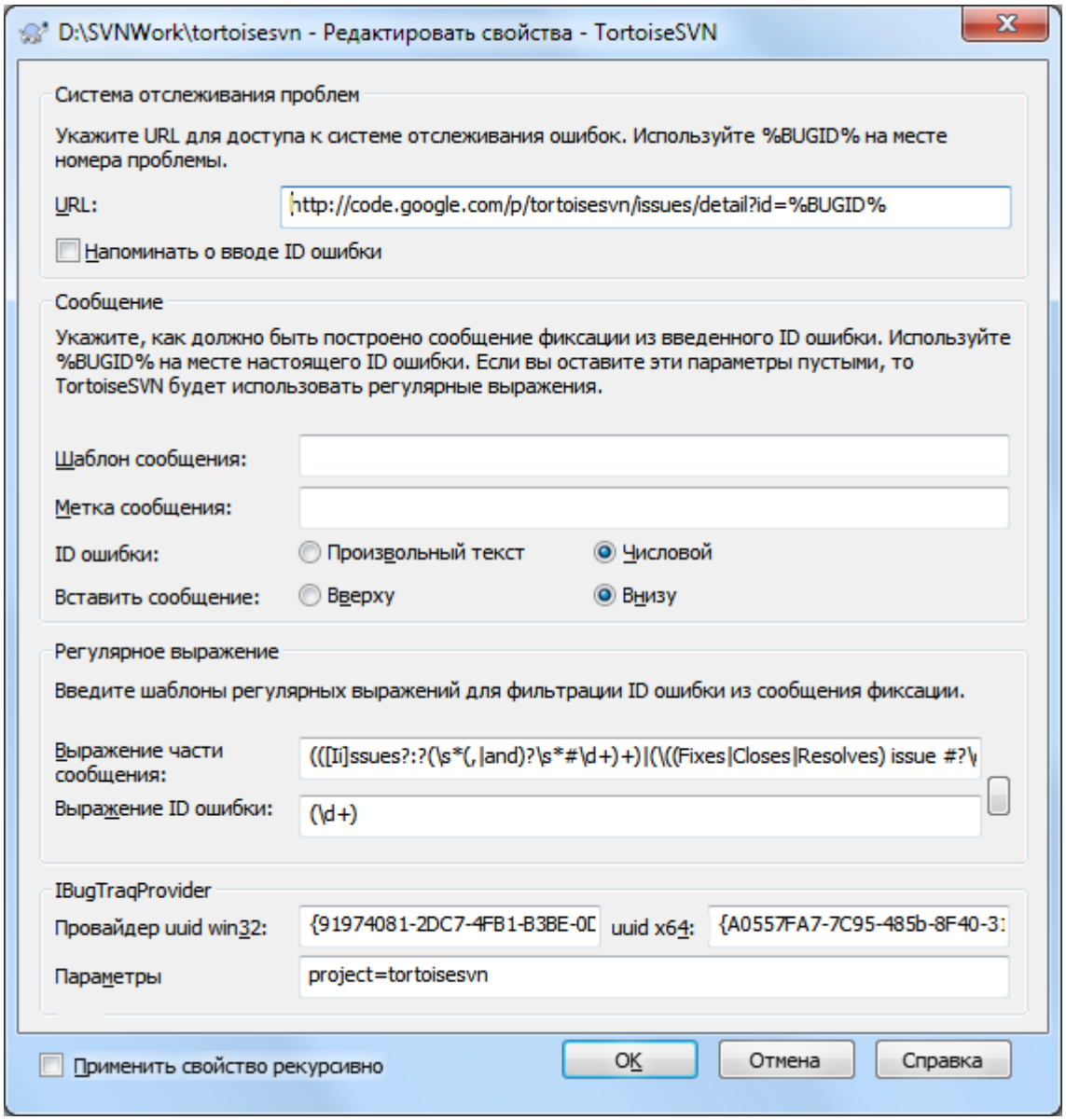


Рисунок 4.69. Диалоговое окно свойств Bugtraq

Когда вы редактируете любое из свойств bugtraq используется специальный редактор для упрощения установки подходящих значений.

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
: Напишите в этом свойстве URL адрес вашего инструмента отслеживания ошибок. Это должен быть правильно сформированный URI и должен содержать `%BUGID%`. `%BUGID%` заменяется номером проблемы, который вы ввели. Это позволяет TortoiseSVN показать ссылку в диалоге журнала, и когда вы смотрите на ревизию журнала сразу можете перейти в инструмент отслеживания ошибок. Вы не обязаны задавать это свойство, но тогда TortoiseSVN покажет только номер проблемы без ссылки на неё. Например, проект TortosieSVN использует `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`.*

4.29.1.1. 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: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`.*

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.*

4.29.1.2. 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 два выражения установлены, то первое выражение используется как предварительный фильтр при поиске выражения, которое содержит ID ошибки. Второе выражение затем извлекает ID ошибки из результатов первого регулярного выражения. Это позволяет вам использовать список ID ошибок и выражения простым языком. Например, вы можете исправить несколько ошибок и строку наподобии этой: «This change resolves issues #23, #24 and #25».

Если вы хотите отловить ID ошибки как это сделано выше в сообщении журнала, то вы можете использовать следующее регулярное выражение, оно же используется в проекте TortoiseSVN: `[Ii]ssues?:?(\s*(,|and)?\s*#\d+)+` and `(\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.

If you are unfamiliar with regular expressions, take a look at the introduction at https://en.wikipedia.org/wiki/Regular_expression, and the online documentation and tutorial at http://www.regular-expressions.info/.

Не всегда просто получить правильное регулярное выражение, поэтому в диалог свойства bugtraq встроен тестовый диалог. Нажмите на кнопке справа от окна редактирования для вызова. Здесь вы можете ввести некоторый тестовый текст, изменять каждое регулярное выражение и видеть результат. Если регулярное выражение неверно, то цвет фона окна редактирования меняется на красный.

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

Для того чтобы система работата, эти свойства должны быть установлены на папках. Когда вы фиксируете файл или папку свойства читаются из этой папки. Если там свойства не найдены, то TortoiseSVN будет искать вверх по дереву папок пока не дойдет до неверсированной папки или корня дерева (например, `C:\`). Если вы можете быть уверены, что каждый пользователь извлекает только из, допустим, `trunk/`, а не какой-то подпапки, тогда достаточно установить

свойства для `trunk/`. Если вы не можете быть в этом уверены, то вы должны установить свойства рекурсивно для каждой подпапки. Настройки свойства глубже в иерархии проекта перекрывают настройки верхних уровней (ближе к `trunk/`).

Начиная с версии 1.8 TortoiseSVN и Subversion используют так называемые `наследуемые свойства`, которое означает, что свойство, заданное для папки, автоматически и неявно устанавливается для всех подпапок. Так что больше нет необходимости задавать свойства для всех папок, а только для корневой.

Только для свойств проекта, т.е. для `tsvn:`, `bugtraq:` и `webviewer:`, вы можете использовать флажок Рекурсивно для установки свойства для всех подпапок в иерархии, без установки его также для всех файлов.

Если вы добавляете новые подпапки в рабочую копию используя TortoiseSVN, то любые свойства проекта, заданные в родительской папке, будут автоматически добавлены в новые дочерние папки.

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 возможности в обозревателе хранилища, если только не запустите его из вашей рабочей копии. Если вы запустили обозреватель хранилища и ввели URL хранилища, эта возможность будет недоступна.

По этой же причине свойства проекта не будут автоматически скопированы, если дочерняя папка добавляется с помощью обозревателя хранилища.

Интеграция с системой отслеживания ошибок не ограничена только TortoiseSVN, она может быть использована с любым клиентом Subversion. Для дополнительной информации прочтите *Issue Tracker Integration Specification* [https://svn.code.sf.net/p/tortoisesvn/code/trunk/doc/notes/issuetrackers.txt] в хранилище исходного кода TortoiseSVN. (Раздел 3, «Лицензия» объясняет как получить доступ к хранилищу.)

4.29.2. 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.

Конечно, любой такой интерфейс будет очень специфичным для Вашей системы отслеживания проблем, поэтому мы не можем предоставить эту часть, и описание создания такой программы выходит за рамки данного руководства. Определение интерфейса и примеры подключаемых модулей на C# и C++/ATL Вы можете получить в папке `contrib` в *TortoiseSVN repository* [https://svn.code.sf.net/p/tortoisesvn/code/trunk/contrib/issue-tracker-plugins]. (Раздел 3, «Лицензия» объясняет, как получить доступ к хранилищу.) Также описание API дано в Глава 7, *Интерфейс IBugtraqProvider*. Другой (рабочий) пример плагина на C# есть в *Gurtle* [http://code.google.com/p/gurtle/], который реализует требуемый 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.

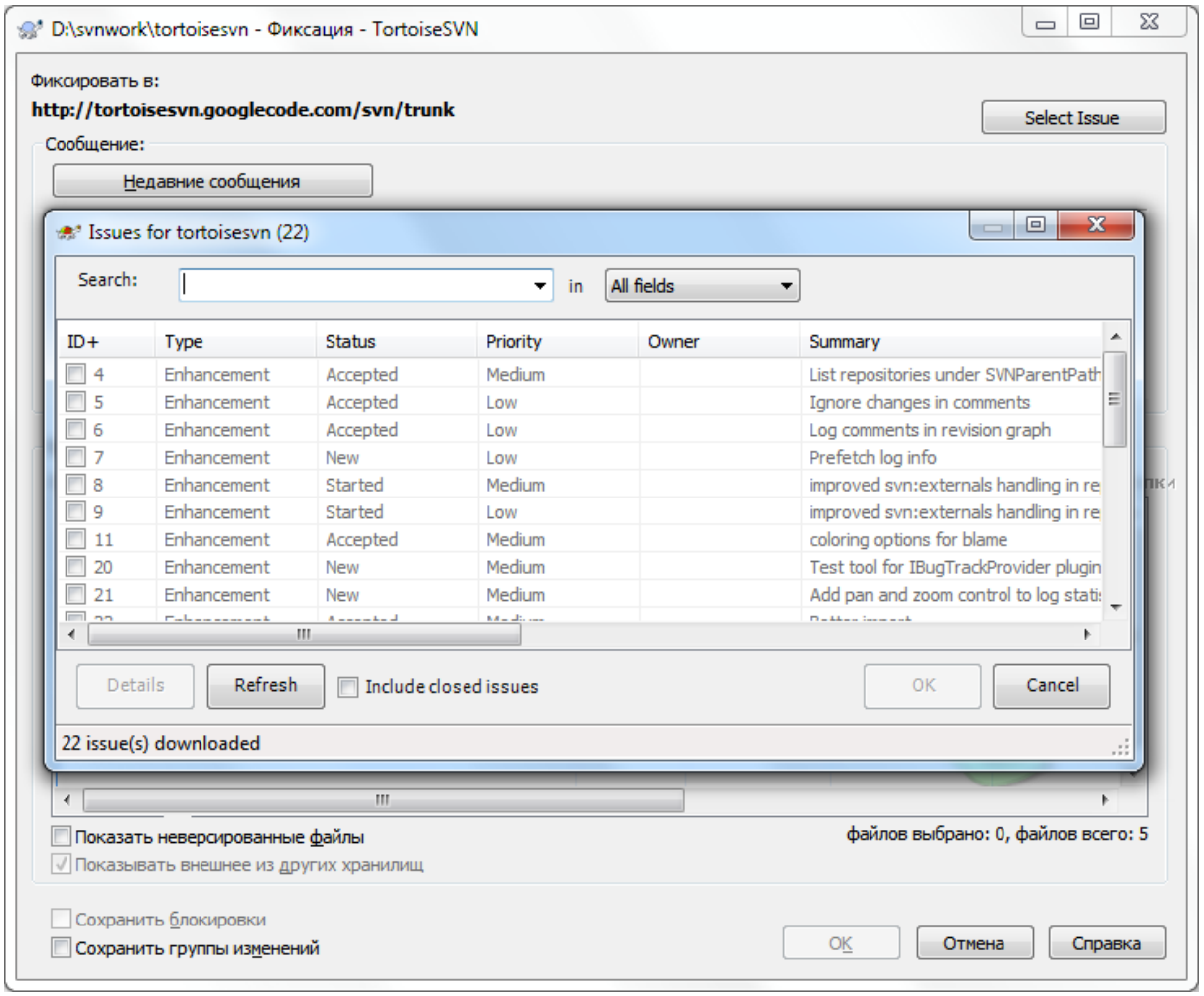


Рисунок 4.70. Пример диалога запроса системы отслеживания проблем

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.

4.30. 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 4.18, «Установки проекта»)

webviewer:revision
: Установите в этом свойстве адрес URL вашего просмотрщика хранилища для просмотра всех изменений в определенной ревизии. Это должен быть правильный URI и должен содержать `%REVISION%`. `%REVISION%` заменяется в вопросе номером ревизии. Это разрешает TortoiseSVN показать в контекстном меню журнала пункт Context Menu → Посмотреть ревизию в веб-обозревателе.

webviewer:pathrevision
: Установите в этом свойстве адрес URL вашего просмотрщика хранилища для просмотра изменений в определенном файле в определенной ревизии. Это должен быть правильный URI и должен содержать

`%REVISION%` и `%PATH%`. `%PATH%` заменяется относительным путём к корню хранилища. Это разрешает TortoiseSVN показать в контекстном меню журнала пункт Context Menu → Посмотреть ревизию для пути в веб-обозревателе. Например, если вы сделаете правый клик в нижней панели диалога журнала на файле `/trunk/src/file`, то `%PATH%` в адресе URL будет заменён 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 4.29, «Интеграция с системами отслеживания ошибок/проблем».

Устанавливайте свойства на папках

Для того чтобы система работата, эти свойства должны быть установлены на папках. Когда вы фиксируете файл или папку свойства читаются из этой папки. Если там свойства не найдены, то TortoiseSVN будет искать вверх по дереву папок пока не дойдет до неверсированной папки или корня дерева (например, `C:\`). Если вы можете быть уверены, что каждый пользователь извлекает только из, допустим, `trunk/`, а не какой-то подпапки, тогда достаточно установить свойства для `trunk/`. Если вы не можете быть в этом уверены, то вы должны установить свойства рекурсивно для каждой подпапки. Настройки свойства глубже в иерархии проекта перекрывают настройки верхних уровней (ближе к `trunk/`).

Только для свойств проекта, т.е. для `tsvn:`, `bugtraq:` и `webviewer:`, вы можете использовать флажок Рекурсивно для установки свойства для всех подпапок в иерархии, без установки его также для всех файлов.

Если вы добавляете новые подпапки в рабочую копию используя TortoiseSVN, то любые свойства проекта, заданные в родительской папке, будут автоматически добавлены в новые дочерние папки.

Ограничения по использованию обозревателя 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 возможности в обозревателе хранилища, если только не запустите его из вашей рабочей копии. Если вы запустили обозреватель хранилища и ввели URL хранилища, эта возможность будет недоступна.

По этой же причине свойства проекта не будут автоматически скопированы, если дочерняя папка добавляется с помощью обозревателя хранилища.

4.31. Настройки TortoiseSVN

Чтобы узнать, для чего предназначены различные параметры, просто оставьте ненадолго указатель мыши на поле ввода/флажке... и появится полезная подсказка.

4.31.1. Общие настройки

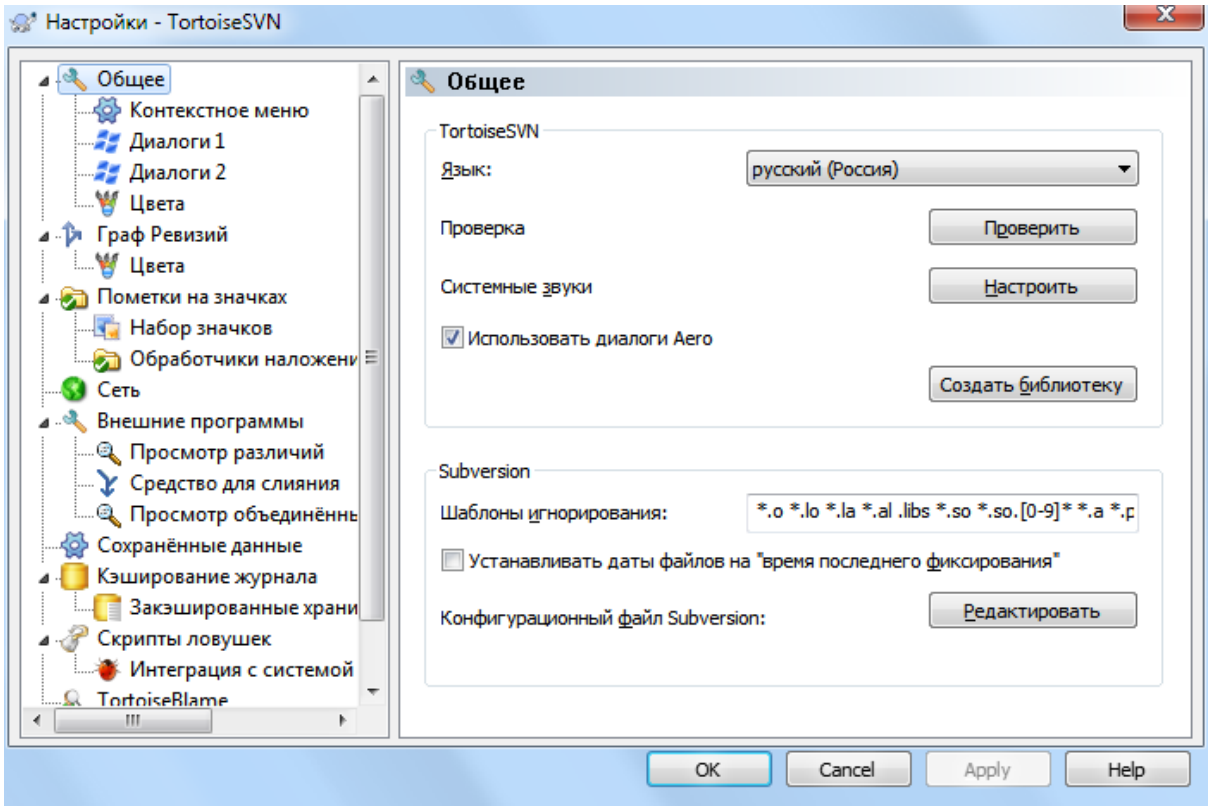


Рисунок 4.71. Страница 'Общее' в диалоге настроек

В этом диалоге можно указать предпочитаемый вами язык интерфейса, а также некоторые специальные настройки Subversion.

Язык
: Выбирает язык пользовательского интерфейса. Конечно, вы должны сначала установить соответствующий языковой пакет, чтобы получить в интерфейсе язык отличный от английского.

Проверить наличие обновлений
: TortoiseSVN будет периодически соединяться с сайтом и проверять доступна ли новая версия программы. Если доступна, то будет показана уведомляющая ссылка в диалоге фиксации. Используйте кнопку Проверить чтобы получить ответ прямо сейчас. Новая версия не будет загружена, вы просто получите информационный диалог, сообщающий что доступна 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.

Использовать Aero диалоги
: В ОС Windows Vista и более поздних системах управляет стилем Aero для диалоговых окон.

Создать библиотеку
: В ОС Windows 7 вы можете создать Библиотеку, в которой сгруппировать рабочие копии, которые разбросаны в различных местах системы.

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 4.14.1, «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.

Внимание

Если вы воспользуетесь файлом настроек Subversion для установки шаблона `global-ignores`, то его значение переопределит сделанные здесь установки. Доступ к файлу настроек Subversion можно получить посредством кнопки Правка, описанной ниже.

Этот шаблон игнорирования повлияет на все ваши проекты. Он не версируется, поэтому не затронет других пользователей. Для сравнения, вы также можете использовать версированное свойство `svn:ignore` или `svn:global-ignores` для исключения файлов или папок из-под управления версиями. Для более подробной информации прочтите Раздел 4.14, «Игнорирование файлов и папок».

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/1.8/svn.advanced.confarea.html]. Раздел *Автоматическая установка свойств (Automatic Property Setting)* [http://svnbook.red-bean.com/en/1.8/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/1.8/svn.advanced.confarea.html#svn.advanced.confarea.windows-registry], чтобы узнать больше.

Применить локальные изменения в svn:externals при обновлении

Эта настройка говорит TortoiseSVN всегда применять локальные изменения к свойству `svn:externals` при обновлении рабочей копии.

4.31.1.1. Настройки контекстного меню

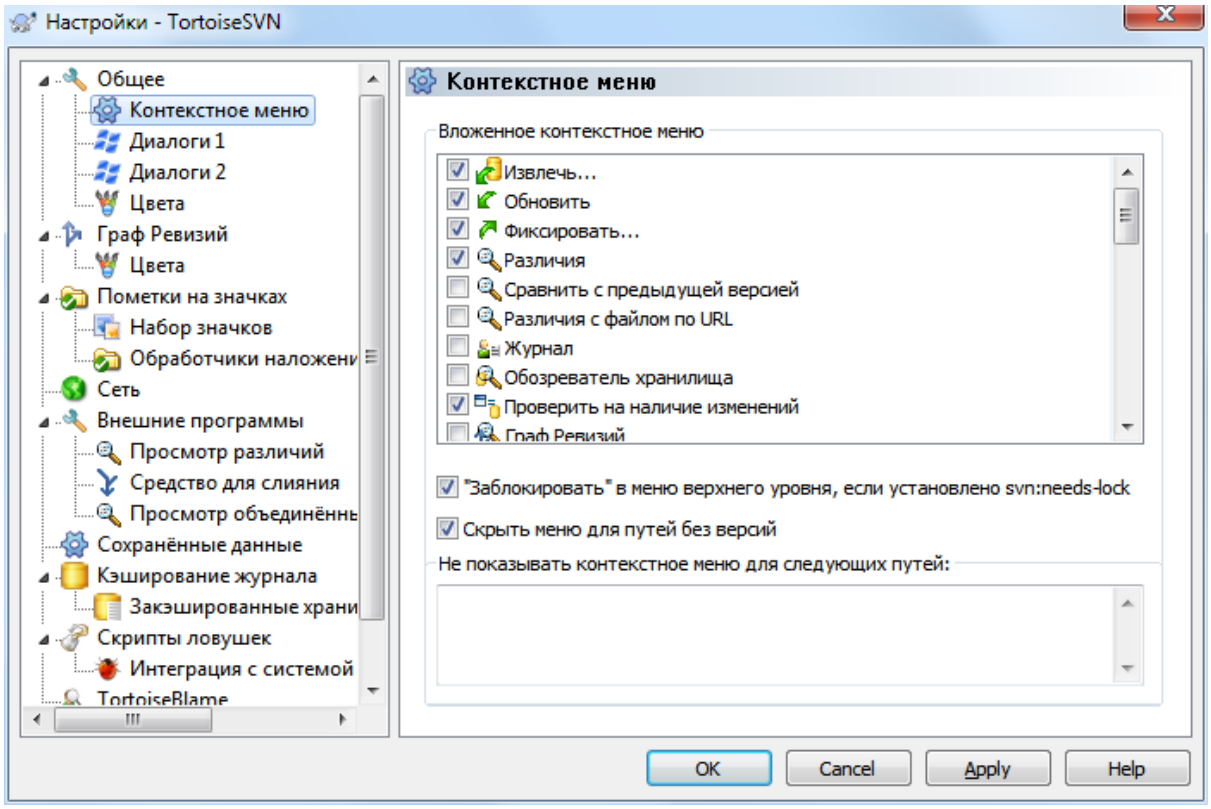


Рисунок 4.72. Страница контекстного меню в диалоге настроек

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.

Большую часть времени вам не понадобится контекстное меню TortoiseSVN, в отличии от папок, которые находятся по управлением Subversion. Для неверсированных папок контекстное меню нужно вам если вы хотите извлечь рабочую копию. Если вы включите настройку `Скрыть меню для неверсированных путей` , то TortoiseSVN не будет добавлять пункты в контекстное меню для неверсированных папок. Но пункты добавляются для всех элементов и путей в версированной папке. И вы можете вернуть пункты для неверсированных папок нажав и удерживая клавишу **Shift** при показе контекстного меню.

Если вы желаете, чтобы контекстное меню TortoiseSVN по некоторым путям в вашем компьютере не показывалось вообще, вы можете указать такие пути в поле снизу.

4.31.1.2. Настройки диалогов TortoiseSVN - 1

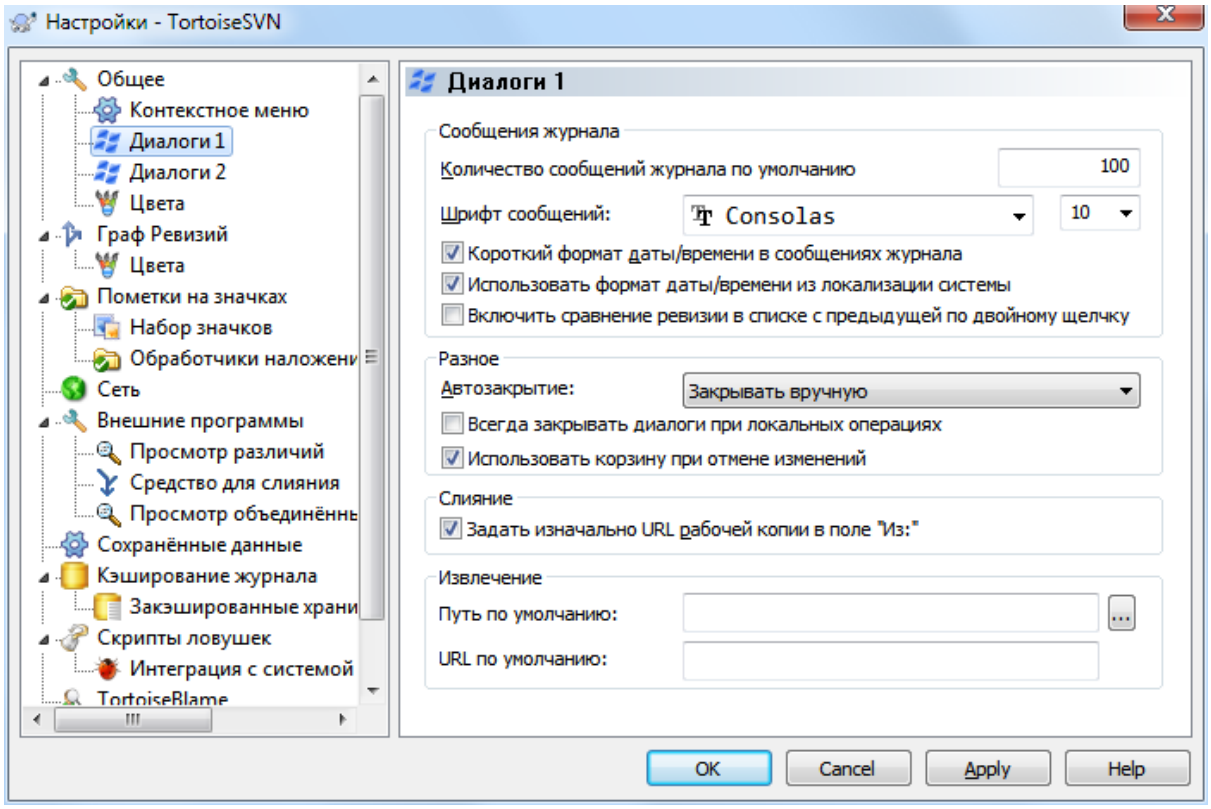


Рисунок 4.73. Страница 'Диалоги 1' в диалоге настроек

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[3]. 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 обнаружили, что часто сравниваете ревизии в верхней панели диалога журнала, то можете использовать эту настройку для разрешения этого действия по двойному щелчку. По умолчанию она не включена, т. к. получение различий часто долгий процесс и многие люди предпочитают избегать ожидания при случайном двойном щелчке. Вот поэтому по умолчанию настройка отключена.

Авто-закрытие
: 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

[3] Да и для ограничения трафика тоже - прим. переводчика

просмотреть все сообщения и проверить, что произошло. Однако, вы 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 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.

Всегда закрывать диалоговые окна для локальных операций

Локальные операции, такие как добавление файлов или отмена изменений, не требуют соединения с хранилищем и выполняются быстро, так что диалог процеса выполнения малоинтересен. Выберите эту настройку если вы хотите, чтобы диалог процеса выполнения закрывался автоматически, кроме случаев возникновения ошибок.

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.

4.31.1.3. Настройки диалогов TortoiseSVN - 2

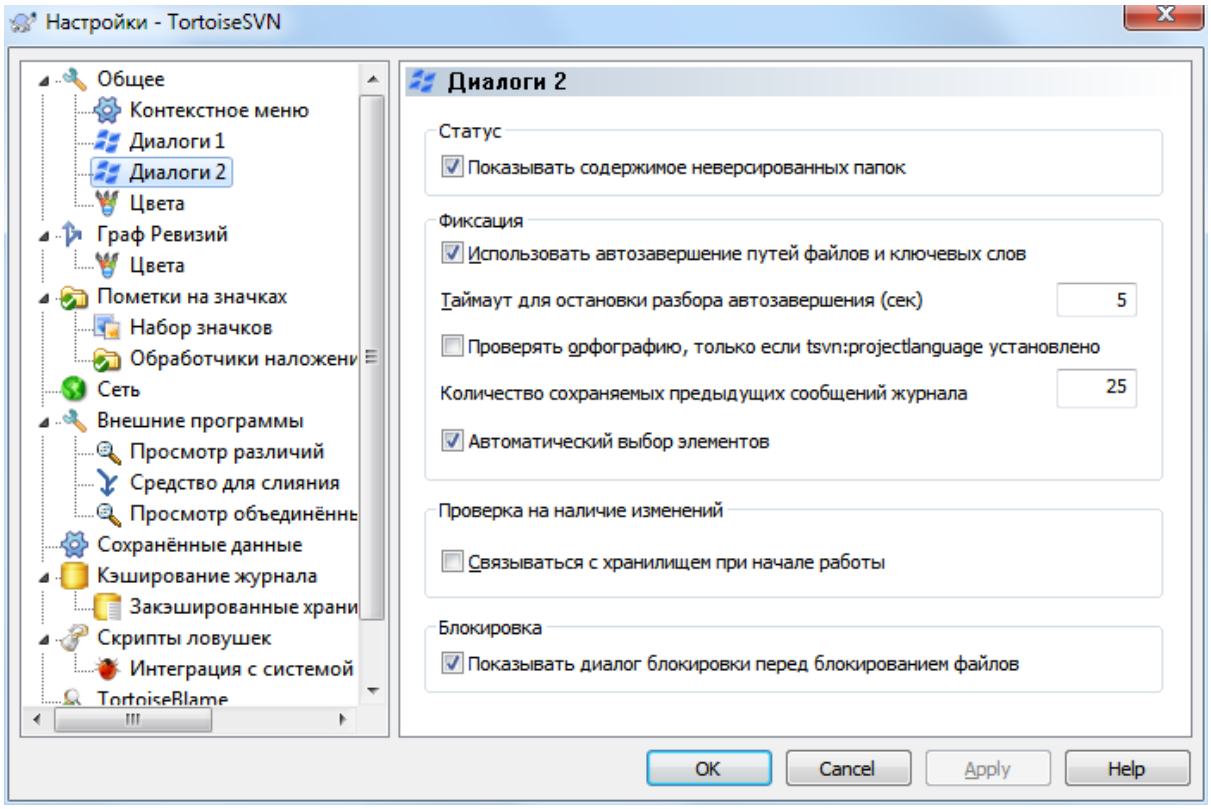


Рисунок 4.74. Страница 'Диалоги 2' в диалоге настроек

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.

В диалоге Проверка на наличие изменений вы можете включить отображение игнорируемых элементов. Если этот флажок отмечен, то при обнаружении игнорируемой папки также будут показаны все её дочерние элементы.

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.

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.

Открыть диалог после фиксации если есть незафиксированные элементы

Это автоматически повторно откроет диалог фиксации в той же директории после успешной фиксации. Диалог открывается только если остались ещё элементы для фиксации.

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.

4.31.1.4. Настройки диалогов TortoiseSVN - 3

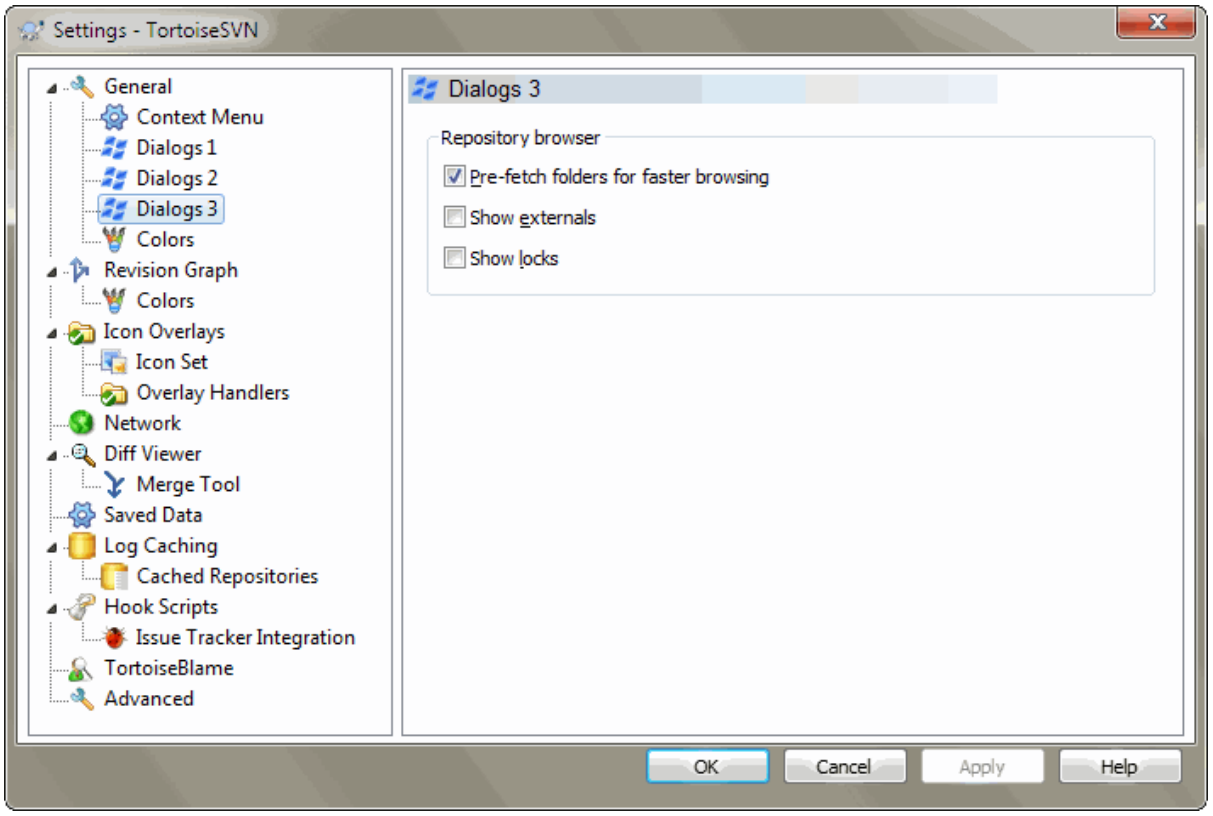


Рисунок 4.75. Страница 'Диалоги 3' в диалоге настроек

Предварительно выбирать папки для быстрого просмотра
: Если этот флажок установлен (значение по умолчанию), то обозреватель хранилища получает информацию о показанных папках в фоновом режиме. Таким образом, как только вы просматриваете одну из этих папок, то информация уже доступна.

  Однако некоторые серверы не могут обрабатывать множество запросов по этой причине, или же при неправильных настройках такое количество запросов воспринимается как что-то неправильное и начинает их блокировать. В этом случае вы можете отключить предварительную выборку.

Показать внешние
: Если этот флажок установлен (значение по умолчанию), то обозреватель хранилища показывает файлы и папки, которые включены с помощью свойства `svn:externals`, как обычные файлы и папки, но с оверлейным значком для обозначения того, что они из внешнего источника.

  Возможность предварительной выборки, описанная выше, может создать слишком большую нагрузку на слабые серверы. В этом случае вы можете здесь отключить эту возможность.

4.31.1.5. Настройки цветов в TortoiseSVN

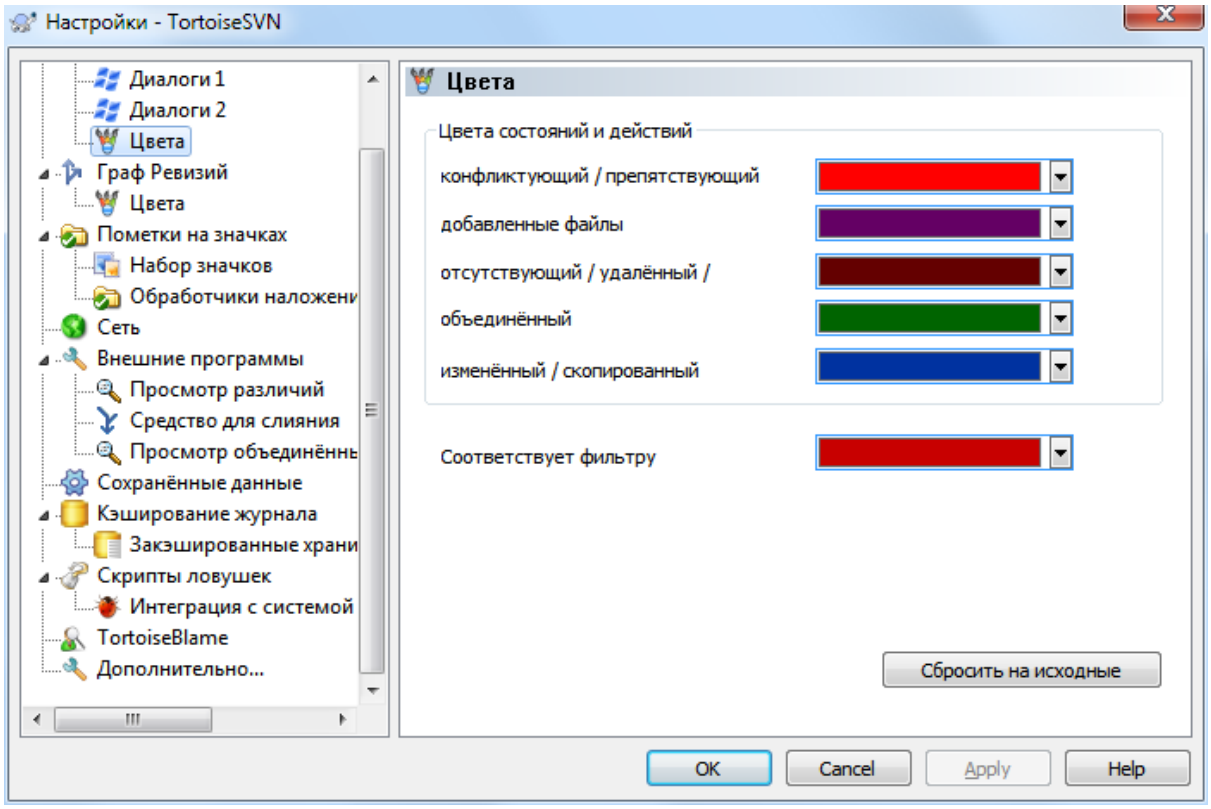


Рисунок 4.76. Страница 'Цвета' в диалоге настроек

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.

Соответствие фильтру
: При использовании фильтра в диалоге журнала искомые условия подсвечены этим цветом в результате.

4.31.2. Настройки графа ревизий

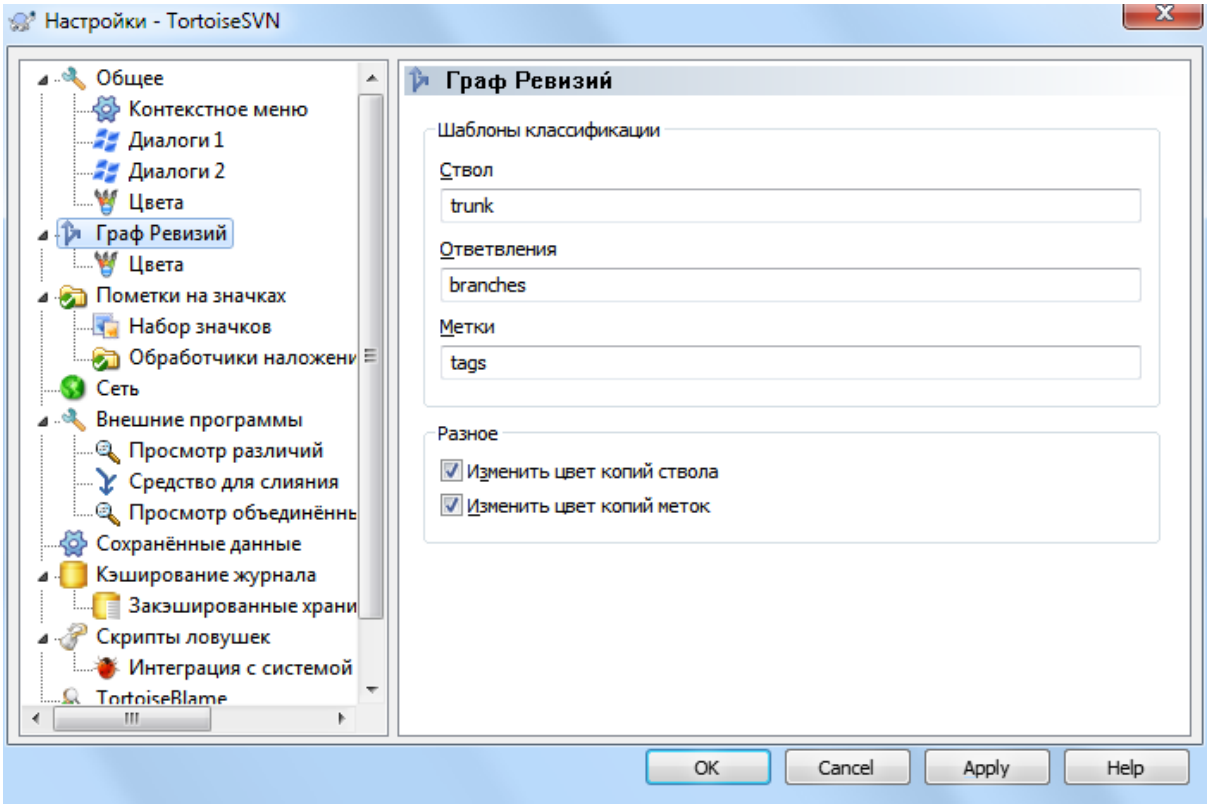


Рисунок 4.77. Страница 'Граф ревизий' в диалоге настроек

Шаблоны классификации
: Граф ревизий пытается показать более ясную картину структуры вашего хранилища, выделяя ствол, ответвления и метки. Поскольку такой встроенной классификации в Subversion 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 соответствующих 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.

Определение фиксации метки

Пожалуйста, обратите внимание, что эти шаблоны также используются для определения фиксаций в метку, а не только для графа ревизий.

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.

4.31.2.1. Цвета в графе ревизий

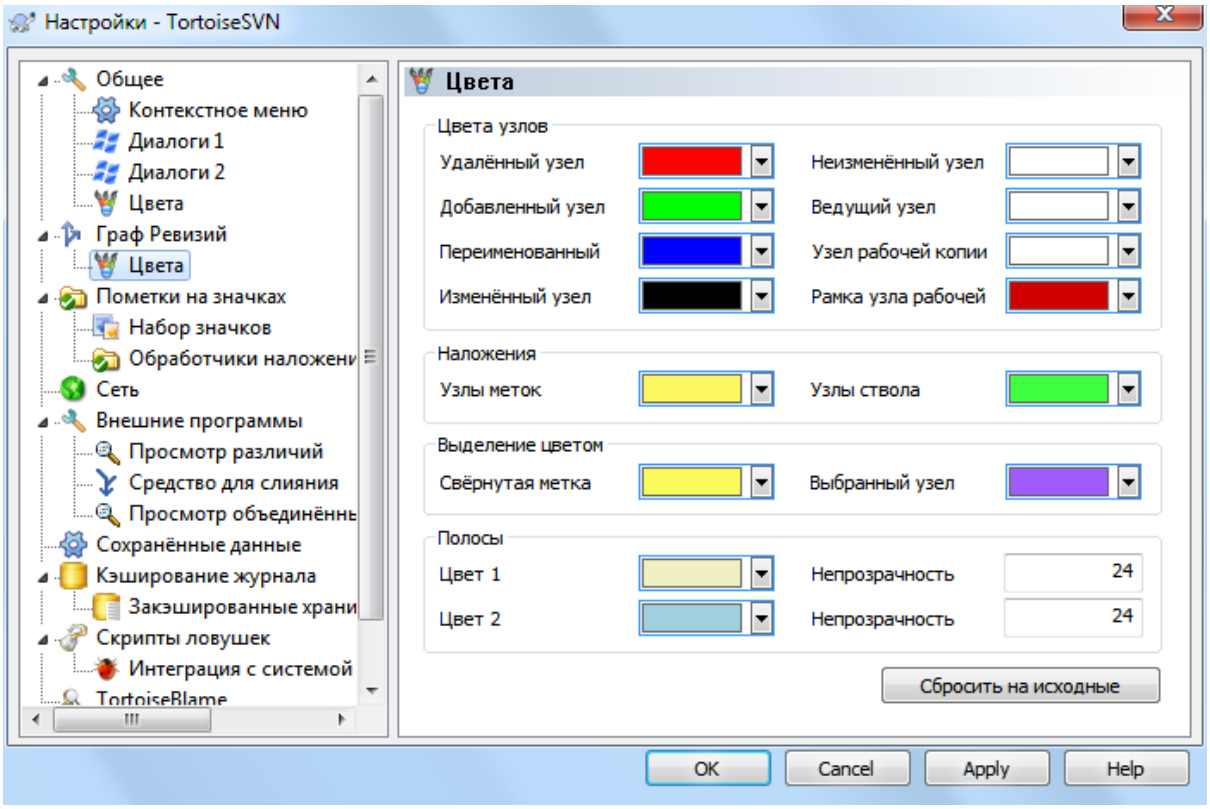


Рисунок 4.78. Страница 'Цвета' графа ревизий в диалоге настроек

Эта страница позволяет настроить используемые цвета. Учтите, что здесь указывается чистый цвет. Большинство узлов раскрашиваются смесью цвета типа узла, цвета фона и, при выборе, цветом классификации.

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.

4.31.3. Настройки пометок на значках

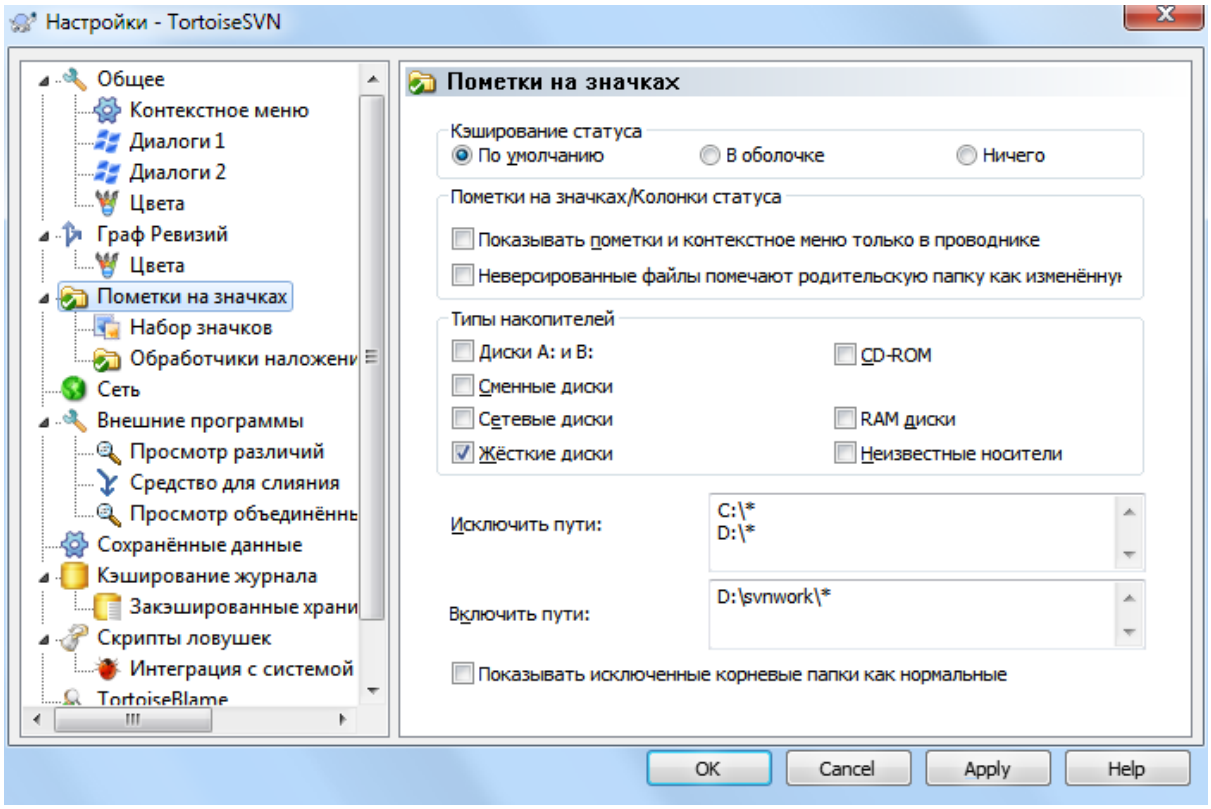


Рисунок 4.79. Страница 'Пометки на значках' в диалоге настроек

Эта страница позволяет выбрать элементы, для которых TortoiseSVN будет показывать пометки на значках.

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».

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.

You can force the status cache to *None* for elevated processes by checking the Disable status cache for elevated processes box. This is useful if you want to prevent another `TSVNCache.exe` process getting created with elevated privileges.

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).

Если у вас есть файлы в списке изменений `ignore-on-commit`, вы можете сделать так, чтобы эти файлы не распространяли свой статус на родительскую папку. Таким образом, если изменены только файлы в этом списке изменений, родительская папка всё же показывается с неизменённой оверлейной иконкой.

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 сообщить TortoiseSVN пути, для которых *не* надо отображать оверлейные значки и колонки статуса. Это полезно в случае когда у вас очень большие рабочие копии, содержащие только библиотеки, которые вы вообще не будете изменять и поэтому вам не нужны оверлеи, или же TortoiseSVN должен смотреть только в определенных папках.

Любой указанный здесь путь будет использоваться рекурсивно, так что ни в одной из дочерних папок не будут показаны оверлеи. Если вы хотите исключить *только* названную папку, то добавьте `?` после 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 эти три настройки взаимодействуют. Для любого заданного пути проверяются списки включения и исключения, ищутся совпадения вверх по иерархии директорий. Когда найдено первое совпадение проверяется соответствие правилу включения или исключения. Если есть конфликт, то определение одиночной директории имеет приоритет над рекурсивным определением, затем включение имеет приоритет над исключением.

Здесь вам поможет пример:

```
Исключить:
C:
C:\develop\?
C:\develop\tsvn\obj
C:\develop\tsvn\bin

Включить:
C:\develop
```

Эти настройки отключают оверлейные значки для диска C:, кроме `c:\develop`. Все проекты ниже этой директории будут отображать оверлейные значки, исключая саму папку `c:\develop`,

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 папка содержит рабочую копию. Флажок Показывать исключённые корневые папки как нормальные позволяет вам этого достичь: с его помощью корневые папки рабочих копий 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.

4.31.3.1. Выбор набора значков

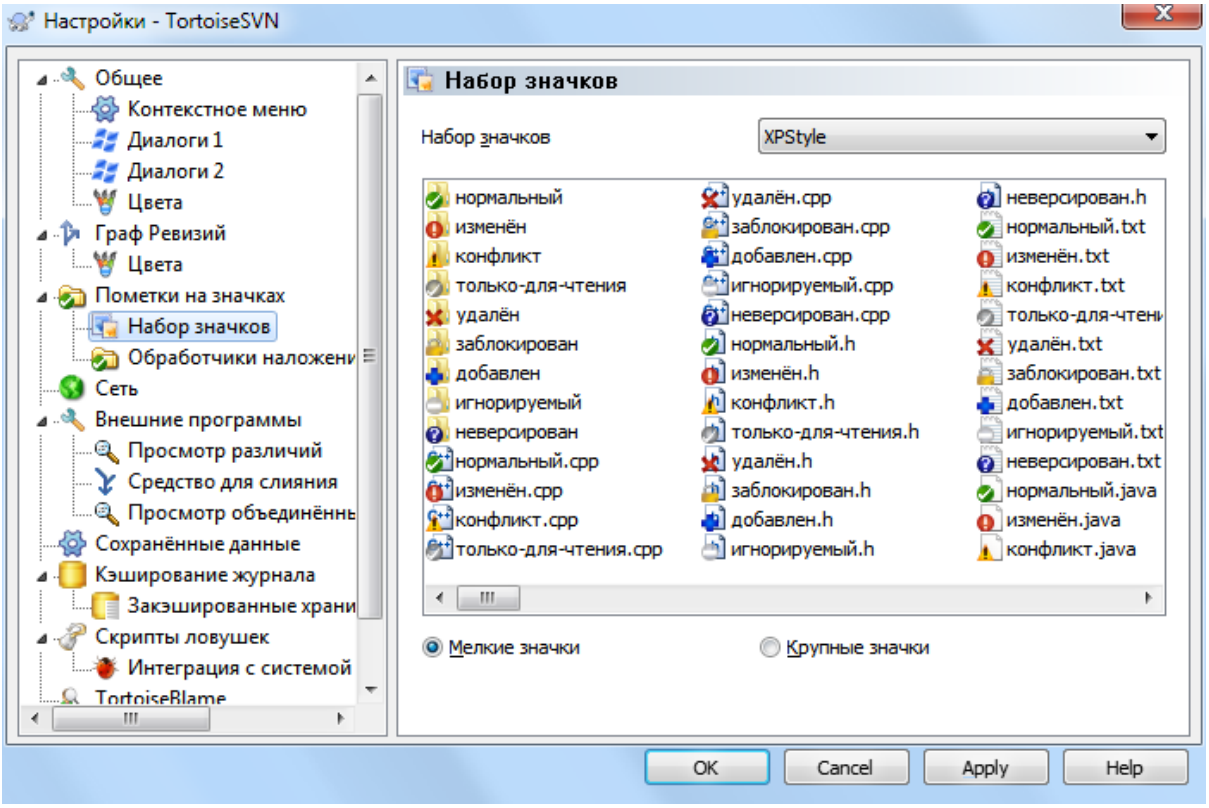


Рисунок 4.80. Страница 'Набор значков' в диалоге настроек

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.

4.31.3.2. Включить обработчики наложения

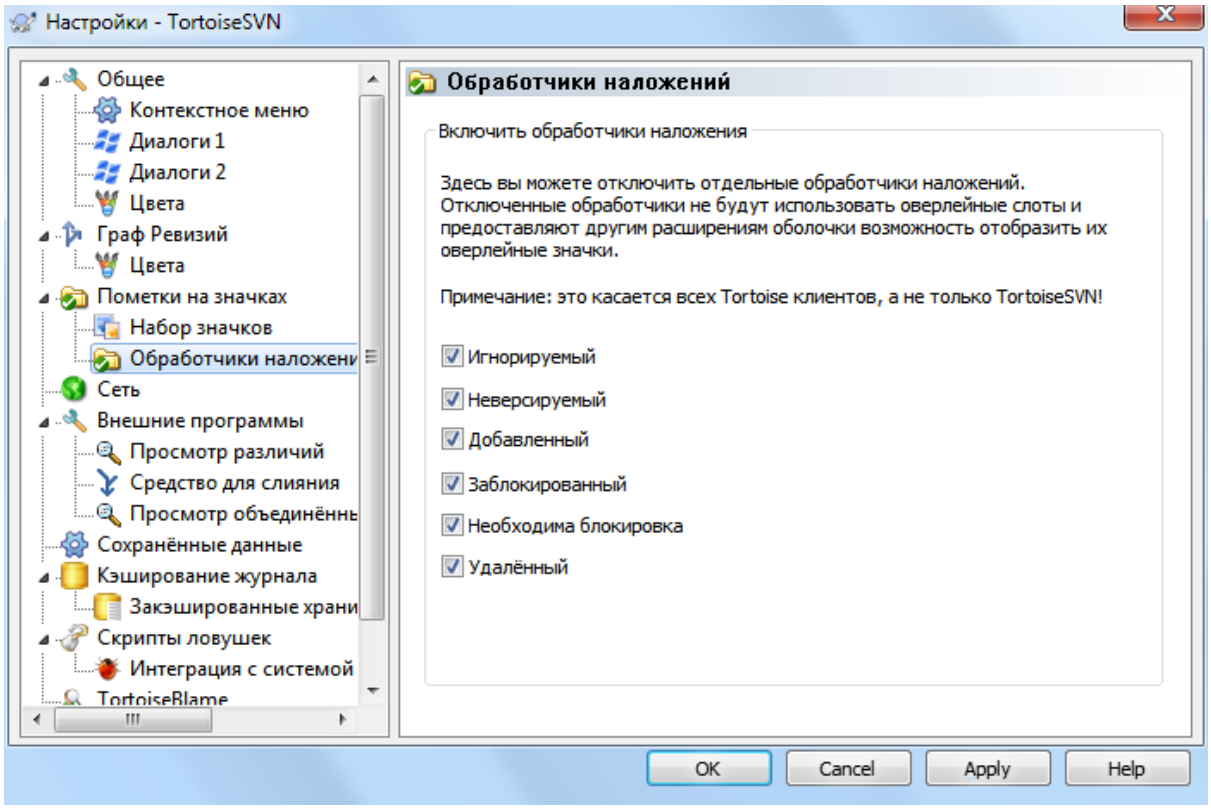


Рисунок 4.81. Страница 'Обработчики значков' в диалоге настроек.

По причине того, что количество оверлейных значков строго ограничено, вы можете отключить некоторые обработчики, чтобы загрузить те, которые вы хотите. Потому что TortoiseSVN использует общий компонент TortoiseOverlays, который используется совместно с другими Tortoise клиентами (например, TortoiseCVS, TortoiseHg) эта настройка также повлияет и на эти клиенты.

4.31.4. Настройки сети

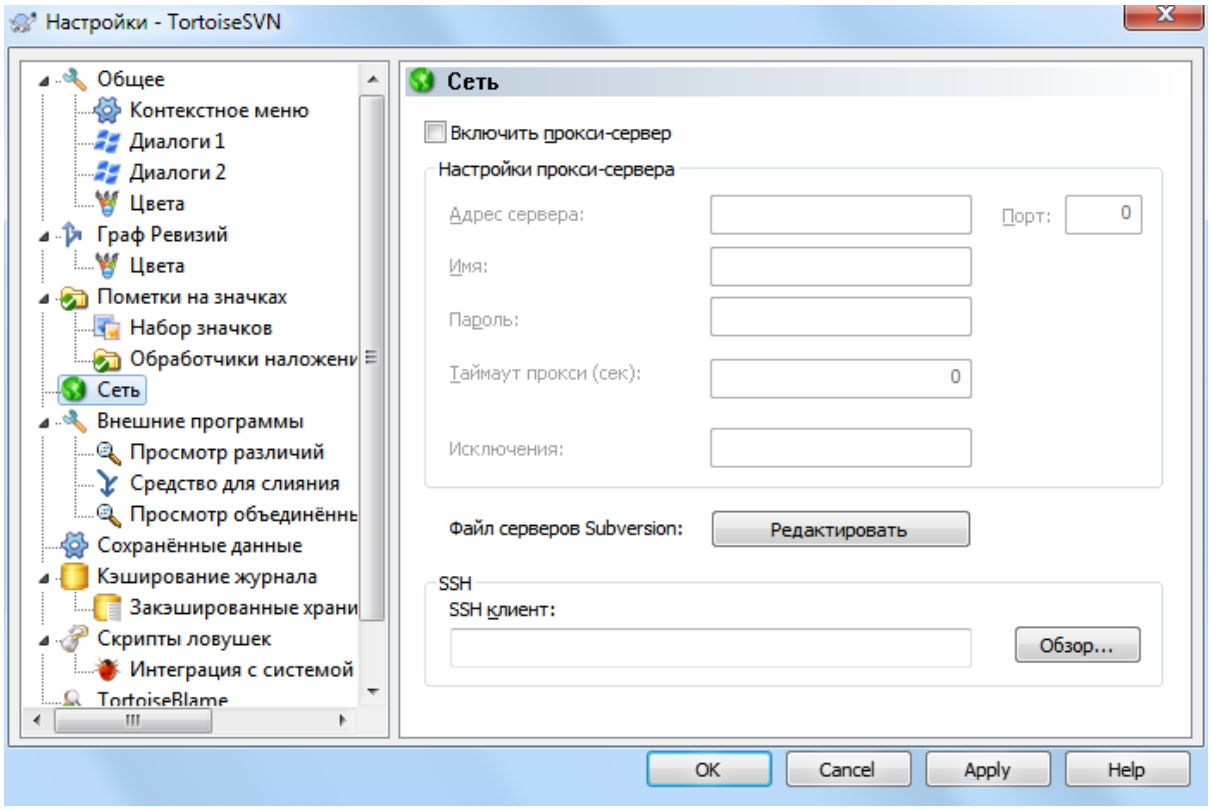


Рисунок 4.82. Страница 'Сеть' в диалоге настроек

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/1.8/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.

TortoisePlink does not have any documentation of its own because it is just a minor variant of Plink. Find out about command line parameters from the *PuTTY website* [https://www.chiark.greenend.org.uk/~sgtatham/putty/].

Чтобы избежать постоянного запроса пароля, вы можете также рассмотреть применение инструмента кэширования паролей, такого как Pageant. Его также можно скачать с веб-сайта PuTTY.

Finally, setting up SSH on server and clients is a non-trivial process which is beyond the scope of this help file. However, you can find a guide in the TortoiseSVN FAQ listed under *Subversion/TortoiseSVN SSH How-To* [https://tortoisesvn.net/ssh_howto.html].

4.31.5. Настройки внешних программ

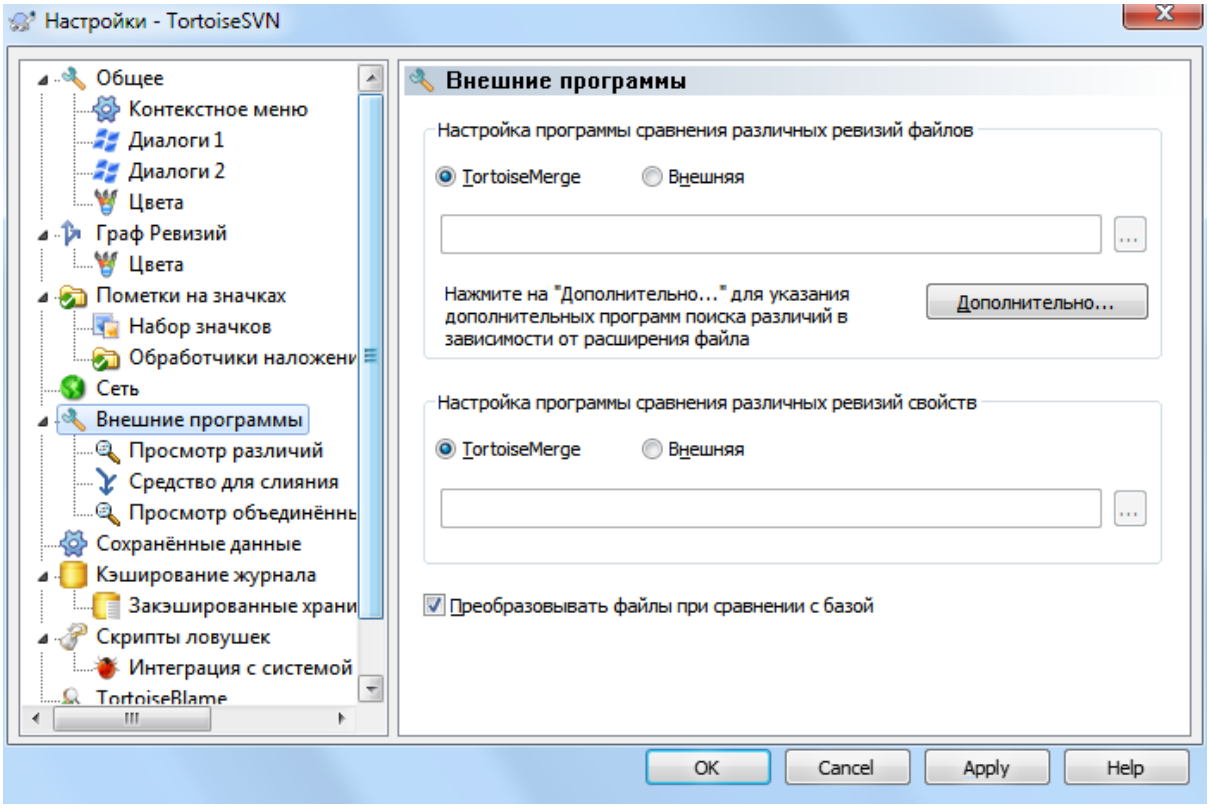


Рисунок 4.83. Страница 'Просмотр различий' в диалоге настроек

Здесь вы можете задать программу для проведения сравнения/слияния, которую будет использовать TortoiseSVN. По умолчанию используется TortoiseMerge, устанавливаемая совместно с TortoiseSVN.

Прочтите Раздел 4.11.6, «Внешние инструменты просмотра различий/слияния», где содержится список некоторых внешних программ сравнения/слияния, которые пользователи используют совместно с TortoiseSVN.

4.31.5.1. 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

%nqbname
Заголовок окна базового файла без кавычек

%mine
Ваш собственный файл, с вашими изменениями

%yname
Заголовок окна для вашего файла

%nqyname
Заголовок окна вашего файла без кавычек

%burl
Адрес URL исходного файла, если доступен

%nqburl
URL исходного файла без кавычек, если доступен

%yurl
Адрес URL второго файла, если доступен

%nqyurl
URL второго файла без кавычек, если доступен

%brev
Ревизия исходного файла, если доступна

%nqbrev
Ревизия исходного файла без кавычек, если доступен

%yrev
Ревизия второго файла, если доступна

%nqyrev
Ревизия второго файла без кавычек, если доступен

%peg
Опорная (peg) ревизия, если доступна

%nqpeg
Опорная (peg) ревизия без кавычек, если доступна

%fname
Имя файла. Пустая строка, если сравниваются два различных файла вместо двух состояний одного и того же файла.

%nqfname
Имя файла без кавычек

Заголовки окон это не просто имена файлов. TortoiseSVN рассматривает их как имена для отображения и создает соответствующие имена. Так, напимер, если вы сравниваете файл ревизии 123 с файлом в вашей рабочей копии, то имена будут `имя файла : ревизия 123` и `имя файла : рабочая копия`.

Например, с ExamDiff Pro:

```
C:\Path-To\ExamDiff.exe %base %mine --left_display_name:%bname
                                    --right_display_name:%yname
```

или с KDiff3:

```
C:\Path-To\kdiff3.exe %base %mine --L1 %bname --L2 %yname
```

или с WinMerge:

```
C:\Path-To\WinMerge.exe -e -ub -dl %bname -dr %yname %base %mine
```

или с Araxis:

```
C:\Path-To\compare.exe /max /wait /title1:%bname /title2:%yname
                       %base %mine
```

или с UltraCompare:

```
C:\Path-To\uc.exe %base %mine -title1 %bname -title2 %yname
```

или с DiffMerge:

```
C:\Path-To\DiffMerge.exe -nosplash -t1=%bname -t2=%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.

В нижней части диалога вы можете конфигурировать программу просмотра для объединенных файлов различий (файлов заплаток). Параметры не требуются. Настройка Default для использования TortoiseUDiff, которая устанавливается вместе с TortoiseSVN, и цветовое кодирование добавленых и удаленных строк.

Так как объединенные различия (Unified Diff) это простой текстовый формат, вы можете использовать ваш любимый текстовый редактор.

4.31.5.2. 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

%nqbname
: Заголовок окна базового файла без кавычек

%mine
: ваш собственный файл, с вашими изменениями

%yname
: Заголовок окна для вашего файла

%nqyname
: Заголовок окна вашего файла без кавычек

%theirs
: файл, каков он в хранилище

%tname
: заголовок окна для файла из хранилища

%nqtname
: Заголовок окна файла в хранилище без кавычек

%merged
: конфликтующий файл, результат операции слияния

%mname
: заголовок окна для объединённого файла

%nqmname
: Заголовок окна слитого файла без кавычек

%fname
: Имя конфликтующего файла

%nqfname
: Имя конфликтующего файла без кавычек

Например, с Perforce Merge:

```
C:\Path-To\P4Merge.exe %base %theirs %mine %merged
```

или с KDiff3:

```
C:\Path-To\kdiff3.exe %base %mine %theirs -o %merged
                      --L1 %bname --L2 %yname --L3 %tname
```

или с Araxis:

```
C:\Path-To\compare.exe /max /wait /3 /title1:%tname /title2:%bname
```

```
                        /title3:%yname %theirs %base %mine %merged /a2
```

или с WinMerge (2.8 or later):

```
C:\Path-To\WinMerge.exe %merged
```

или с DiffMerge:

```
C:\Path-To\DiffMerge.exe -caption=%mname -result=%merged -merge
      -nosplash -t1=%yname -t2=%bname -t3=%tname %mine %base %theirs
```

4.31.5.3. Дополнительные настройки сравнения/слияния

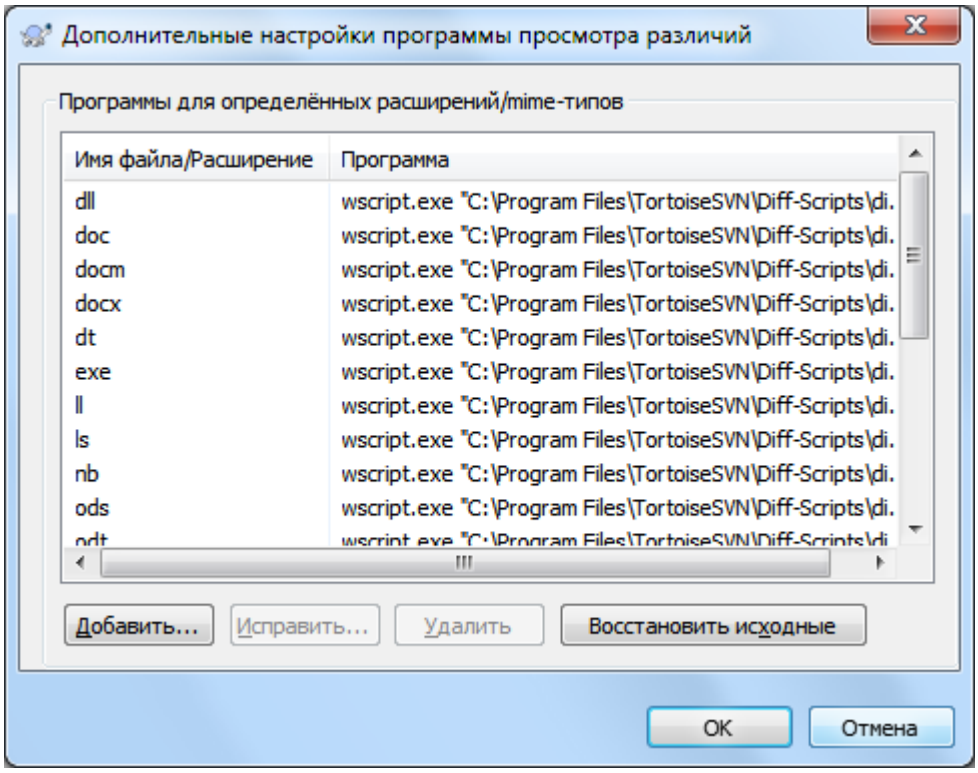


Рисунок 4.84. Окно дополнительных настроек сравнения/слияния в диалоге настроек

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`.

4.31.6. Настройки сохранённых данных

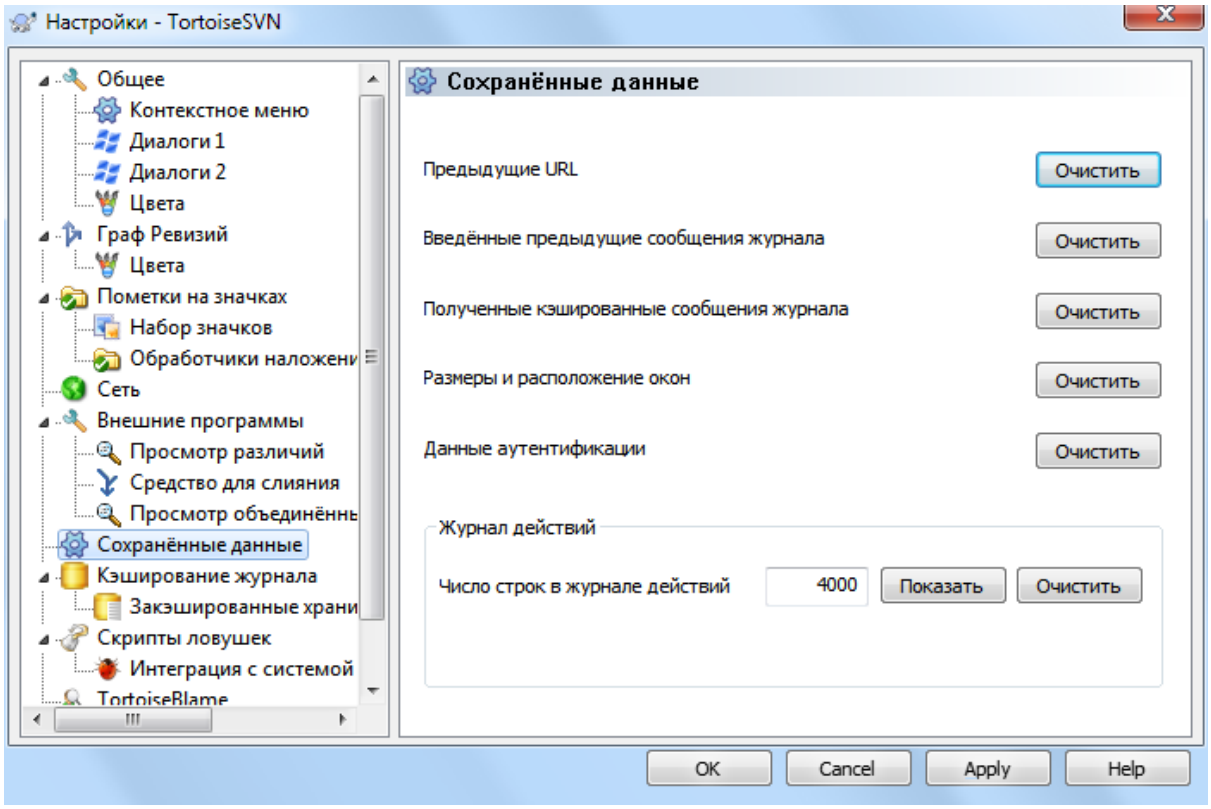


Рисунок 4.85. Страница 'Сохранённые данные' в диалоге настроек

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 хотите очистить данные аутентификации только для одного отдельного сервера, то используйте кнопку Очистить... вместо кнопки Очистить всё.

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.

4.31.7. Кэширование журнала

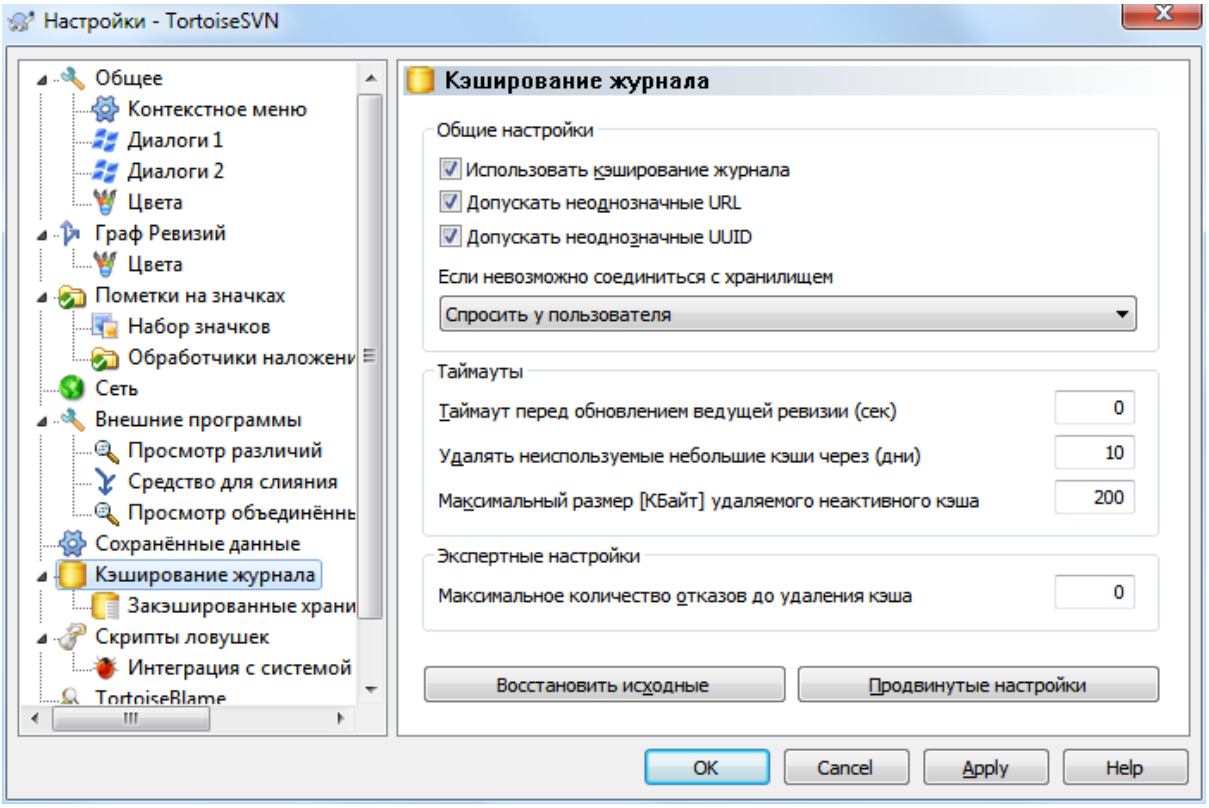


Рисунок 4.86. Страница 'Кэширование журнала' в диалоге настроек

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 один 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.

4.31.7.1. 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.

4.31.7.2. Статистика кэша журнала

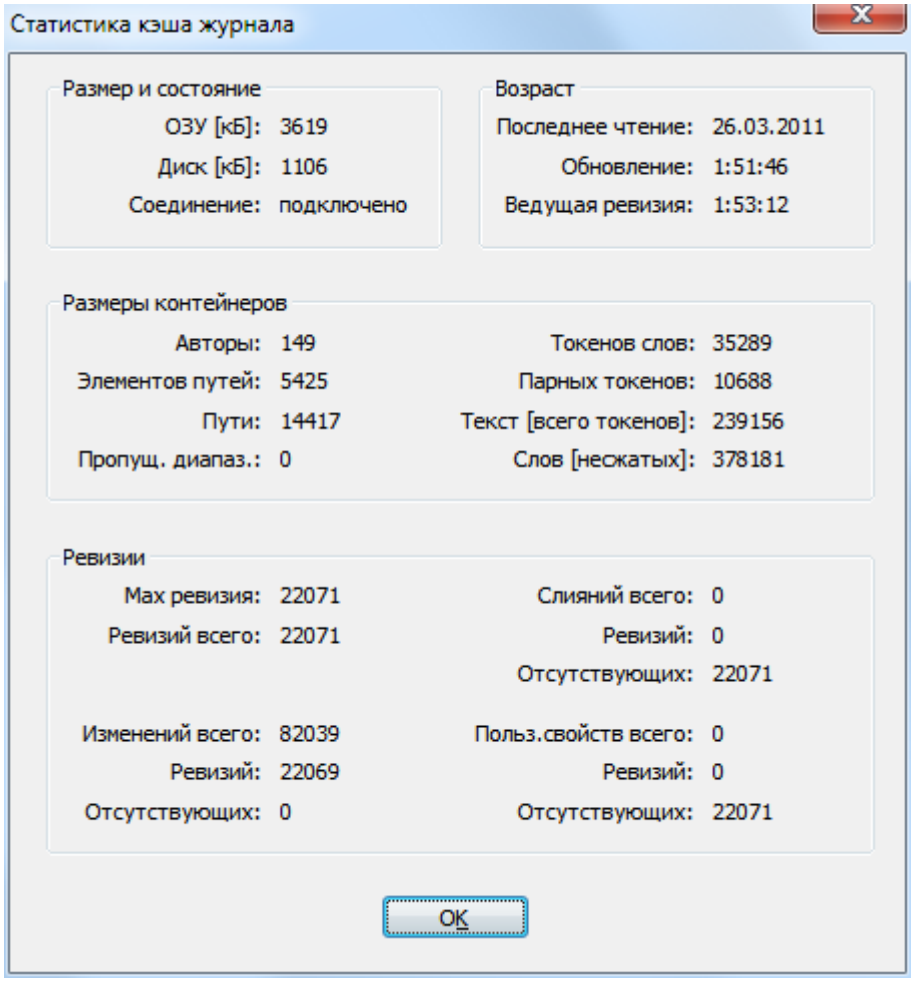


Рисунок 4.87. Окно 'Статистика кэша журнала', открываемое из диалога настроек

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.

4.31.8. Скрипты ловушек, выполняемые на стороне клиента

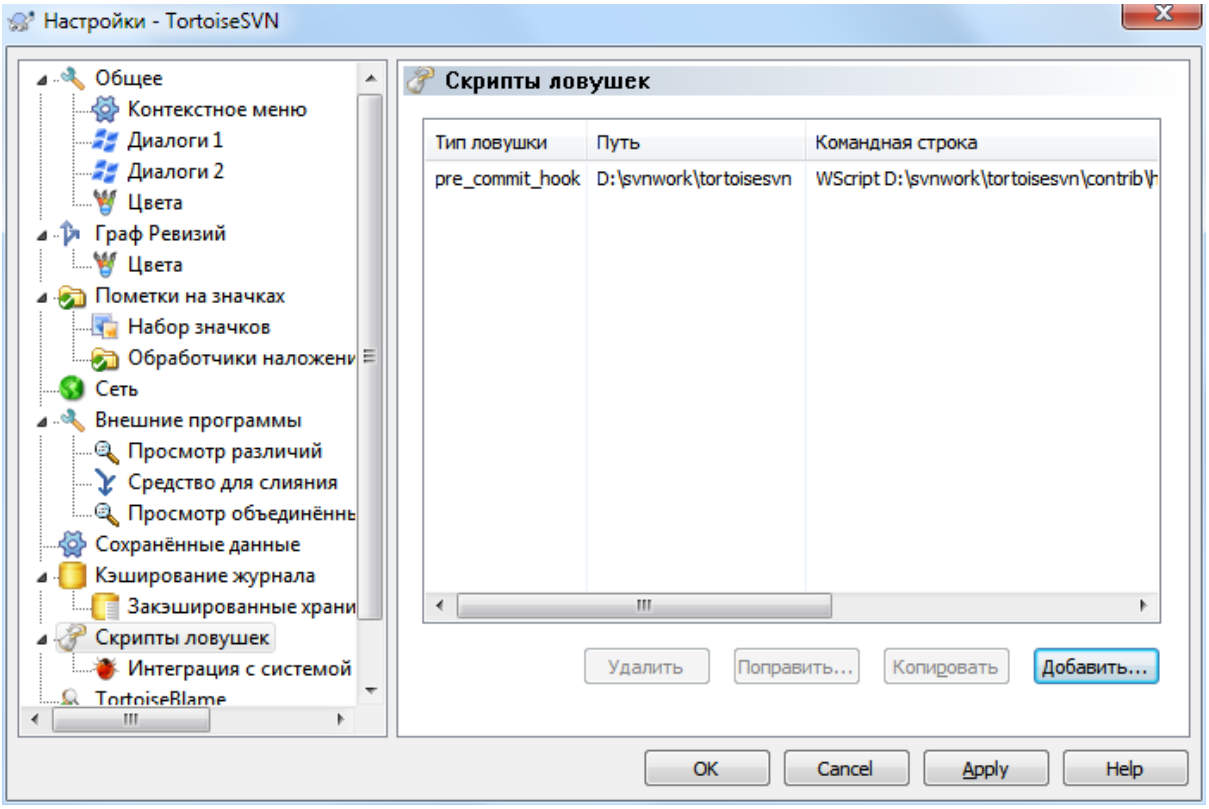


Рисунок 4.88. Страница 'Скрипты ловушек' в диалоге настроек

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.

Обратите внимание, что вы также можете задать скрипты-обработчики с помощью специальных свойств в вашей рабочей копии. Для более подробной информации см. Раздел 4.18.2, «Свойства проекта в TortoiseSVN».

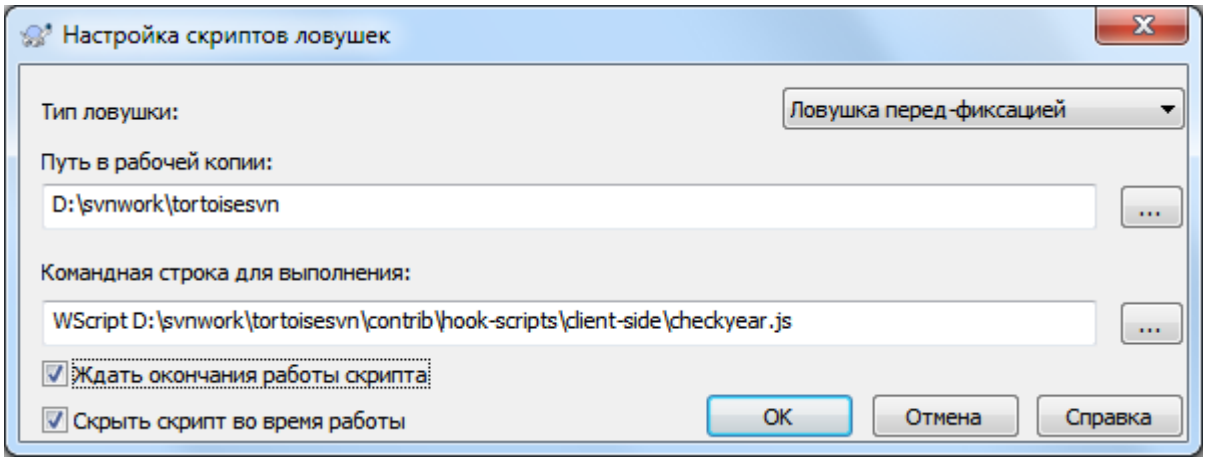


Рисунок 4.89. Окно 'Настройка скрипта ловушки', открываемое из диалога настроек

Чтобы добавить скрипт ловушки, просто щёлкните на Добавить... и заполните необходимые данные.

В настоящее время доступны скрипты-обработчики следующих типов

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 должны учитывать, что из-за того, что ловушка вызывается на ранней стадии, полный список выбранных для фиксации объектов не доступен.

Ручная перед-фиксацией
: If this is specified, the commit dialog shows a button Run Hook which when clicked runs the specified hook script. The hook script receives a list of all checked files and folders and the commit message if there was one entered.

Проверка-фиксации
: Вызывается после того как пользователь нажмет кнопку OK в диалоге фиксации, но до закрытия диалога фиксации. Эта ловушка получает список всех отмеченных файлов. Если ловушка вернет ошибку, то диалог фиксации останется открытым.

  Если возвращённое сообщение об ошибке содержит пути разделённые символами перевода строки, то такие пути станут выделенными в диалоге фиксации после отображения сообщения об ошибке.

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
: Called after the commit finishes successfully.

Начало-обновления
: Вызывается перед отображением диалога обновить-до-ревизии.

Перед-обновлением
: Вызывается перед фактическим началом операции Subversion 'обновление' или 'переключение'.

После-обновления
: Вызывается после окончания операции обновления, переключения или извлечения (независимо, успешного или нет).

Предварительное соединение
: Вызывается перед попыткой соединения с хранилищем. Вызывается не более одного раза в пять минут.

Пред-блокировка
: Вызывается перед попыткой заблокировать файл.

Пост-блокировка
: Вызывается после того, как файл был заблокирован.

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[4].

Далее вы должны указать командную строку для выполнения, начинающуюся с пути к скрипту ловушки или исполняемому файлу. Это может быть пакетный файл, исполняемый файл, или любой другой файл имеющий корректное расширение файла, например, скрипт на Perl. Обратите внимание, что скрипт не должен быть указан с помощью UNC пути, т. к. оболочка Windows не разрешает выполнение таких скриптов по причинам ограничений безопасности.

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`

Ручная перед-фиксацией
: `PATHMESSAGEFILECWD`

Проверка-фиксации
: `PATHMESSAGEFILECWD`

Перед-фиксацией
: `PATHDEPTHMESSAGEFILECWD`

После-фиксации
: `PATHDEPTHMESSAGEFILEREVISIONERRORCWD`

Начало-обновления
: `PATHCWD`

Перед-обновлением
: `PATHDEPTHREVISIONCWD`

После-обновления
: `PATHDEPTHREVISIONERRORCWDRESULTPATH`

Предварительное соединение
: no parameters are passed to this script. You can pass a custom parameter by appending it to the script path.

Пред-блокировка
: `PATHLOCKFORCEMESSAGEFILEERRORCWD`

Пост-блокировка
: `PATHLOCKFORCEMESSAGEFILEERRORCWD`

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.

[4]Для обозначения корня иерархии используется символ * - прим. переводчика

Обратите внимание, что для операций выполняемых удалённо, например, в обозревателе хранилища, эти пути не локальные, а URL-адреса задействованных элементов.

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.

LOCK
Или `true` при блокировке, или `false` при разблокировке.

FORCE
Либо `true`, либо `false`, в зависимости от того, была ли операция принудительной или нет.

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.

RESULTPATH
Путь ко временному файлу, который содержит все пути, которые были так или иначе затронуты операцией. Каждый путь в отдельной строке во временном файле.

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.

Примерный клиентский скрипт ловушки может быть найден в папке `contrib` в *хранилище TortoiseSVN* [https://svn.code.sf.net/p/tortoisesvn/code/trunk/contrib/hook-scripts]. (Раздел 3, «Лицензия» объясняет, как получить доступ к хранилищу.)

При отладке скриптов ловушек вы можете продублировать ход процесса на консоль или добавить паузу, чтобы пристановить закрытие окна после выполнения скрипта. По причине того, что ввод-вывод перенаправлен, это не будет работать как обычно. Однако, вы можете перенаправить явно ввод и вывод на устройство CON чтобы обойти это. Например,

```
echo Checking Status > con
pause < con > con
```

Небольшой инструмент добавленный в папку установки TortoiseSVN называется `ConnectVPN.exe`. Вы можете использовать этот инструмент, сконфигурированный как ловушку перед-соединением для автоматического подключения к вашей VPN перед тем как TortoiseSVN пытается соединится с хранилищем. Просто передайте этому инструменту имя соединения VPN первым параметром.

4.31.8.1. 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 4.29.2, «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.

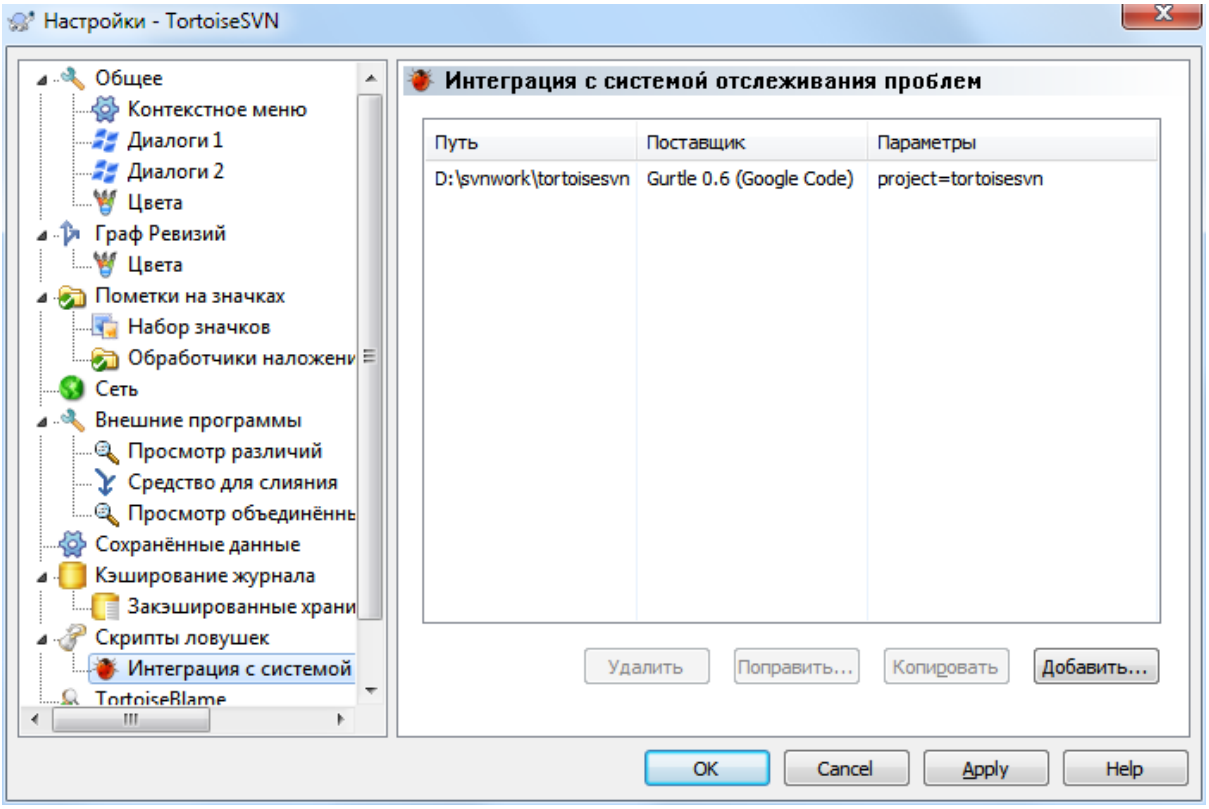


Рисунок 4.90. Страница интеграции с системой отслеживания проблем в диалоге настроек

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 хотите, чтобы все пользователи использовали один и тот же COM модуль для вашего проекта, вы можете указать этот модуль вместе со свойствами `bugtraq:provideruuid`, `bugtraq:provideruuid64` и `bugtraq:providerparams`.

bugtraq:provideruuid
: Это свойство определяет IBugtraqProvider COM UUID, например, `{91974081-2DC7-4FB1-B3BE-0DE1C8D6CE4E}`. (Это пример UUID *Gurtle bugtraq provider* [http://code.google.com/p/gurtle/], который является провайдером для системы отслеживания ошибок *Google Code* [http://code.google.com/hosting/].)

bugtraq:provideruuid64
: Это такое же свойство как и `bugtraq:provideruuid` только для 64-битной версии IBugtraqProvider.

bugtraq:providerparams
: Это свойство задаёт параметры, передаваемые IBugtraqProvider.

Прочтите документацию вашего модуля IBugtraqProvider, чтобы узнать, что указывать в данных двух свойствах.

4.31.9. Настройки TortoiseBlame

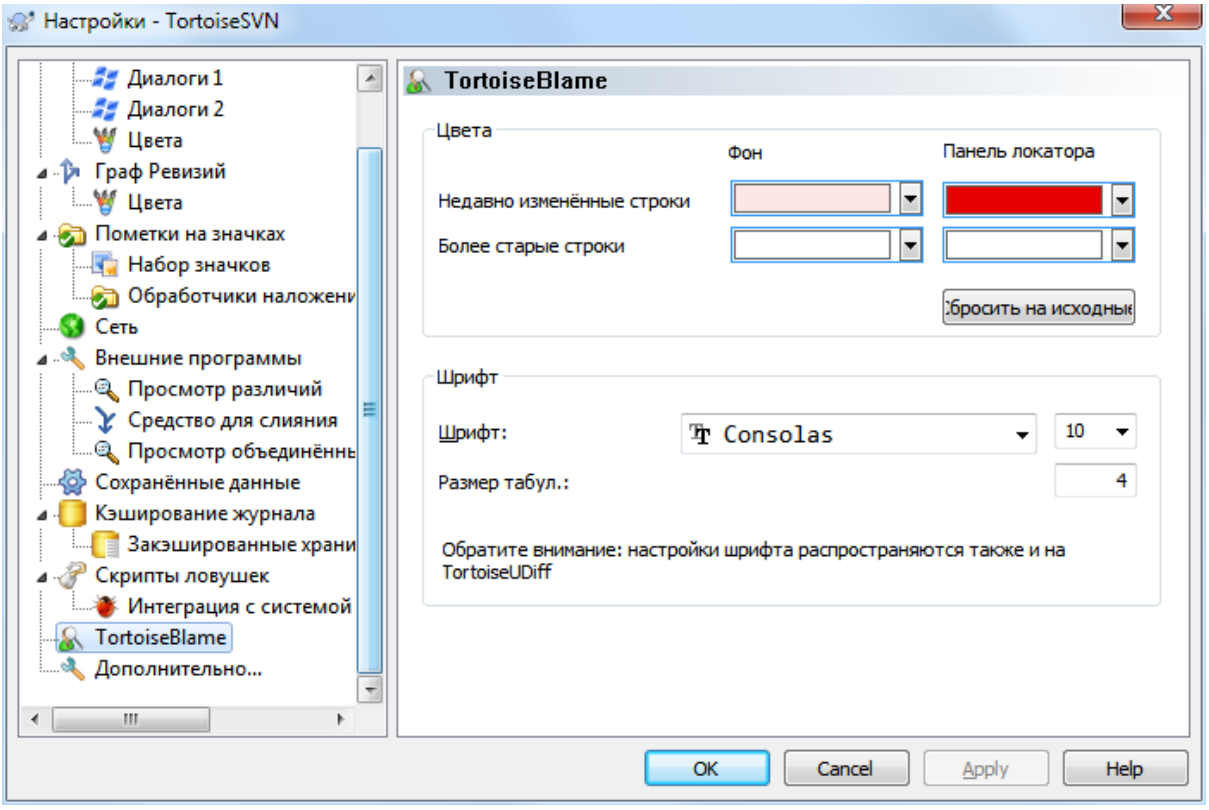


Рисунок 4.91. Страница TortoiseBlame в диалоге настроек

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.

Вы можете указать разные цвета для использования на полосе навигации. По умолчанию на полосе навигации используется насыщенный контраст при этом фон основного окна остается светлым, чтобы вы смогли прочитать текст.

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.

4.31.10. Настройки TortoiseUDiff

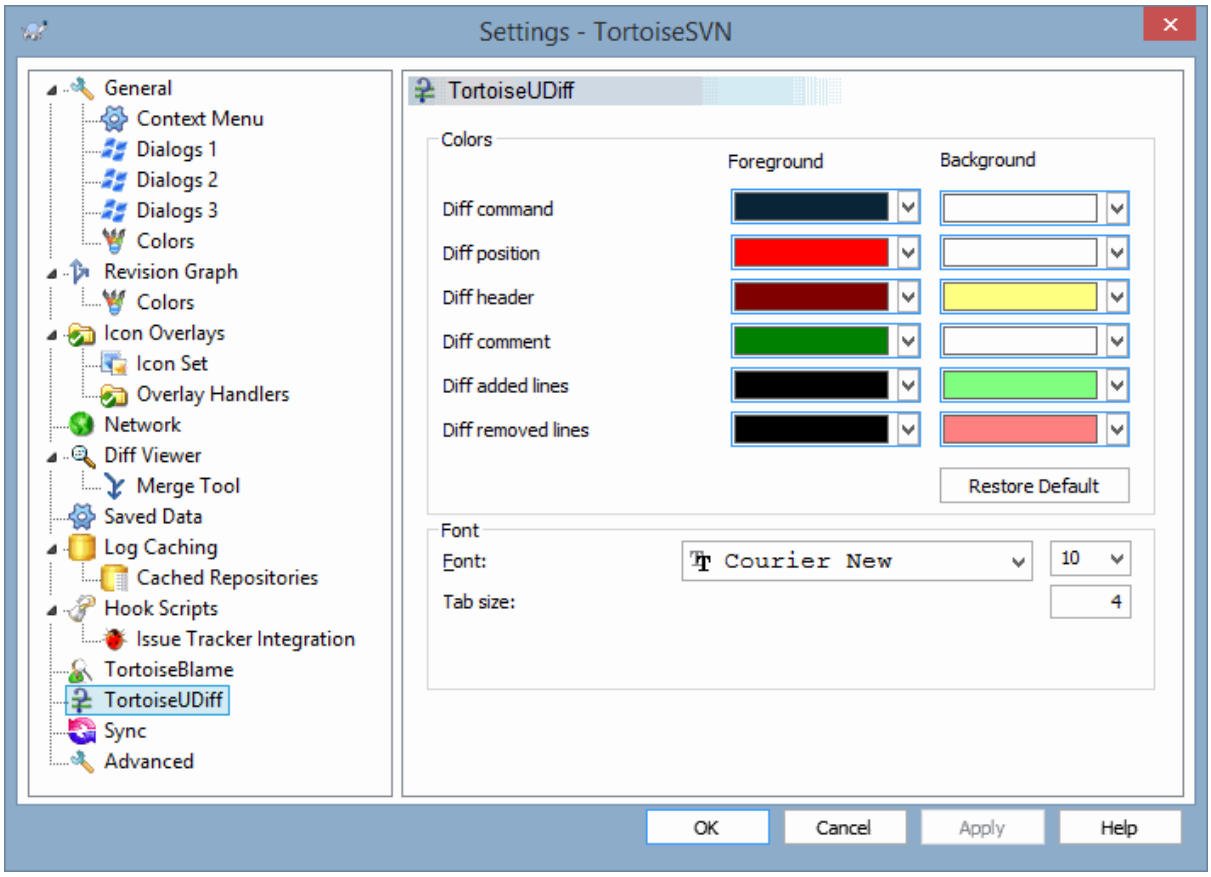


Рисунок 4.92. Диалог Настройки, страница TortoiseUDiff

Настройки используемые TortoiseUDiff управляются из основного контекстного меню, не напрямую из TortoiseUDiff.

Цвета
: Цвета используемые TortoiseUDiff по умолчанию обычно в порядке, но вы можете настроить их здесь.

Шрифт
: Вы можете выбрать начертание и размер шрифта, используемого для отображения текста.

Размер табул.
: Определяет, сколько пробелов использовать для замены символов табуляции, обнаруженных в файле различий.

4.31.11. Экспортировать TSVN настройки

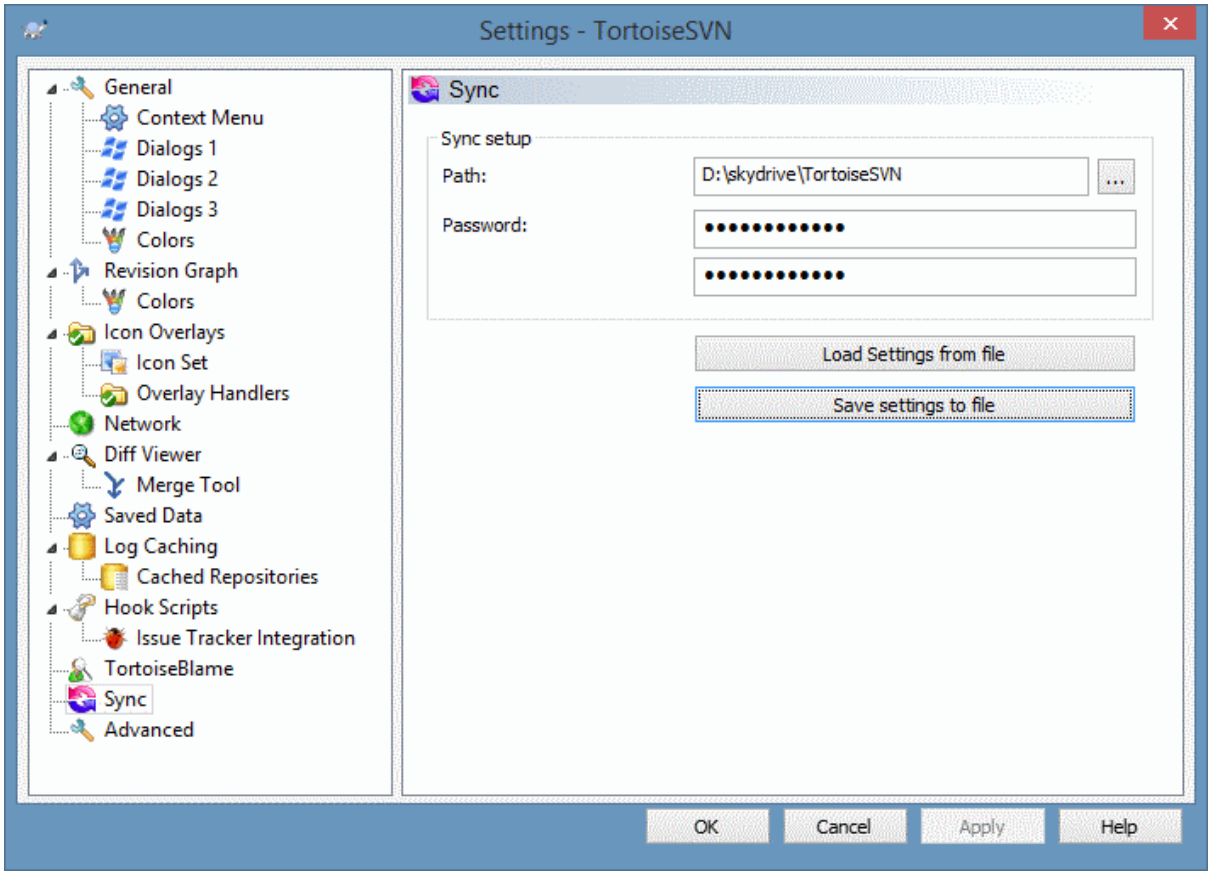


Рисунок 4.93. Диалог Настройки, страница Синхронизация

Вы можете синхронизировать все настройки TortoiseSVN через зашифрованный файл. Файл шифруется с помощью введённого пароля, так что вам не стоит беспокоиться, если храните этот файл в "облачной" папке, например, OneDrive, GDrive, DropBox, ...

Когда путь и пароль заданы TortoiseSVN будет синхронизировать все настройки автоматически и поддерживать их в синхронизации.

Вы также можете экспортировать/импортировать зашифрованные файлы со всеми настройками вручную. Когда вы делаете это, у вас спросят путь файла и пароль для шифрации/дешифрации файла настроек.

При экспорте настроек вручную вы также можете выборочно включить все локальные настройки, которые не включаются при обычном экспорте или синхронизации. Локальные настройки это настройки, которые включают локальные пути, которые обычно отличаются между компьютерами. Эти локальные настройки включают сконфигурированные инструменты различия и слияния, и скрипты-обработчики.

4.31.12. Дополнительные настройки

Некоторые редко используемые настройки доступны только на дополнительной странице диалога настроек. Эти настройки вносят изменения прямо в реестр и необходимо предварительно ознакомиться с тем, для чего используются эти настройки. Не рекомендуется изменять эти настройки, если вы не уверены, что вам необходимо их изменить.

AllowAuthSave
: Иногда несколько пользователей используют одну и ту же учетную запись на одном компьютере. В таких ситуациях не рекомендуется сохранять данные аутентификации. Установка значения этой настройки в `false` отключает кнопку `сохранить аутентификацию` в диалоге аутентификации.

AllowUnversionedObstruction
: Если при обновлении добавляется новый файл из хранилища, который уже существует в локальной рабочей копии как файл без версии, то действие по умолчанию - это оставить локальный файл, отображая его как (возможно) измененённую версию нового файла из хранилища. Если вы хотите, чтобы TortoiseSVN создавал конфликт в таких случаях, то установите значение `false`.

AlwaysExtendedMenu
: Как и в обозревателе, TortoiseSVN показывает дополнительные команды, если зажата клавиша **Shift** при открытии контекстного меню. Чтобы вынудить TortoiseSVN всегда показывать дополнительные команды, установите значение `true`.

AutoCompleteMinChars
: Минимальное количество символов, начиная с которых редактор показывает показывает всплывающее окно автозаполнения. Значение по умолчанию — `3`.

AutocompleteRemovesExtensions
: Список автозавершения, показываемый в редакторе сообщений фиксации, отображает список имен файлов для фиксации. Чтобы оставить имена файлов, но убрать расширения файлов, установите значение `true`.

BlockPeggedExternals
: Файлы внешних репозиториев, связанные с указанной ревизией, по умолчанию недоступны для фиксации. Это сделано из-за того, что их последующие обновления будут отменять сделанные изменения до тех пор, пока связанные ревизии не будут скорректированы.

  Установите значение в `false` в случае, если вы хотите зафиксировать изменения указанных файлов из внешних репозиториев.

BlockStatus
: Если вы не хотите, чтобы обозреватель обновлял показания статуса, пока выполняется другая команда TortoiseSVN (например, Обновить, Фиксировать, ...), то установите значение `true`.

CacheTrayIcon
: Чтобы добавить иконку кэширования в системный лоток для программы TSVNCache, установите значение `true`. Это может быть полезным только для разработчиков, поскольку позволяет изящно завершить программу.

ColumnsEveryWhere
: Дополнительные колонки, которые TortoiseSVN добавляет к подробному виду в обозревателе Windows, обычно активны только в рабочей копии. Если вы хотите, чтобы это было доступно везде, а не только в рабочей копии, то установите значение `true`. Имейте в виду, что дополнительные колонки появляются только в Windows XP. В Vista и более поздних версиях ОС эта функция больше не поддерживается. Однако некоторые заменители проводника от других производителей поддерживают такое даже для версий Windows после XP.

ConfigDir
: Здесь можно указать другое месторасположение для конфигурационного файла Subversion. Это затронет все операции TortoiseSVN.

CtrlEnter
: В большинстве диалоговых окон TortoiseSVN вы можете использовать сочетание клавиш **Ctrl+Enter**, чтобы закрывать окна, также как и при нажатии кнопки ОК. Если же вы хотите выключить эту функцию, то установите значение `false`.

Debug
: Установите это в `true` если хотите чтобы при каждом запуске TortoiseProc.exe появлялся диалог показывающий командную строку с параметрами запуска.

DebugOutputString
: Установите это в `true` если хотите чтобы TortoiseSVN печатал отладочные сообщения во время выполнения. Эти сообщения могут быть отловлены только специальными инструментами.

DialogTitles
: Формат по умолчанию (значение 0) заголовков диалогов `url/path - name of dialog - TortoiseSVN`. Если вы установите значение в 1, то формат изменится на `name of dialog - url/path - TortoiseSVN`.

DiffBlamesWithTortoiseMerge
: TortoiseSVN позволяет вам назначать внешнюю программу просмотра изменений. Многие из таких программ, однако, не совместимы с программой авторства изменений (Раздел 4.24.2, «Авторство различий»), поэтому вы можете использовать TortoiseMerge в этом случае. Для этого установите значение `true`.

DlgStickySize
: Это значение определяет количество пикселей, на которое диалог приближается к границе перед тем как приклеится к ней. Значение по умолчанию 3. Для отключения этого значения установите значение в ноль.

FixCaseRenames
: Некоторые приложения изменяют регистр имени файлов без предупреждения, но в действительности никакой необходимости в таких изменениях нет. Например, изменение `file.txt` на `FILE.TXT` не повредит обычному Windows-приложению, но Subversion в таких ситуациях чувствителен к регистру. Так что TortoiseSVN автоматически исправляет такие изменения.

  Если же вы не хотите, чтобы TortoiseSVN автоматически исправлял такие изменения регистра, то можете установить это значение в `false`.

FullRowSelect
: Контроль за списком статуса, используемый в различных диалогах (например, фиксировать, проверить на наличие изменений, добавить, обновить до ревизии, ...), использует полностроковое выделение (т.е. если вы выбираете одну запись, то выделяется весь ряд, а не только первая колонка). Это прекрасно, но выделение выбранного ряда также затрагивает фоновое изображение внизу справа, которое в результате может неправильно отобразиться. Чтобы отключить полностроковое выделение, установите значение `false`.

GroupTaskbarIconsPerRepo
: Эта настройка определяет, как группируются иконки различных диалогов и окон TortoiseSVN на панели задач Windows 7. Эта настройка не работает в Vista.

1. Значение по умолчанию — 0. При такой настройке иконки сгруппированы по типу приложения. Все диалоги TortoiseSVN сгруппированы вместе, все окна TortoiseMerge сгруппированы вместе, ...

Рисунок 4.94. Панель задач с группировкой по-умолчанию

2. Если установлено значение 1, то вместо всех диалогов сгруппированных в одну группу на приложение, они группируются по хранилищу. Например, если у вас есть диалог журнала сообщений и диалог фиксации, открытые для хранилища `A`, и диалог проверки наличия изменений и диалог журнала для хранилища `B`, то отображаются две группы иконок приложения в панели задач Windows 7, одна группа для каждого хранилища. Но окна TortoiseMerge не группируются с диалогами TortoiseSVN.

Рисунок 4.95. Панель задач с группировкой по хранилищам

3. Если установлено значение 2, то группировка работает также как и при значении 1, за исключением того, что окна TortoiseSVN, TortoiseMerge, TortoiseBlame, TortoiseIDiff и TortoiseUDiff

группируются вместе. Например, если у вас есть открытый диалог фиксации и вы делаете двойной щелчок на изменённом файле, то открываемое окно просмотра различий TortoiseMerge будет помещено в ту же группу иконок на панели задач, что и иконка диалога фиксации.

Рисунок 4.96. Панель задач с группировкой по хранилищам

4. Если установлено значение 3, то группировка работает также как при значении 1, но группировка выполняется в соответствии с рабочей копией, а не хранилищем. Это полезно когда все ваши проекты находятся в одном хранилище, но для каждого проекта создана отдельная рабочая копия.

5. Если установлено значение 4, то группировка работает также как при значении 2, но группировка выполняется в соответствии с рабочей копией, а не хранилищем.

HideExternalInfo
: Если установлено `false`, то каждое `svn:externals` при обновлении отображается отдельно.

  Если установлено `true` (по умолчанию), то информация об обновлении внешних показывается только если обновление затрагивает внешние, т.е. изменены каким-либо образом. Иначе ничего не отображается, как и с обычными файлами и папками.

GroupTaskbarIconsPerRepoOverlay
: Это не работает если настройка `GroupTaskbarIconsPerRepo` установлена в значение 0 (см. выше).

  Если эта настройка установлена в `true`, то каждая иконка на панели задач Windows 7 отображает маленький цветной прямоугольник с оверлейным значком, показывающий для какого хранилища используется диалог/окно.

Рисунок 4.97. Группировка панели задач с цветным оверлеем хранилища

IncludeExternals
: По умолчанию, TortoiseSVN всегда выполняет обновление с включенными внешними объектами. Это помогает избежать проблем с несовместимостью рабочих копий. Если же, однако, у вас много заданных внешних объектов, то обновление может занять долгое время. Если установить значение `false`, то обновление будет проходить с выключенными внешними объектами. Чтобы выполнить обновление с включёнными внешними объектами, используйте команду `Обновить до ревизии...` или верните значение `true`.

LogFindCopyFrom
: Когда диалог журнала запускается из мастера объединений, объединенные ревизии отображаются серым цветом, но при этом также отображаются ревизиии за пределами точки ветвления. Эти ревизии отображаются черным цветом, потому что они не могут быть объединены.

Если эта опция `включена` , то TortoiseSVN попытается найти ревизию, от которой была создана ветка, и скроет все ревизии, которые находятся за пределами этой ревизии. Так как процесс может занять неопределенное время, эта опция отключена по умолчанию. Кроме того, эта функция не работает с некоторыми серверами SVN (например, Google Code Hosting, см. *вопрос # 5471 citetitle> ulink>).* [http://code.google.com/p/support/issues/detail?id=5471]

LogMultiRevFormat
Строка формата для сообщений журнала при множественном выборе ревизий в диалоге журнала.

Вы можете использовать следующие подстановки в строке формата:

%1!ld!
заменяется номером ревизии

%2!s!
заменяется коротким сообщением ревизии

LogStatusCheck
Диалоговое окно журнала показывает ревизию и путь к рабочей копии жирным шрифтом. Но это требует, чтобы диалоговое окно журнала скопировало статус этого пути. Поскольку для очень больших рабочих копий это может занять довольно много времени, вы можете установить значение `false`, чтобы деактировать эту функцию.

MergeLogSeparator
Когда вы выполняете слияние ревизий из другого отвлетвления, и доступна информация об отслеживании за изменениями, то сообщения журнала из ревизий, которые вы собираетесь слить, будут собраны в одно сообщение журнала. Предопределённая строка используется для разделения отдельных сообщений журнала в слитых ревизиях. Если вы желаете, вы можете установить значение в виде того разделителя, который вы хотите использовать.

NumDiffWarning
Если вы хотите показать изменения сразу для большего количества элементов, чем указано в этой настройке, вы увидите диалоговое окно с предупреждением. Значение по умолчанию 10.

OldVersionCheck
TortoiseSVN проверяет на наличие новой версии примерно раз в неделю. Если существует новая версия, то диалоговое окно фиксации показывает ссылку с этой информацией. Если вы предпочитаете старое поведение, когда появляется диалоговое окно, извещающее об обновлении, то установите значение `true`.

RepoBrowserTrySVNParentPath
Обозреватель хранилища пробует получить веб-страницу, которую генерирует сервер SVN сконфигурированный директивой SVNParentPath, чтобы получить список всех хранилищ. Чтобы отключить такое поведение, установите это значение в `false`.

ScintillaDirect2D
Эта опция включает использование ускоренную отрисовку Direct2D в элементе Scintilla, который используется как поле ввода, например, в диалоге фиксации, и также для просмотра унифицированных различий. С некоторыми графическими картами, тем не менее, это иногда работает некорректно, так что не всегда виден курсор для ввода текста. Если такое происходит, то вы можете выключить эту возможность установив значение `false`.

OutOfDateRetry
This parameter specifies how TortoiseSVN behaves if a commit fails due to an out-of-date error:

0
The user is asked whether to update the working copy or not, and the commit dialog is not reopened after the update.

1
This is the default. The user is asked whether to update the working copy or not, and the commit dialog is reopened after the update so the user can proceed with the commit right away.

2
: Similar to `1`, but instead of updating only the paths selected for a commit, the update is done on the working copy root. This helps to avoid inconsistent working copies.

3
: The user is not asked to update the working copy. The commit simply fails with the out-of-date error message.

ShellMenuAccelerators
: TortoiseSVN использует ускорители для своих пунктов контекстного меню обозревателя. Поскольку это может привести к дублированию ускорителей (например, у `SVN Фиксировать` есть **Alt-C** ускоритель, но у пункта меню обозревателя `Копировать` он тоже есть). Поэтому, если вы не хотите или вам не нужно использовать ускорители в пунктах TortoiseSVN, установите значение `false`.

ShowContextMenuIcons
: Это может пригодиться, если вы используете другой обозреватель файлов (не обозреватель Windows) или если у вас некорректно отображается контекстное меню. Установите значение `false`, если вы не хотите, чтобы TortoiseSVN отображал иконки для пунктов меню в контекстной оболочке. Верните значение `true`, чтобы снова отображать иконки.

ShowAppContextMenuIcons
: Если вы не хотите, чтобы TortoiseSVN показывал иконки для контекстного меню в своих диалогах, установите значение `false`.

StyleCommitMessages
: Диалоговые окна журнала и фиксации используют стили (например, жирный, курсивный шрифт) в сообщениях фиксации (прочитайте Раздел 4.4.5, «Сообщения журнала при фиксации», чтобы узнать больше). Если вы не хотите использовать это, установите значение `false`.

UpdateCheckURL
: Это значение содержит URL, по которому TortoiseSVN загружает текстовый файл для проверки на наличие обновлений. Это может полезно для администраторов компаний, которые не хотят, чтобы пользователи обновляли TortoiseSVN без их разрешения.

VersionCheck
: TortoiseSVN проверяет на наличие новой версии примерно раз в неделю. Если вы не хотите, чтобы TortoiseSVN выполнял эту проверку, то установите значение `false`.

4.32. 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.

While working on TortoiseSVN we love to listen to music. And since we spend many hours on the project we need a *lot* of music. Therefore we have set up some wish-lists with our favourite music CDs and DVDs: https://tortoisesvn.net/donate.html Please also have a look at the list of people who contributed to the project by sending in patches or translations.

Глава 5. Монитор проекта

Монитор проекта — это полезный инструмент, который следит за хранилищами и уведомляет вас в случае если есть новые фиксации.

Проекты могут отслеживаться через путь рабочей копии или напрямую через адреса URL их хранилищ.

Монитор проекта сканирует каждый проект через заданный интервал времени, и каждый раз когда обнаруживаются новые фиксации показывается всплывающее окно с уведомлением. Также иконка, добавленная в системный трей, изменяется чтобы показать, что есть новые фиксации.

Snarl

If *Snarl* [https://snarl.fullphat.net/] is installed and active, then the project monitor automatically uses Snarl to show the notifications about newly detected commits.

5.1. Добавление проектов к отслеживанию

Если вы впервые запускаете монитор проекта, то древовидное представление слева пусто. Для добавления проектов кликните по кнопке Добавить проект в верхней части диалога.

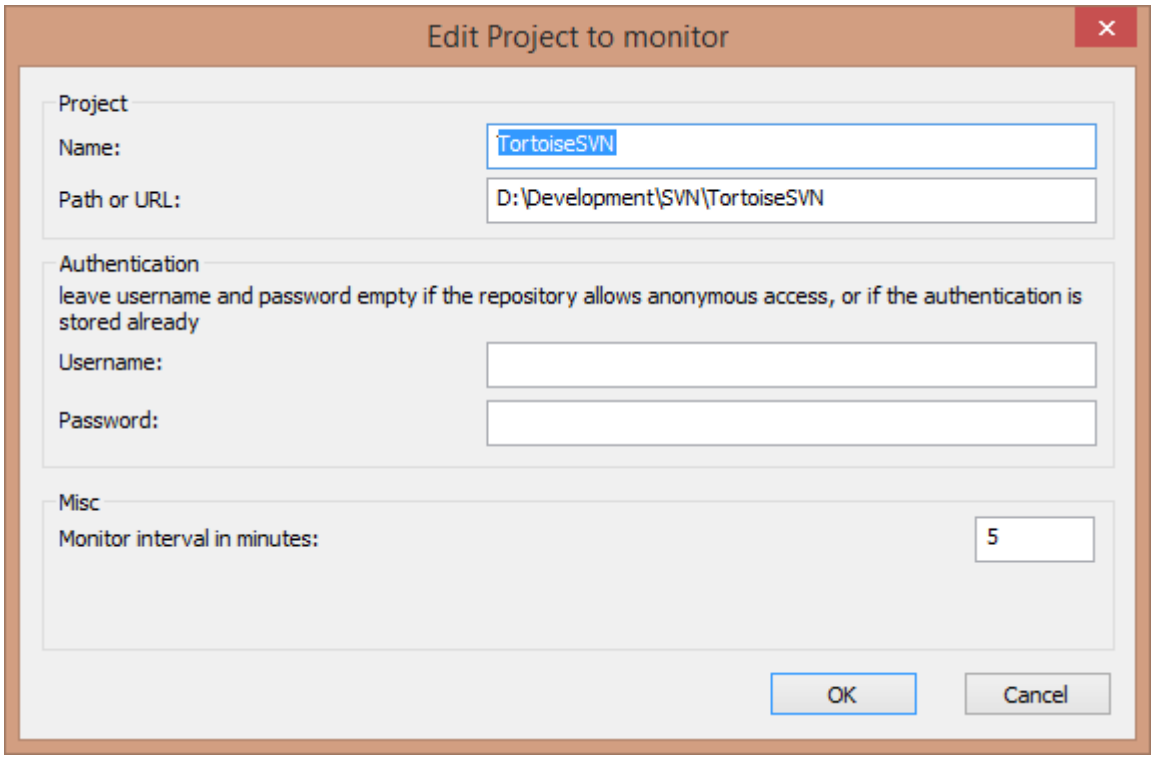


Рисунок 5.1. Диалог редактирования проекта монитора проекта

Чтобы добавить проект для мониторинга введите требуемую информацию. Имя проекта обязательно и должно быть заполнено, вся остальная информация необязательна.

Если поле `Путь или URL` оставить пустым, то добавляется папка. Это полезно для группировки отслеживаемых проектов.

Поля `Имя пользователя` и `Пароль` должны быть заполнены только если к хранилищу нет анонимного доступа на чтение, и только если аутентификация не сохранена Subversion. Если вы подключаетесь к отслеживаемому хранилищу с помощью TortoiseSVN или других svn-клиентов и вы уже сохранили данные

аутентификации, то вы должны оставить это пустым: вы не должны редактировать эти проекты вручную если пароль изменится.

`Интервал монитора в минутах` устанавливает количество минут между проверками. Наименьший интервал одна минута.

интервал проверки

Если много пользователей отслеживают одно и то же хранилище и полоса пропускания на сервере ограничена, то администратор хранилища может установить минимальное значение интевала проверки в файле `svnrobots.txt`. Более подробное объяснение как это работает можно найти на вебсайте монитора проекта:

http://stefanstools.sourceforge.net/svnrobots.html [http://stefanstools.sourceforge.net/svnrobots.html]

5.2. Диалог монитора

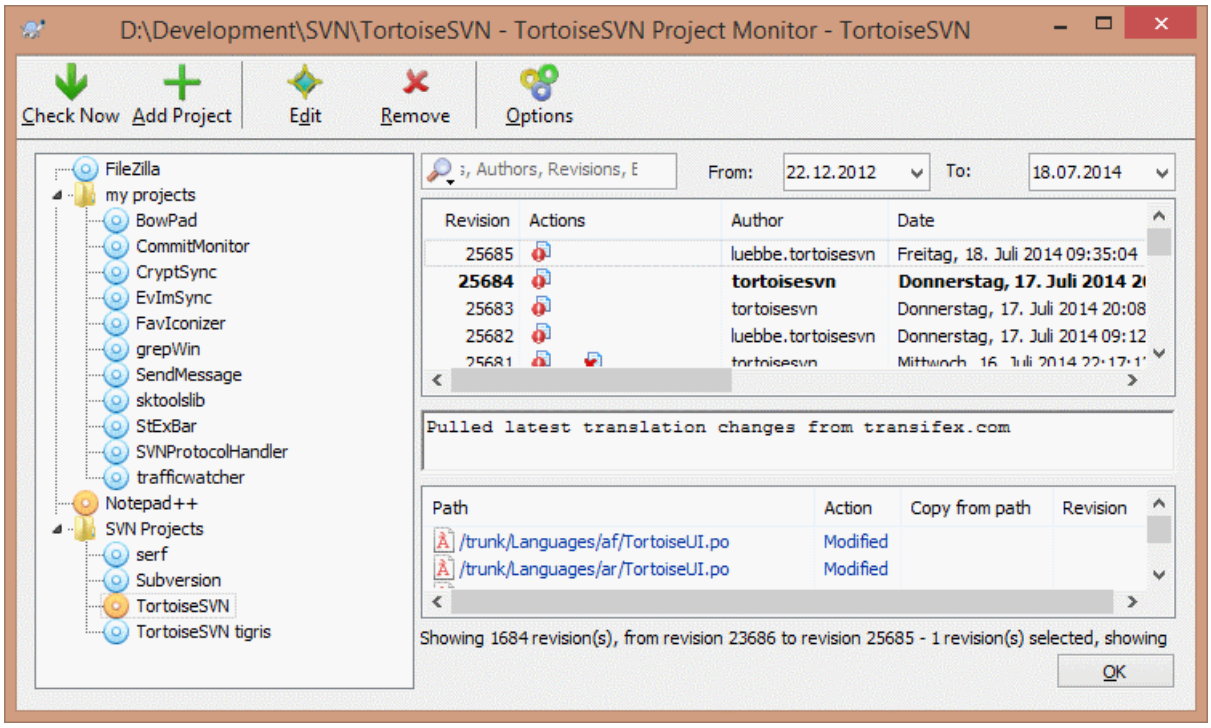


Рисунок 5.2. Основной диалог монитора проекта

Монитор проекта показывает все отслеживаемые проекта слева в древовидном представлении. Проекты можно перемещать, например, один проект может быть перемещен под другой проект, делая его подпроектом.

Клик на строке проекта отображает справа все сообщения журнала этого проекта.

Проекты, в которых произошли обновления, отображаются жирным шрифтом. Количество новых коммитов заключено в скобки и находится справа. Клик на строке проекта помечает его как содержащий только прочитанные уведомления.

5.2.1. Основные операции

Панель инструментов в верхней части диалога позволяет конфигурировать и управлять монитором проекта.

Проверить сейчас
: Тогда как каждый отслеживаемый проект проверяется в соответствии с заданным интервалом, нажатие на эту кнопку немедленно запустит проверку всех проектов. Обратите внимание, что если есть обновления, то уведомление не появится пока все проекты не будут проверены.

Добавить проект
: Открывает новый диалог настройки нового проекта для мониторинга.

Редактировать
: Открывает диалог конфигурации выбранного проекта.

Удалить
: Удаляет выбранный проект после показа диалога подтверждения.

Пометить все как прочитанное
: Отмечает все ревизии во всех проектах как прочитанные. Обратите внимание, что если вы выберете проект с непрочитанными ревизиями, то эти ревизии автоматически помечаются как прочитанные когда вы выбираете другой проект.

  Если вы удерживаете нажатой клавишу **Shift**, когда кликаете на кнопку, то все имеющиеся ошибочные состояния также будут удалены.

Обновить все
: Выполняет команду **Обновить** для всех отслеживаемых рабочих копий. Проекты отслеживаемые по URL не обновляются, только те, которые настроены на путь рабочей копии.

Параметры
: Показывает диалог для конфигурирования поведения монитора проекта.

Глава 6. 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».

6.1. Командная строка SubWCRev

SubWCRev читает статус Subversion всех файлов в рабочей копии, по умлочанию исключая внешние ссылки. SubWCRev записывает наибольшую найденную ревизию и фиксирует время изменения этой ревизии, также записывается были ли локальные изменения в рабочей копии или обнвление смешанных ревизий. Номер ревизии, диапазон ревизий обновления и статус модификации выводится в 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.

Вы можете указать шабоны игнорирования для SubWCRev, чтобы исключить определенные файлы и папки из наблюдения. Шаблоны считываются из файла с именем `.subwcrevignore`. Файл считывается из определенной папки, а также из корневой папки рабочей копии. Если файл не существует, то никакие файлы и папки не игнорируются. Файл `.subwcrevignore` может содержать несколько шаблонов, разделенных новой строкой. Шаблоны сопоставляются с путями корня хранилища и с путями каталога, где находится файл `.subwcrevignore`. Например, для игнорирования всех файлов в папке `doc` рабочей копии TortoiseSVN файл `.subwcrevignore` должен содержать следующие строки:

```
/trunk/doc
/trunk/doc/*
```

Или, если файл `.subwcrevignore` извлечен из trunk и находится в корне рабочей копии, то он должен содержать шаблоны

```
doc
doc/*
```

- такие же, как и в примере выше.

Чтобы игнорировать все изображения, шаблоны игнорирования могут быть установлены следующим образом:

```
*.png
*.jpg
```

```
*.ico
*.bmp
```

Важно

Шаблоны игнорирования чувствительны к регистру, также как в Subversion.

Подсказка

Чтобы создать файл с начальной точкой в проводнике Windows введите `.subwcrevignore`. Обратите внимание на точку в конце.

Есть некоторое количество опциональных параметров, влияющих на работу SubWCRev. Если вы используете более одного, то они должны быть заданы одной группой, 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.
-N	Если указан данный параметр, SubWCRev будет завершаться с `ERRORLEVEL 11`, если рабочая копия содержит неверсированные элементы, которые не игнорируются.
-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.
-E	Если указан ключ, такой как `-e`, то внешние и явные изменения игнорируются, в случае если эти изменения - внешние изменения в параметрах.
-x	Если указан этот параметр, SubWCRev будет выводить номера ревизий в шестнадцатеричном виде.
-X	Если указан этот параметр, SubWCRev будет выводить номера ревизий в шестнадцатеричном виде, с префиксом '0X'.
-F	Если задан этот переключатель, SubWCRev будет игнорировать любые `.subwcrevignore` файлы и включит все файлы.
-q	Если указан данный параметр, SubWCRev выполнит подстановку ключевых слов без вывода статуса рабочей копии в stdout.

Таблица 6.1. Список доступных параметров командной строки

Если ошибок нет, то SubWCRev возвращает ноль. Но, в случае возникновения ошибки сообщение об ошибке пишется в stderr и выводится на консоль. Возвращаемые коды ошибок следующие:

Код ошибки	Описание
1	Ошибка синтаксиса. Один или несколько параметров в командной строке неверны.
2	Файл или папка указанные в команде не были найдены

Код ошибки	Описание
3	Входной файл не может быть открыт, или целевой файл не может быть создан.
4	Не удалось выделить память. Это может произойти если, например, исходный файл слишком большой.
5	Исходный файл не может быть проверен должным образом.
6	Ошибка SVN: Subversion вернул ошибку когда SubWCRev пытался получить информацию из рабочей копии.
7	В рабочей копии есть локальные изменения. Требуется параметр `-n`.
8	В рабочей копии есть смешанные ревизии. Требуется параметр `-m`.
9	Выходной файл уже существует. Требуется параметр `-d`.
10	Указанный путь не является рабочей копией или её частью.
11	В рабочей копии имеются неверсированные файлы или папки. Требуется параметр `-N`.

Таблица 6.2. Список кодов ошибок SubWCRev

6.2. 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.
$WCREV&$	Заменяется на наибольшую зафиксированную ревизию в рабочей копии, дополняется значением после символа &. Например: `$WCREV&0xFFFF$`
$WCREV-$, $WCREV+$	Заменяется на наибольшую зафиксированную ревизию в рабочей копии, добавляя или отнимая значение после символа + или -. Например: `$WCREV-1000$`
$WCDATE$, $WCDATEUTC$	Заменяется на дату и время фиксации наибольшей ревизии. По умолчанию используется международный формат: `yyyy-mm-dd hh:mm:ss`. В качестве альтернативы вы можете указать пользовательский формат, который будет использован в `strftime()`, например: `$WCDATE=%a %b %d %I:%M:%S %p$`. Для получения списка возможных символов форматирования обратитесь к *онлайн-справочнике* [http://msdn.microsoft.com/en-us/library/fe06s4ak.aspx].
$WCNOW$, $WCNOWUTC$	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 ревизий обновления в рабочей копии. Если рабочая копия в согласованном состоянии, то это будет единственная ревизия. Если рабочая копия содержит смешанные ревизии, или устарела, или преднамеренно была обновлена до определенной ревизии, то диапазон будет показан в форме 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.
$WCUNVER$	`$WCUNVER?TText:FText$` заменяется на `TText` если в рабочей копии есть неверсированные элементы, или `FText` если нет.

Ключевое слово	Описание
$WCEXTALLFIXED$	Если все внешние изменения заменены на явные. то `$WCEXTALLFIXED?TText:FText$` заменяется на `TText`. Иначе он заменяется на `FText`.
$WCISTAGGED$	Если URL хранилища содержит шаблон классификации тегов. то `$WCISTAGGED?TText:FText$` заменяется на `TText`. Иначе он заменяется на `FText`.
$WCURL$	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.
$WCLOCKDATE$, $WCLOCKDATEUTC$	Заменяется на дату блокировки. Может быть использован формат даты/времени, описанный для `$WCDATE$`.
$WCLOCKOWNER$	Заменяется на имя владельца блокировки.
$WCLOCKCOMMENT$	Заменяется на комментарий блокировки.
$WCUNVER$	`$WCUNVER?TText:FText$` заменяется на `TText`, если в рабочей копии присутствуют неверсионные файлы или каталоги, или на `FText`, если таких файлов нет.

Таблица 6.3. Список доступных ключевых слов

SubWCRev напрямую не поддерживает вложение выражений, поэтому вы не можете использовать, к примеру, выражение

```
#define SVN_REVISION "$WCMIXED?$WCRANGE$:$WCREV$$"
```

Но вы можете обойти это с помощью других средств. Например,

```
#define SVN_RANGE $WCRANGE$
#define SVN_REV $WCREV$
#define SVN_REVISION "$WCMIXED?SVN_RANGE:SVN_REV$"
```

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 `$WCLOCKCOMMENT$`.

6.3. Пример для ключевых слов

Нижеприведённый пример показывает, как происходит замена ключевых слов при переходе от шаблонного файла к целевому.

```
// Проверочный файл для SubWCRev
```

```
char *Revision      = "$WCREV$";
char *Revision16    = "$WCREV&0xFF$";
char *Revisionp100  = "$WCREV+100$";
char *Revisionm100  = "$WCREV-100$";
char *Modified      = "$WCMODS?Modified:Not modified$";
char *Unversioned   = "$WCUNVER?Unversioned items found:no unversioned items$";
char *Date          = "$WCDATE$";
char *CustDate      = "$WCDATE=%a, %d %B %Y$";
char *DateUTC       = "$WCDATEUTC$";
char *CustDateUTC   = "$WCDATEUTC=%a, %d %B %Y$";
char *TimeNow       = "$WCNOW$";
char *TimeNowUTC    = "$WCNOWUTC$";
char *RevRange      = "$WCRANGE$";
char *Mixed         = "$WCMIXED?Mixed revision WC:Not mixed$";
char *ExtAllFixed   = "$WCEXTALLFIXED?All externals fixed:Not all externals fixed$";
char *IsTagged      = "$WCISTAGGED?Tagged:Not tagged$";
char *URL           = "$WCURL$";
char *isInSVN       = "$WCINSVN?versioned:not versioned$";
char *needslck      = "$WCNEEDSLOCK?TRUE:FALSE$";
char *islocked      = "$WCISLOCKED?locked:not locked$";
char *lockdateutc   = "$WCLOCKDATEUTC$";
char *lockdate      = "$WCLOCKDATE$";
char *lockcustutc   = "$WCLOCKDATEUTC=%a, %d %B %Y$";
char *lockcust      = "$WCLOCKDATE=%a, %d %B %Y$";
char *lockown       = "$WCLOCKOWNER$";
char *lockcmt       = "$WCLOCKCOMMENT$";

#if $WCMODS?1:0$
#error Source is modified
#endif

// End of file
```

После запуска `SubWCRev.exe путь\к\рабочей\копии testfile.tmpl testfile.txt`, выходной файл `testfile.txt` будет выглядеть подобно этому:

```
// Тестовый файл для SubWCRev

char *Revision      = "22837";
char *Revision16    = "53";
char *Revisionp100  = "22937";
char *Revisionm100  = "22737";
char *Modified      = "Модифицированный";
char *Unversioned   = "нет неверсированных элементов";
char *Date          = "2012/04/26 18:47:57";
char *CustDate      = "Четверг, 26 Апрель 2012";
char *DateUTC       = "2012/04/26 16:47:57";
char *CustDateUTC   = "Четверг, 26 Апрель 2012";
char *TimeNow       = "2012/04/26 20:51:17";
char *TimeNowUTC    = "2012/04/26 18:51:17";
char *RevRange      = "22836:22837";
char *Mixed         = "Смешанная ревизия WC";
char *ExtAllFixed   = "Все внешние зафиксированы";
char *IsTagged      = "Не отмечен";
char *URL           = "https://svn.code.sf.net/p/tortoisesvn/code/trunk";
char *isInSVN       = "версирован";
```

```
char *needslck      = "FALSE";
char *islocked      = "не заблокирован";
char *lockdateutc   = "1970/01/01 00:00:00";
char *lockdate      = "1970/01/01 01:00:00";
char *lockcustutc   = "Четверг, 01 Январь 1970";
char *lockcust      = "Четверг, 01 Январь 1970";
char *lockown       = "";
char *lockcmt       = "";

#if 1
#error Источник модифицирован
#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.

6.4. 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`.
.GetWCInfo2	`GetWCInfo()` с четырьмя параметрами эквивалентен параметру командной строки `-E`.
.Revision	Наибольшая ревизия фиксации в рабочей копии. Соответствует `$WCREV$`.
.Date	Дата и время наибольшей ревизии фиксации. Соответствует `$WCDATE$`.
.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
.HasUnversioned	Истина, если есть неверсированные элементы
.Url	Заменяется на адрес URL хранилища рабочей копии использованной в `GetWCInfo`. Эквивалентно `$WCURL$`.
.IsSvnItem	True, если элемент версирован.

Метод	Описание
.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 6.4. Поддерживаемые методы COM/автоматизации

Следующий пример показывает, как может быть использован этот интерфейс.

```
// testCOM.js - файл javascript
// Проверочный сценарий для объекта COM/Automation 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.GetWCInfo2(
    filesystem.GetAbsolutePathName("..\\.."), 1, 1, 1);

wcInfoString1 = "Revision = " + revObject1.Revision +
                "\nMin Revision = " + revObject1.MinRev +
                "\nMax Revision = " + revObject1.MaxRev +
                "\nDate = " + revObject1.Date +
                "\nURL = " + revObject1.Url + "\nAuthor = " +
                revObject1.Author + "\nHasMods = " +
                revObject1.HasModifications + "\nIsSvnItem = " +
                revObject1.IsSvnItem + "\nNeedsLocking = " +
                revObject1.NeedsLocking + "\nIsLocked = " +
                revObject1.IsLocked + "\nLockCreationDate = " +
                revObject1.LockCreationDate + "\nLockOwner = " +
                revObject1.LockOwner + "\nLockComment = " +
                revObject1.LockComment;
wcInfoString2 = "Revision = " + revObject2.Revision +
                "\nMin Revision = " + revObject2.MinRev +
                "\nMax Revision = " + revObject2.MaxRev +
                "\nDate = " + revObject2.Date +
                "\nURL = " + revObject2.Url + "\nAuthor = " +
                revObject2.Author + "\nHasMods = " +
                revObject2.HasModifications + "\nIsSvnItem = " +
                revObject2.IsSvnItem + "\nNeedsLocking = " +
                revObject2.NeedsLocking + "\nIsLocked = " +
                revObject2.IsLocked + "\nLockCreationDate = " +
```

```
                revObject2.LockCreationDate + "\nLockOwner = " +
                revObject2.LockOwner + "\nLockComment = " +
                revObject2.LockComment;
wcInfoString3 = "Revision = " + revObject3.Revision +
                "\nMin Revision = " + revObject3.MinRev +
                "\nMax Revision = " + revObject3.MaxRev +
                "\nDate = " + revObject3.Date +
                "\nURL = " + revObject3.Url + "\nAuthor = " +
                revObject3.Author + "\nHasMods = " +
                revObject3.HasModifications + "\nIsSvnItem = " +
                revObject3.IsSvnItem + "\nNeedsLocking = " +
                revObject3.NeedsLocking + "\nIsLocked = " +
                revObject3.IsLocked + "\nLockCreationDate = " +
                revObject3.LockCreationDate + "\nLockOwner = " +
                revObject3.LockOwner + "\nLockComment = " +
                revObject3.LockComment;
wcInfoString4 = "Revision = " + revObject4.Revision +
                "\nMin Revision = " + revObject4.MinRev +
                "\nMax Revision = " + revObject4.MaxRev +
                "\nDate = " + revObject4.Date +
                "\nURL = " + revObject4.Url + "\nAuthor = " +
                revObject4.Author + "\nHasMods = " +
                revObject4.HasModifications + "\nIsSvnItem = " +
                revObject4.IsSvnItem + "\nNeedsLocking = " +
                revObject4.NeedsLocking + "\nIsLocked = " +
                revObject4.IsLocked + "\nLockCreationDate = " +
                revObject4.LockCreationDate + "\nLockOwner = " +
                revObject4.LockOwner + "\nLockComment = " +
                revObject4.LockComment;

WScript.Echo(wcInfoString1);
WScript.Echo(wcInfoString2);
WScript.Echo(wcInfoString3);
WScript.Echo(wcInfoString4);
```

Следующий пример показывает, как может быть использован COM-объект SubWCRev из C#:

```
using LibSubWCRev;
SubWCRev sub = new SubWCRev();
sub.GetWCInfo("C:\\Путь\\к\\файлу\\МойФайл.cc", true, true);
if (sub.IsSvnItem == true)
{
    MessageBox.Show("версирован");
}
else
{
    MessageBox.Show("не версирован");
}
```

Глава 7. Интерфейс IBugtraqProvider

Для более тесной интеграции с системами отслеживания проблем, нежели простое использование свойств `bugtraq:`, TortoiseSVN может применять подключаемые модули с COM интерфейсом. С такими подключаемыми модулями возможно получать информацию непосредственно из системы отслеживания проблем, взаимодействовать с пользователем и предоставлять информацию о нерешённых проблемах обратно TortoiseSVN, проверять сообщения журнала, вводимые пользователем и даже выполнять действия после успешной фиксации для, например, закрытия проблемы.

Мы не можем предоставить информацию и руководство о том, как вы должны реализовывать COM-объект на вашем любимом языке программирования, но у нас есть образцы подключаемых модулей (plugins) на C ++/ATL и C# в папке `contrib/issue-tracker-plugins` нашего хранилища. В этой папке вы также можете найти требуемые include-файлы для сборки. (Раздел 3, «Лицензия» рассказывает как получить доступ к хранилищу.)

Важно

Вам следует предоставить и 32-битную, и 64-битную версию вашего подключаемого модуля, потому что 64-разрядная версия TortoiseSVN не может использовать 32-х разрядные модули и наоборот.

7.1. Соглашение об именование

Если вы выпускаете модуль для системы отслеживания проблем для TortoiseSVN, то, пожалуйста, *не* называйте его *Tortoise<ЧтоТоТам>*. Мы хотели бы зарезервировать префикс *Tortoise* для клиентов управления версиями, интегрируемых в оболочку Windows. Например: TortoiseCVS, TortoiseSVN, TortoiseHg, TortoiseGit и TortoiseBzr — всё это клиенты управления версиями.

Пожалуйста, назовите ваш плагин для клиента Tortoise *Turtle<ЧтоТоТам>*, где *<ЧтоТоТам>* ссылается на систему отслеживания проблем, к которой вы подключаетесь. Как вариант, выберите наименование, которое звучит как *Turtle*, но имеет другую первую букву. Вот хороший пример:

- Gurtle - плагин для отслеживания проблем в Google Code
- TurtleMine - плагин для отслеживания ошибок в Redmine
- VurtleOne - плагин для отслеживания ошибок в VersionOne

7.2. Интерфейс IBugtraqProvider

Версии TortoiseSVN 1.5 и выше могут использовать подключаемые модули, реализующие интерфейс IBugtraqProvider. Интерфейс предоставляет несколько методов, которые модули могут использовать для взаимодействия с системой отслеживания проблем.

```
HRESULT ValidateParameters (
  // Родительское окно для любого пользовательского интерфейса,
  // который необходимо показывать при проверке.
  [in] HWND hParentWnd,

  // Строка параметров, которая должна быть проверена.
  [in] BSTR parameters,

  // Строка допустима?
  [out, retval] VARIANT_BOOL *valid
);
```

Этот метод вызывается из диалога настроек, где пользователь может добавить и настроить подключаемый модуль. Строка `parameters` может быть использована модулем для получения дополнительной необходимой информации, такой как URL системы отслеживания проблем, учётные данные для подключения и т.п. Модуль должен проверить строку `parameters` и показать окно ошибки в случае неправильной строки. Параметр `hParentWnd` должен быть использован как родительское окно для любого диалога, показываемого модулем. Модуль должен возвращать TRUE если проверка строки `parameters` прошла успешно. Если модуль возвращает FALSE, диалог настроек не позволит пользователю добавить модуль к пути в рабочей копии.

```
HRESULT GetLinkText (
  // Родительское окно для любого пользовательского интерфейса,
  // который необходимо показать (в основном для ошибок).
  [in] HWND hParentWnd,

  // Строка параметров, на случай если вам необходимо узнать у вашей
  // веб-службы (например), каков правильный текст.
  [in] BSTR parameters,

  // Какой текст вы желаете показать?
  // Используйте локаль текущего потока.
  [out, retval] BSTR *linkText
);
```

Модуль может предоставить здесь строку, которая используется в диалоге фиксации TortoiseSVN для кнопки вызова модуля, например "Выберите проблему" или "Выберите тикет". Убедитесь, что строка не очень длинная, иначе она может не поместиться на кнопку. Если метод возвращает ошибку (например, `E_NOTIMPL`), для кнопки используется текст по умолчанию.

```
HRESULT GetCommitMessage (
  // Родительское окно для пользовательского интерфейса модуля
  [in] HWND hParentWnd,

  // Параметры для вашего модуля.
  [in] BSTR parameters,
  [in] BSTR commonRoot,
  [in] SAFEARRAY(BSTR) pathList,

  // Текст, уже содержащийся в сообщении фиксации.
  // Ваш модуль должен включить этот текст в новое сообщение,
  // если необходимо.
  [in] BSTR originalMessage,

  // Новый текст для сообщения фиксации.
  // Заменяет исходное сообщение.
  [out, retval] BSTR *newMessage
);
```

Это главный метод подключаемого модуля. Этот метод вызывается диалогом фиксации TortoiseSVN, когда пользователь нажимает на кнопку модуля.

Строка `parameters` - это строка, которую пользователь должен ввести в диалоге настроек, когда он конфигурирует подключаемый модуль. Обычно модуль использует её для поиска URL адреса системы отслеживания ошибок и/или информации для подключения и проч.

Строка `commonRoot` содержит родительский путь всех элеметов, выбранных для показа в диалоге фиксации. Обратите внимание, что это *не*путь всех элементов, которые выбрал пользователь в диалоге фиксации. Для диалогов ответвлений/меток этот путь должен быть скопирован.

Параметр `pathList` содержит массив путей (в виде строк), которые пользователь выбрал для фиксации.

Параметр `originalMessage` содержит текст, введенный в окно сообщений журнала в диалоге фиксации. Если пользователь еще не ввёл никакого текста, эта строка будет пустой.

Параметр `newMessage` возвращает строку, скопированную в поле ввода сообщения журнала в диалоге фиксации поверх существовавшей там записи. Если подключаемый модуль не изменяет строку `originalMessage`, он должен возвращать ту же строку, иначе любой введенный пользователем текст будет потерян.

7.3. Интерфейс IBugtraqProvider2

В TortoiseSVN 1.6 был добавлен новый интерфейс, предоставляющий расширенную функциональность для подключаемых модулей. Это интерфейс IBugtraqProvider2, наследующий от IBugtraqProvider.

```
HRESULT GetCommitMessage2 (
  // Родительское окно для пользовательского интерфейса модуля
  [in] HWND hParentWnd,

  // Параметры для вашего модуля.
  [in] BSTR parameters,
  // Общий URL фиксации
  [in] BSTR commonURL,
  [in] BSTR commonRoot,
  [in] SAFEARRAY(BSTR) pathList,

  // Текст, уже содержащийся в сообщении фиксации.
  // Ваш модуль должен включить этот текст в новое сообщение,
  // если необходимо.
  [in] BSTR originalMessage,

  // Содержимое поля bugID (если показано)
  [in] BSTR bugID,

  // Изменённое содержимое поля bugID
  [out] BSTR * bugIDOut,

  // Вы можете назначать собственные свойства ревизии для фиксации
  // путём установки следующих двух параметров.
  // Внимание: Оба массива должны быть одного размера.
  // Для каждого имени свойства должно иметься значение!

  // Список имён свойств ревизии.
  [out] SAFEARRAY(BSTR) * revPropNames,

  // Список значений свойств ревизии.
  [out] SAFEARRAY(BSTR) * revPropValues,

  // Новый текст для сообщения фиксации.
  // Заменяет исходное сообщение.
  [out, retval] BSTR * newMessage
);
```

Этот метод вызывается из диалога фиксации TortoiseSVN, когда пользователь нажимает на кнопку подключаемого модуля. Этот метод вызывается вместо `GetCommitMessage()` и здесь используются те же параметры. Параметры описаны в документации к `GetCommitMessage`.

Параметр `commonURL` - это родительский URL для всех элементов, выбранных для показа в диалоге фиксации. Это по существу URL адрес `commonRoot`.

В параметре `bugID` находится содержимое поля идентификатора ошибки bug-ID (при отображении параметр настраивается свойством `bugtraq:message`).

Значение параметра `bugIDOut` используется для заполнения поля идентификатора ошибки bug-ID, когда метод завершает работу.

Значения параметров `revPropNames` и `revPropValues` могут содержать пары имя/значение для свойств ревизии устанавливаемых при фиксации. Подключаемый модуль должен удостовериться, что оба массива при возвращении имеют одинаковый размер! Также каждое имя свойства в `revPropNames` должно иметь соответствующее значение в `revPropValues`. Если никакие свойства ревизии не установлены, подключаемый модуль должен вернуть пустой массив.

```
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
);
```

Этот метод вызывается непосредственно перед закрытием диалога фиксации и началом фиксации. Модуль может использовать этот метод для проверки выбранных для фиксации файлов/папок и/или сообщения фиксации, введённого пользователем. Параметры те же, что и для `GetCommitMessage2()`, но с отличаютсятем, что `commonURL` теперь общий URL всех *отмеченных* элементов, и `commonRoot` - корневой путь всех отмеченных элементов.

`commonURL` - это источник URL адреса копии для диалога ответвления/метки, и `commonRoot` - задает целевой URL для копии.

Возвращаемый параметр `errorMessage` должен содержать или сообщение об ошибке, которое TortoiseSVN покажет пользователю, или должен быть пустым для начала фиксации. Если возвращается сообщение об ошибке, TortoiseSVN показывает это сообщение в диалоге и оставляет диалог фиксации открытым, чтобы пользователь мог исправить ошибку. Поэтому модуль должен возвращать сообщение об ошибке, информирующее пользователя, *что* именно не так и как это исправить.

```
HRESULT  OnCommitFinished (
  // Родительское окно для любого пользовательского интерфейса,
  // который необходимо показать (в основном для ошибок).
  [in] HWND hParentWnd,

  // Общий корень всех фиксируемых путей.
  [in] BSTR commonRoot,

  // Все фиксируемые пути.
  [in] SAFEARRAY(BSTR) pathList,


  // Текст, уже содержащийся в сообщении фиксации.
  [in] BSTR logMessage,

  // Ревизия фиксации.
  [in] ULONG revision,


  // Ошибка, которую необходимо показать пользователю,
```

```
  // если эта функция возвращает что-то, отличное от S_OK
  [out, retval] BSTR * error
);
```

Этот метод вызывается после успешной фиксации. Модуль может использовать этот метод, например, для закрытия выбранной проблемы или для добавления информации о фиксации к проблеме. Параметры те же, что и для `GetCommitMessage2`.

```
HRESULT HasOptions(
  // Предоставляет ли модуль возможности по настройке?
  [out, retval] VARIANT_BOOL *ret
);
```

Этот метод вызывается из диалога настроек, в котором пользователь может настраивать подключаемые модули. Если модуль предоставляет свой собственный диалог параметров посредством `ShowOptionsDialog`, он должен вернуть здесь TRUE, иначе он должен возвращать FALSE.

```
HRESULT ShowOptionsDialog(
  // Родительское окно для диалога настроек
  [in] HWND hParentWnd,

  // Параметры для вашего модуля.
  [in] BSTR parameters,

  // Строка параметров
  [out, retval] BSTR * newparameters
);
```

Этот метод вызывается из диалога настроек, когда пользователь нажимает на кнопку "Options". Она отображается, если `HasOptions` возвращиетTRUE. Подключаемый модуль может отображать диалог настроек для упрощения настройки модуля пользователем.

Строка `parameters` содержит уже установленные/введенные строковые параметры модуля.

Возвращаемый параметр `newparameters` должен содержать строку параметров, которую модуль составляет на основании информации, собранной в собственном диалоге настройки параметров. Эта строка `paramameters` передаётся всем остальным методам IBugtraqProvider и IBugtraqProvider2.

Приложение A. Часто задаваемые вопросы (ЧаВо, FAQ)

Because TortoiseSVN is being developed all the time it is sometimes hard to keep the documentation completely up to date. We maintain an *online FAQ* [https://tortoisesvn.net/faq.html] which contains a selection of the questions we are asked the most on the TortoiseSVN mailing lists <dev@tortoisesvn.tigris.org> and <users@tortoisesvn.tigris.org>.

Мы также поддерживаем *Issue Tracker* [https://sourceforge.net/p/tortoisesvn/tickets/] проекта, в котором Вы можете узнать о некоторых вещах, которые есть в нашем списке To-Do и ошибках, которые уже исправлены. Если Вы считаете, что нашли ошибку, или хотите запросить новую функциональность, сначала проверьте здесь — может, кто-то другой уже сделал это перед Вами.

Если у вас есть вопрос, ответа на который вы не нашли, то лучшее место, где его можно задать - это список рассылки.

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 перетянуть правой кнопкой /action> файлы в новое местоположение. Просто сделайте правый клик не отпуская кнопку мыши на файлах, которые вы хотите переместить/скопировать. Затем перетяните файлы в новое местоположение и отпустите кнопку мыши. Появится контекстное меню, в котором вы можете выбрать Context Menu → SVN Копировать версированные файлы сюда. или Context Menu → 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-ловушка 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.

Для более подробной информации об этих свойствах проекта обратитесь к Раздел 4.18, «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

Безусловно самый простой способ отменить изменения из одной или нескольких ревизий, заключается в использовании диалога журнала ревизий.

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 хотите вернуть. Если вы хотите отменить диапазон ревизий, то выберите первую и, удерживая клавишу **Shift**, выберите последнюю. Если вы хотите выбрать отдельные ревизии и диапазоны, то используйте клавишу **Ctrl** при выборе ревизий. 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

Если вы хотите ввести номера ревизий списком, то используйте диалог Слияние. В предыдущем методе слияние используется негласно, в этом методе оно применяется явно.

1. В вашей рабочей копии выберите TortoiseSVN → Слить...[1].
2. В диалоге Тип слияния выберите Слияние диапазона ревизий.
3. В поле От: введите полный URL-адрес хранилища вашей папки рабочей копии. Это будет URL по умолчанию.
4. В поле Диапазон ревизий для слияния введите список ревизий для отката (или используйте диалог журнала для их выбора, как описано выше).
5. Убедитесь, что установлен флажок Обратное слияние.
6. В диалоге Параметры слияния согласитесь со значениями по умолчанию.
7. Нажмите кнопку Слить для выполнения слияния.

Вы отменили изменения в вашей рабочей копии. Проверьте, что результаты такие, как вы ожидали, а затем зафиксируйте изменения.

[1] теперь слияние производится при помощи мастера, в котором для этой задачи есть специальный пункт Произвести слияние диапазона ревизий. На следующей странице мастера укажите нужный диапазон ревизий (можно использовать журнал) и отметьте флажок Обратное слияние. Подробнее об этом рассказывает Раздел 4.21.1, «Слияние с диапазоном ревизий» - прим. переводчика

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 способ 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/1.8/svn.reposadmin.maint.html] книги о Subversion.

B.5. Сравнить две ревизии файла или папки

Если вы желаете сравнить две ревизии из истории одного элемента, например, ревизии 100 и 200 одного файла, просто вызовите TortoiseSVN → Показать журнал для вывода истории ревизий этого файла. Выберите две ревизии, которые вы желаете сравнить, после чего воспользуйтесь Контекстное меню → Сравнить ревизии.

Если вы желаете сравнить один и тот же элемент в двух различных деревьях, например, в стволе и в ответвлении, вы можете использовать обозреватель хранилища для открытия этих двух деревьев, выбрать файл в обоих местах, а затем воспользоваться Контекстное меню → Сравнить ревизии.

Если вы желаете сравнить два дерева, например, ствол и помеченный выпуск, чтобы посмотреть, что изменилось, вы можете использовать TortoiseSVN → Граф ревизий. Выберите два узла для сравнения, затем используйте Контекстное меню → Сравнить ведущие ревизии. Будет показан список изменённых файлов, и вы сможете затем выбрать отдельные файлы для подробного просмотра изменений. Вы можете также экспортировать все изменённые файлы в виде дерева, или просто список всех изменённых файлов. Прочтите Раздел 4.11.3, «Сравнение папок» для дополнительной информации. Или же можно использовать Контекстное меню → Объединённые различия ведущих ревизий, чтобы увидеть сводку всех изменений с минимальным контекстом.

B.6. Включить общий подпроект

Иногда вы желаете включить другой проект в вашу рабочую копию, возможно, некоторый библиотечный код. Существуют по крайней мере 4 способа, как это организовать.

B.6.1. Используя svn:externals

Установите свойство `svn:externals` для папки в вашем проекте. Это свойство состоит из одной или нескольких строк; каждая строка содержит имя подпапки, которая будет использоваться как папка для извлечения общего кода, и URL-адрес хранилища, из которого будет производиться извлечение. За полным описанием обращайтесь в Раздел 4.19, «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 в нем будут добавлены в список фиксации, когда вы будете фиксировать ваш главный проект.

Если внешний проект находится в другом хранилище, любые изменения, внесенные во внешний проект будут показаны или указаны, когда Вы фиксируете основной проект, но Вам следует 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 основной код в нескольких проектах и не хотите хранить множество рабочих копий этого кода для каждого проекта, который его использует, вы можете извлечь его в отдельное местоположение, которое будет связано со всеми другими проектами. Пример структуры каталогов:

```
C:\Projects\Proj1
C:\Projects\Proj2
C:\Projects\Proj3
C:\Projects\Common
```

Вы можете поместить ваш общий код в папку Common и ссылаться на него из любого из перечисленных проектов с помощью относительного пути, например `..\..\Common\DSPcore`.

Если ваши проекты разбросаны в несвязанных местах, то вы можете вариант этого, который заключается в следующем, разместите общий код в одном месте и используйте подстановку буквы диска для маппирования, чтобы можно быть прописать явно 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"
```

чтобы создать маппирование диска, используемого в вашем исходном коде. После этого вы можете 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.6.4. Добавить проект в хранилище

Возможно, самым простым способом будет просто добавить проект в подпапку в вашей рабочей копии. Однако, у этого способа недостаток в том, что вам придётся вручную обновлять этот внешний проект.

Чтобы помочь с обновлением TortoiseSVN предоставляет команду в контекстном меню при перетаскивании. Просто перетяните правой кнопкой мыши папку, в которую вы развернули новую

версию внешней библиотеки, в папку в вашей рабочей копии, и затем выберите Контекстное меню → SVN Ответвление вендора здесь. Эта операция скопирует новые файлы в целевую папку, при этом автоматически будут добавлены новые файлы и удалены файлы, которых нет в новой версии.

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 его целевой объект в

```
TortoiseProc.exe /command:repobrowser /path:"адрес/вашего/хранилища"
```

Не забудьте укажать реальный URL хранилища вместо показанного выше примера.

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 ещё не зафиксировали добавление, тогда всё что вам надо сделать это TortoiseSVN → Отменить добавление... чтобы отменить добавление. Затем вам надо добавить файл(ы) в список игнорирования, чтобы ещё раз ошибочно не добавить их позже.

Если файлы уже находятся в хранилище, то их нужно удалить оттуда и добавить в список игнорируемых. К счастью, у TortoiseSVN есть удобный ярлык для этого. TortoiseSVN → Разверсировать и добавить к списку игнорируемых сначала пометит файл/папку для удаления из хранилища, сохранив локальную копию. Также это добавит элемент к списку игнорируемых так, что он не будет по ошибке заново добавлен обратно в Subversion. Сделав это однажды, вам нужно лишь зафиксировать родительскую папку.

B.9. Разверсирование рабочей копии

Если у вас есть рабочая копия, которую вы хотели бы преобразовать в обычную папку без каталога `.svn`, вы можете просто экспортировать её в саму себя. Прочтите Раздел 4.27.1, «Выведение рабочей копии из-под управления версиями», чтобы узнать, как это сделать.

B.10. Удаление рабочей копии

Если у вас есть рабочая копия, которая больше не нужна, то как избавится от неё начисто? Просто — удалите её в проводнике Windows! Рабочие копии — это частные локальные объекты, и они самодостаточны. Удаление рабочей копии в проводнике Windows совсем не влияет на данные в хранилище.

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 должен быть установлен от *Имени системы* и не от*Имени пользователя*. Это связано с тем что TortoiseSVN необходимы CRT и MFC DLL файлы, которые могут быть установлены *на компьютер* и не *на пользователя*. Если вам действительно необходимо установить TortoiseSVN on a per user basis, тогда вы должны сперва установить MFC и CRT пакет версии 12 от Microsoft на каждом компьютере, на котором будете устанавливать TortoiseSVN на пользователя.

Вы можете настроить файл MSI так, чтобы все пользователи получили одни и те же настройки. Настройки TSVN сохраняются в ключе реестра `HKEY_CURRENT_USER\Software\TortoiseSVN` и общие настройки Subversion (которые влияют на все клиенты Subversion) сохраняются в файле config в папке `%APPDATA%\Subversion`. Если вам требуется помощь в настройке MSI, то попробуйте поискать на форумах MSI transform или поищите в сети «MSI transform».

C.2. Перенаправление проверки обновлений

TortoiseSVN проверяет есть ли новая версия каждые несколько дней. Если доступна новая версия, то об этом отображается соответствующее уведомление в диалоге фиксации.

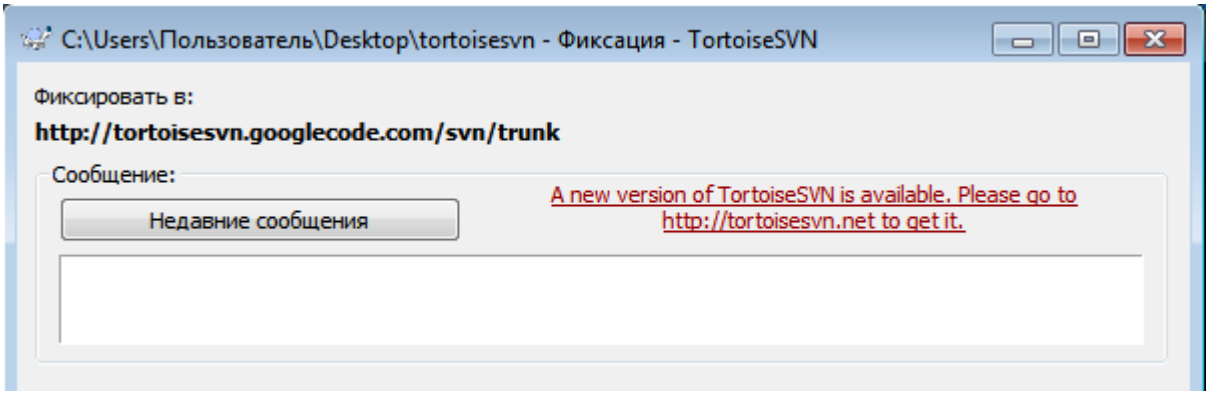


Рисунок C.1. Диалог фиксации показывающий уведомление об обновлении

Если вы отвечаете за большое количество пользователей в вашем домене, то, возможно, вы захотите, чтобы пользователи использовали только утвержденную версию, а не должны были каждый раз устанавливать последнюю. Возможно, вы не хотите, чтобы показывалось уведомление об обновлении и пользователи должны немедленно обновляться.

Версии TortoiseSVN 1.4.0 и позже позволяют вам перенаправить проверку обновления на интранет-сервер. Вы можете установить в ключе реестра `HKCU\Software\TortoiseSVN\UpdateCheckURL` (строковое значение) адрес URL, указывающий на файл в вашем интранет. Этот текстовый файл должен быть следующего формата:

```
1.9.1.6000
```

```
Новая версия TortoiseSVN доступна для скачивания!
http://192.168.2.1/downloads/TortoiseSVN-1.9.1.6000-svn-1.9.1.msi
```

Первая строка в этом файле — версия. Вы должны быть уверены, что версия соответствует версии в пакете инсталляции TortoiseSVN. Вторая строка — это пользовательский текст, показывающийся в диалоге фиксации. Вы можете написать там всё что угодно. Только помните, что место в диалоге фиксации ограничено. Слишком длинные сообщения будут обрезаны! Третья строка — это адрес URL нового пакета инсталляции. Этот адрес URL открывается когда пользователь щелкает на метке пользовательского сообщения в диалоге фиксации. Вы также можете просто направить пользователе на вебстраницу вместо прямой ссылки на файл MSI. Адрес URL открывается в браузере по умолчанию, так что если вы укажете вебстраницу, она будет показана пользователю. Если вы укажете пакет MSI, то браузер спросит пользователя куда локально сохранить MSI пакет.

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 функция всё ещё доступна в TortoiseSVN и других svn-клиентах. Чтобы включить её вы должны установить переменную среды Windows `ASPDOTNETHACK` в `1`. Вообще-то, значение этой переменной среды не важно - если переменная есть, то функция активна.

Важно

Обратите внимание, что этот хак нужен только если вы до сих пор используете VS.NET2002. Все последующие версии Visual Studio *не* требуют активации этого хака! Так что если вы не используете этот древний инструмент, то НЕ ПОЛЬЗУЙТЕСЬ ЭТИМ!

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
0x0000000000000004	Фиксировать
0x0000000000000008	Добавить
0x0000000000000010	Убрать изменения
0x0000000000000020	Очистка

Значение	Пункт меню
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
0x0000000100000000	Объединить все
0x0000000200000000	Разница с предыдущей версией
0x0000000400000000	Вставить
0x0000000800000000	Обновите рабочую копию
0x0000001000000000	Сравнить позже
0x0000002000000000	Сравнить с 'filename'
0x0000004000000000	Объединённые различия
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"`.

Есть специальная команда, которая не требует параметр `/command:abcd` но, если в командной строке ничего не задано, то вместо этого запускается монитор проекта. Если задан параметр `/tray`, то монитор проекта запускается в скрытом режиме и только добавляет свой значок в область на панели задач.

Так как некоторые команды могут принимать список целевых путей (например, фиксация нескольких конкретных файлов) в параметре `/path` можно задать несколько путей, разделённых символом `*`.

You can also specify a file which contains a list of paths, separated by newlines. The file must be in UTF-16 format, without a *BOM* [https://en.wikipedia.org/wiki/Byte-order_mark]. If you pass such a file, use `/pathfile` instead of `/path`. To have TortoiseProc delete that file after the command is finished, you can pass the parameter `/deletepathfile`. If you don't pass `/deletepathfile`, you have to delete the file yourself or the file gets left behind.

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"`. Это переопределит путь по умолчанию, в том числе и установку из реестра.

Для автоматического закрытия окна выполнения по окончанию команды без предварительного постоянного изменения настройки вы можете указать параметр `/closeonend`.

- `/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

Для закрытия окна выполнения для локальных операций при отсутствии ошибок и конфликтов, укажите параметр `/closeforlocal`.

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	Открывает диалог журнала. Параметр `/path` задает файл или папку, для которой должен быть показан журнал. Могут быть указаны дополнительные параметры:
	• `/startrev:xxx`,
	• `/endrev:xxx`,
	• `/strict` включает флажок 'остановиться на копировании',
	• `/merge` включает флажок 'включая слитые ревизии',
	• `/datemin:"{datestring}"` устанавливает начальную дату фильтра и
	• `/datemax:"{datestring}"` устанавливает конечную дату фильтра. Формат даты тот же, которые используется в датах ревизий svn.
	• `/findstring:"filterstring"` заполняет текст фильтра,
	• `/findtext` фильтр будет использовать текст, а не регулярное выражение, или
	• `/findregex` фильтр будет использовать регулярное выражение, а не простой текстовый поиск, и
	• `/findtype:X` где X - число от 0 до 511. Числа - сумма следующих параметров:
	  • `/findtype:0` фильтр по всему
	  • `/findtype:1` фильтр по сообщениям
	  • `/findtype:2` фильтр по пути
	  • `/findtype:4` фильтровать по авторам
	  • `/findtype:8` фильтр по ревизиям
	  • `/findtype:16` не используется
	  • `/findtype:32` фильтр по ID ошибки
	  • `/findtype:64` не используется
	  • `/findtype:128` фильтр по дате
	  • `/findtype:256` фильтр по диапазону дат
	• Если задан `/outfile:path\to\file`, то выбранные ревизии записываются в этот файл при закрытии диалога журнала. Ревизии записываются в том же формате, который используется при указании ревизии в диалоге слияния.
	An svn date revision can be in one of the following formats:

Команда	Описание
	• {2006-02-17} • {15:30} • {15:30:00.200000} • {"2006-02-17 15:30"} • {"2006-02-17 15:30 +0230"} • {2006-02-17T15:30} • {2006-02-17T15:30Z} • {2006-02-17T15:30-04:00} • {20060217T1530} • {20060217T1530Z} • {20060217T1530-0500}
:checkout	Открывает диалог извлечения. `/path` определяет директорию назначения и `/url` определяет адрес URL, из которого извлекается. Если вы укажете ключ `/blockpathadjustments`, путь для автоматического извлечения блокируется. `/revision:XXX` определяет ревизию для извлечения.
:import	Открывает диалог импорта. `/path` определяет директорию с данными для импорта. Вы также можете указать переключатель `/logmsg` для передачи предопределенного сообщения журнала в диалог импорта. Или если вы хотите передать сообщение журнала в командной строке, то используйте `/logmsgfile:путь`, где `путь` указывает на файл с сообщением журнала.
:update	Обновляет рабочую копию в `/path` до ревизии HEAD. Если задан параметр `/rev`, то показывается диалог, спрашивая у пользователя до какой ревизии обновить. Для отключения диалога укажите номер ревизии `/rev:1234`. Другие параметры `/nonrecursive`, `/ignoreexternals` и `/includeexternals`. `/stickydepth` указывает на то, что указанная глубина должна быть постоянной, создавая неполное извлечение. Параметр `/skipprechecks` может быть установлен для пропуска всех проверок, которые были выполнены перед обновлением. Если это указано, то кнопка Журнал недоступна, и контекстное меню для показа различий также недоступно после обновления.
: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 отмененнные операции и разблокирует рабочую копию в `/path`. Вы также должны пропустить `/cleanup`, чтобы действительно выполнить очистку. Используйте `/noui`, чтобы исключить

Команда	Описание
	появление диалога (с сообщением о завершении очистки или об ошибке). /noprogressui также отключает диалог процесса выполнения. /nodlg отключает появление диалога, где пользователь может выбрать, что именно должно быть выполнено при очистке. Доступные действия могут быть указаны в параметрах /cleanup для очистки статуса, /breaklocks для снятия блокировок, /revert для отмены незафиксированных изменений, /delunversioned, /delignored, /refreshshell, /externals, /fixtimestamps и /vacuum.
: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. /path задает целевую директорию, а /url — адрес URL, на который переключиться.
:export	Экспортирует рабочую копию /path в другую директорию. Если /path указывает на неверсированную директорию, то диалог запросит адрес URL, который экспортировать в директорию /path. Если вы указываете ключ /blockpathadjustments, то отключается автоматическая настройка пути экспорта.
:dropexport	Экспортирует рабочую копию /path в директорию указанную в /droptarget. Этот экспорт не использует диалог экспорта и выполняется напрямую. Параметр /overwrite указывает, что существующие файлы перезаписываются без подтверждения пользователя, и параметр /autorename указывает, что если файлы уже существуют, то экспортируемые файлы переименовываются автоматически во избежание их перезаписывания. Параметр /extended может определять либо localchanges для экспорта только файлов изменённых локально, либо unversioned для экспорта также всех неверсированных элементов.
:dropvendor	Copies the folder in /path recursively to the directory specified in /droptarget. New files are added automatically, and missing files get removed in the target working copy, basically ensuring that source and destination are exactly the same. Specify /noui to skip the confirmation dialog, and /noprogressui to also disable showing the progress dialog.
:merge	Открывает диалог слияния. /path определяет директорию назначения. Для слияния диапазона ревизий доступны следующие параметры: /fromurl:URL, /revrange:строка. Для слияния двух деревьев хранилища доступны следующие параметра: /fromurl:URL, /tourl:URL, /fromrev:xxx и /torev:xxx.
:mergeall	Открывает диалог 'Слить все'. Параметр /path задаёт целевую папку.
:copy	Открывает диалог ответвления/метки. /path это рабочая копия для ответвления/метки. А /url — URL назначения. Если URL начинается с ^, предполагается, что он относится к корню хранилища. Чтобы сразу отметить опцию Переключить рабочую копию на новую ветку/метку. вы можете передать параметр /switchaftercopy. Чтобы отметить опцию Создавать промежуточные папки передайте параметр /makeparents. Вы можете также указать параметр /logmsg, чтобы передать предопределенное сообщение журнала в диалог ответвления/метки. Или, если вы не хотите передавать сообщение журнала в командной строке используйте /logmsgfile:путь, где путь указывает на файл содержащий сообщение журнала.
:settings	Открывает диалог настроек.

Команда	Описание
: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 внешнюю программу сравнения, указанную в настройках TortoiseSVN. Параметр `/path` задает первый файл. Если параметр `/path2` указан, то программа сравнения запускается с этими двумя файлами. Если не указан параметр `/path2`, то файл `/path` сравнивается с его Базовой версией. Если у указанного файла изменились свойства, то внешняя программа сравнения запускается для каждого измененного свойства. Чтобы предотвратить это - укажите параметр `/ignoreprops`. Чтобы указать номера конкретных ревизий используйте параметр `/startrev:xxx` и `/endrev:xxx`, и для указания стержневой (peg) ревизии используйте параметр `/pegrevision:xxx`. Если параметр `/blame` указан, а `/path2` - нет, тогда сравнение авторства выполнится для указанных ревизий первого файла. Параметр `/line:xxx` задает строку, на которую будет перемещен курсор в окне с результатами сравнения.
: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. Если у указанного URL также изменились свойства, то внешнее средство сравнения будет запущено для каждого измененного свойства. Чтобы предотвратить это - укажите параметр `/ignoreprops`.
: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 наличия изменений. Параметр `/path` задаёт папку рабочей копии. Если указан параметр `/remote`, то диалог связывается с хранилищем при начале работы, как если бы пользователь нажал кнопку Проверить хранилище.
:repobrowser	Запускает диалог браузера хранилища указывающего на адрес URL рабочей копии, заданной в `/path` или `/path` напрямую указывает адрес URL. Дополнительный параметр `/rev:xxx` может использоваться для определения ревизии, которая должна быть показана в браузере хранилища. Если параметр `/rev:xxx` пропущен, то по умолчанию будет HEAD ревизия. Если `/path` указывает на адрес URL, параметр `/projectpropertiespath:path/to/wc` определяет путь, из которого читать и использовать свойства проекта. Если задан `/outfile:path\to\file`, то выбранный адрес URL и ревизия записывается в этот файл при закрытии обозревателя хранилища. Первая

Команда	Описание
	строка в этом файле содержит адрес URL, вторая строка — ревизию в текстовом формате.
: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	Создает патч-файл для пути заданного в `/path`. Чтобы пропустить диалог "Сохранить Как" вы можете передать `/savepath:путь`, чтобы явно указать путь сохранения патч-файла. Чтобы избежать запуска программы просмотра объединённых различий патч-файла, передайте `/noview`.
:revisiongraph	Показывает граф ревизий для пути заданного в `/path`. Для создания файла с изображением графа ревизии для опеределенного пути, но без показа окна графа, передайте `/output:path` с путем к выходному файлу. Выходной файл должен иметь расширение, чтобы граф ревизий действительно экспортировался. Вот эти расширения: literal>.svg С тех пор как граф ревизий имеет много параметров, которые влияют на отображение, вы также можете установить параметры при создании выходного файла изображения. Передавайте эти параметры в `/options:XXXX`, где `XXXX` — десятичное значение. Лучший способ найти требуемые параметры — запустить граф ревизий обычным способом, установить все параметры пользовательского интерфейса и закрыть граф. Затем параметры, которые вам необходимо передать в командной строке могут быть прочитаны из реестра `HKCU\Software\TortoiseSVN\RevisionGraphOptions`.
: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 свойств для пути заданного в `/path`. Для работы с версированными свойствами этой команде нужна рабочая копия. Свойства ревизии могут быть просмотрены/изменены если `/path` является адресом URL и указано `/rev:XXX`.

Команда	Описание
	Чтобы непосредственно открыть диалог для определённого свойства, передайте имя свойства в параметре `/property:name`.
:sync	Экспортирует/импортирует настройки, либо в зависимости от текущих настроек или если экспортированные настройки новее, или как указано. Если передан путь как `/path`, путь используется для сохранения или чтения настроек. Параметр `/askforpath` покажет пользователю диалог открытия/сохранения, чтобы выбрать путь экспорта/импорта. Если ни параметр `/load`, ни параметр `/save` не заданы, то TortoiseSVN определяет экспортировать или импортировать настройки по тому какие из них более поздние. Если файл экспорта более поздний, чем текущие настройки, то настройки загружаются из файла. Если текущие настройки более поздние, то настройки экспортируются в файл настроек. Если задан параметр `/load`, то настройки импортируются из файла настроек. Если задан параметр `/save`, то текущие настройки экспортируются в файл настроек. Параметр `/local` включает локальные настройки в экспорт, т.е. настройки, которые ссылаются на локальные пути.

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. Обработчик Tsvncmd для URL

Возможен также вызов TortoiseProc с веб-страницы при использовании специальных URL.

TortoiseSVN регистрирует новый протокол `tsvncmd:`, который может быть использован для создания гиперссылок, которые выполняет команды TortoiseSVN. Команды и параметры те же самые, как при запуске TortoiseSVN из командной строки.

Формат URL для `tsvncmd:` выглядит следующим образом:

```
tsvncmd:command:cmd?parameter:paramvalue?parameter:paramvalue
```

где `cmd` является одной из разрешенных команд, `parameter` является названием параметра, таким как `path` или `revision`, и `paramvalue` является значением этого параметра. Список разрешенных параметров зависит от используемой команды.

Доступны следующие команды для `tsvncmd:` URL:

- `:update`

- `:commit`
- `:diff`
- `:repobrowser`
- `:checkout`
- `:export`
- `:blame`
- `:repostatus`
- `:revisiongraph`
- `:showcompare`
- `:log`

Простой пример адреса URL может выглядеть так:

```
<a href="tsvncmd:command:update?path:c:\svn_wc?rev:1234">Обновить</a>
```

или в более сложном случае:

```
<a href="tsvncmd:command:showcompare?
url1:https://svn.code.sf.net/p/stefanstools/code/trunk/StExBar/src/setup/Setup.wxs?
url2:https://svn.code.sf.net/p/stefanstools/code/trunk/StExBar/src/setup/Setup.wxs?
revision1:188?revision2:189">compare</a>
```

D.3. 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
```

D.4. Комманды TortoiseUDiff

Программа просмотра унифицированных различий имеет только два параметра командной строки:

Параметр	Описание
:patchfile	Путь к файлу унифицированных различий.
:p	Активирует режим канала. Унифицированные различия читаются из консольного ввода.

Таблица D.3. Список доступных параметров

Примеры (должны быть введены в одной строке):

```
TortoiseUDiff.exe /patchfile:"c:\diff.patch"
```

Если вы создали различия другой командой, то вы можете использовать TortoiseUDiff для показа этих различий напрямую:

```
svn diff | TortoiseUDiff.exe /u
```

это также сработает, если вы пропустите параметр `/p`:

```
svn diff | TortoiseUDiff.exe
```

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.

> **Примечание**
>
> 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 адрес расположения хранилища показывается просто как `URL`, и в качестве примера такого URL может служить `https://svn.code.sf.net/p/tortoisesvn/code/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 [-depth ARG] [--ignore-externals] [-r rev] URL ПУТЬ
```

The depth combo box items relate to the `-depth` argument.

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] [-depth ARG] [--ignore-externals] ПУТЬ
```

The depth combo box items relate to the `-depth` argument.

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 "LogMessage" [-depth ARG] [--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 "LockMessage" [--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 PATH
svn log -r N:M PATH
```

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.

Оверлейный значок *Необходима блокировка* обозначает, что установлено свойство `svn:needs-lock`.

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.

В действительности, вы можете обнаружить, что не все из этих значков используются в вашей системе. Это потому, что количество разрешенных оверлейных значков в Windows ограничено 15. Windows использует 4 из них, а остальные 11 могут быть использованы другими приложениями. Если больше нет свободных слотов для оверлеев, то TortoiseSVN старается быть *Good Citizen (TM)* и ограничивает свое использование оверлеев и дает шанс другим приложениям.

С тех пор как существуют клиенты Tortoise для других систем управления, мы создали общий компонент, который отвечает за отображение оверлейных значков. Технические детали сейчас несущественны, всё что вам необходимо знать это то, что этот общий компонент позволяет всем клиентам Tortoise использовать одни и те же оверлейные значки и таким образом ограничение в 11 доступных слотов не достигается при установке более чем одного клиента Tortoise. Конечно, есть один небольшой недостаток: все клиенты Tortoise используют те же оверлейные значки, так что вы не можете понять какую систему управления версиями использует рабочая копия.

- *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.
- *Заблокирован* загружается если возможно, но сбрасывается в *Нормальный* если слотов недостаточно.
- *Добавлен* загружается если возможно, но сбрасывается в *Изменен* если слотов недостаточно.

Приложение G. Языковые пакеты и проверка правописания

Стандартный установочный файл поддерживает только английский язык, но вы можете загрузить отдельные языковые пакеты и словари для проверки орфографии после установки.

G.1. Языковые пакеты

The TortoiseSVN user interface has been translated into many different languages, so you may be able to download a language pack to suit your needs. You can find the language packs on our *translation status page* [https://tortoisesvn.net/translation_status_dev.html]. And if there is no language pack available, why not join the team and submit your own translation ;-)

Каждый пакет локализации упакован в инсталлятор `.msi`. Просто запустите программу установки и следуйте инструкциям. После завершения установки перевод будет доступен.

The documentation has also been translated into several different languages. You can download translated manuals from the *support page* [https://tortoisesvn.net/support.html] on our website.

G.2. Проверка правописания

TortoiseSVN uses the Windows spell checker if it's available (Windows 8 or later). Which means that if you want the spell checker to work in a different language than the default OS language, you have to install the spell checker module in the Windows settings (`Settings > Time & Language > Region & Language`).

TortoiseSVN will use that spell checker if properly configured with the `tsvn:projectlanguage` project property.

In case the Windows spell checker is not available, TortoiseSVN can also use spell checker dictionaries from *OpenOffice* [https://openoffice.org] and *Mozilla* [https://mozilla.org].

Инсталлятор автоматически добавляем словари для US и UK English. Если вы хотите другие языки, то самый простой способ установить один из пакетов локализации TortoiseSVN. Будет установлены соответствующие файлы словарей, также как и локализованный пользовательский интерфейс TortoiseSVN. После завершения установки будет доступен и словарь.

Или вы можете установить словари самостоятельно. Если у вас установлен OpenOffice или Mozilla, вы можете скопировать эти словари, размещенные в папках установки этих приложений. Или же вам надо скачать требуемые словари 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 папку `%APPDATA%\TortoiseSVN\dic`. Если такой папки нет, то вы сначала вы должны создать её. TortoiseSVN также будет искать папку `Languages` в папке установки TortoiseSVN (обычно это `C:\Program Files\TortoiseSVN\Languages`); это место куда языковые пакеты складывают свои файлы. Тем не менее, папка %APPDATA% не требует администраторских привилегий и, таким образом, имеет более высокий приоритет. 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 4.18, «Установки проекта».

2. Если язык проекта не задан, или этот язык не установлен, попробовать язык, соответствующий локализации Windows.

3. Если точная Windows локаль не работает, то попробуйте язык «Base», например `de_CH` (Swiss-German) вернется в `de_DE` (German).

4. Если ничего из этого не сработало, тогда язык по умолчанию - английский, включённый в стандартную установку.

Глоссарий

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

Символы

C

F

G

I

M

S

T

U

V

W