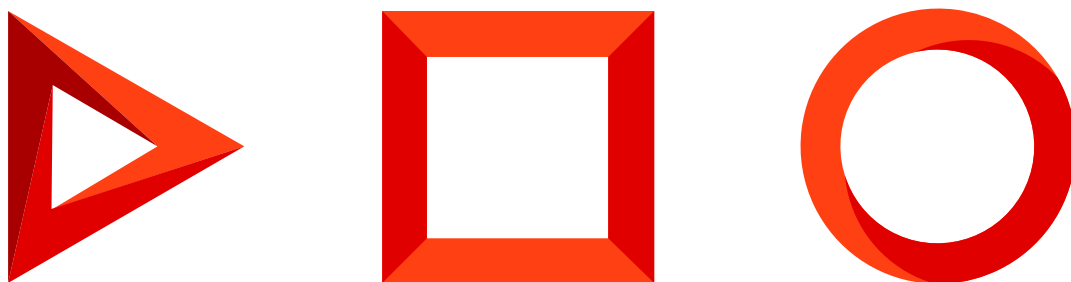


Системы контроля версий

Контроль версий в Subversion

Версия 8.0



Эта документация предоставляется с ограничениями на использование и защищена законами об интеллектуальной собственности. За исключением случаев, прямо разрешенных в вашем лицензионном соглашении или разрешенных законом, вы не можете использовать, копировать, воспроизводить, переводить, транслировать, изменять, лицензировать, передавать, распространять, демонстрировать, выполнять, публиковать или отображать любую часть в любой форме или посредством любые значения. Обратный инжиниринг, дизассемблирование или декомпиляция этой документации, если это не требуется по закону для взаимодействия, запрещены.

Информация, содержащаяся в данном документе, может быть изменена без предварительного уведомления и не может гарантировать отсутствие ошибок. Если вы обнаружите какие-либо ошибки, сообщите нам о них в письменной форме.

Содержание

Контроль версий в Subversion	4
Основные понятия	4
Модели версионирования	5
Работа с файлами в SVN	7
Рабочая копия, используемая приложением Creatio	7
Клиентское приложение для работы с SVN	7
Создать пакет в режиме разработки в файловой системе	8
1. Создать пакет в приложении	8
2. Выгрузить созданный пакет в файловую систему	9
3. Создать необходимые каталоги для пакета в хранилище SVN	10
4. Создать рабочую копию версионной ветки пакета	12
5. Зафиксировать в хранилище каталог пакета	13
Установить существующий пакет из SVN в режиме разработки в файловой системе	14
Последовательность установки пакета из хранилища SVN в режиме разработки в файловой системе	14
Алгоритм реализации примера	15
Привязать к SVN не связанный с хранилищем пакет	20
Последовательность привязки существующего пакета к хранилищу	20
Алгоритм реализации примера	22
Альтернативный вариант реализации примера	28
Обновить и зафиксировать пакет в SVN в режиме разработки в файловой системе	30
1. Обновить пакет из хранилища SVN	30
2. Изменить содержимое пакета	31
3. Зафиксировать пакет в хранилище	32
Создать пакет при переходе в режим разработки в файловой системе	33
Последовательность создания пакета при переходе в режим разработки в файловой системе	33
Последовательность реализации примера	34

Контроль версий в Subversion



Легкий

Creatio позволяет использовать любые системы контроля версий. В этой статье мы рассмотрим применение наиболее популярной из них — Subversion (SVN).

Subversion (SVN) — это бесплатная система управления версиями с открытым исходным кодом.

Основа SVN — хранилище, которое содержит данные в форме иерархии файлов и каталогов — т. н. дерева файлов.

Возможные **действия** пользователей с хранилищем SVN:

- **Чтение данных** других пользователей системы, к которым они предоставили доступ:
 - Чтение файлов других пользователей системы и дерева каталогов.
 - Чтение дерева каталогов.
 - Просмотр предыдущих версий файлов и дерева каталогов.
- **Изменение данных:**
 - Создание новых каталогов и файлов.
 - Переименование каталогов и файлов.
 - Изменение содержимого файлов.
 - Удаление каталогов и файлов.
- **Запись данных** для предоставления доступа другим пользователям системы.

В одном из нижеприведенных случаев система контроля версий SVN **рекомендуется к использованию** для:

- Приложений Creatio на платформе **.NET Framework**.
- Приложений, в которых разработка ведется, в основном, low-code инструментами.
- Cloud-приложений.
Для on-site приложений рекомендуется использовать Git. Работа с системой контроля версий Git описана в статье [Контроль версий в Git](#).

Инструкция по настройке и использованию SVN содержится в [документации SVN](#).

Основные понятия

Хранилище — центральная база данных, обычно расположенная на файловом сервере и содержащая файлы со своей историей версий. Хранилище может быть доступно посредством различных сетевых протоколов или с локального диска.

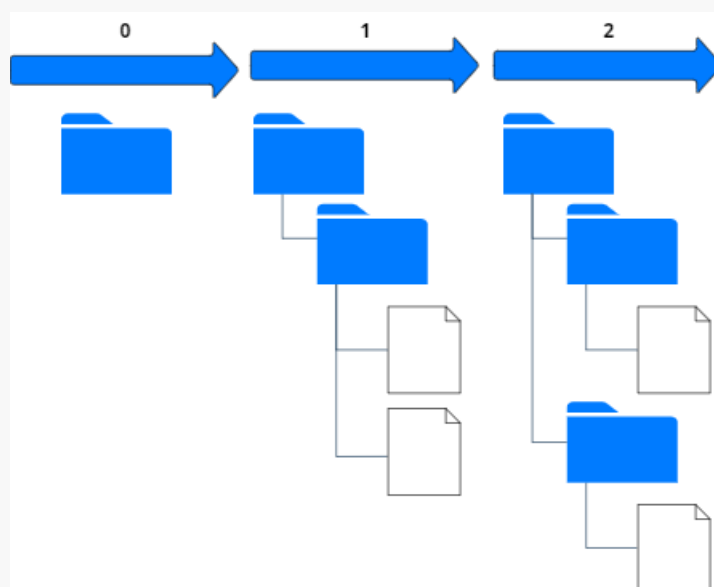
Рабочая копия — каталог на локальном компьютере, с которым работает пользователь. Рабочая копия содержит копию файлов, которые были в хранилище до того, как их начал изменять пользователь. Таким образом можно узнать, какие конкретно изменения были выполнены.

Важно. Изменения можно просмотреть только для текстовых файлов. Для бинарных файлов можно узнать только сам факт изменения.

Ревизия — состояние дерева файловой системы. Ревизия подразумевает весь набор изменений файлов и каталогов как единое изменение.

Фиксация изменений дерева файловой системы — это атомарная операция, которая позволяет зафиксировать ревизию.

Ревизии в хранилище можно представить в виде **серии деревьев файловой системы** — массива номеров ревизий, начинающегося с 0 и растущего слева направо. Под каждым номером расположено дерево файловой системы — "снимок" состояния хранилища после фиксации.



На заметку. В отличие от других систем управления версиями, номера ревизий в SVN относятся к деревьям целиком, а не к отдельным файлам.

Общая **последовательность работы** с файлами в рабочей копии:

1. Получение из хранилища последней версии файлов.
2. Локальная работа с файлами.
3. Фиксация файлов в хранилище.

Модели версионирования

При работе с SVN может возникнуть ситуация, когда разработчики работают над одной и той же функциональностью, реализованной в одном и том же файле. Если первый разработчик сохранит свои изменения первым, а второй — несколькими секундами позже, то изменения, внесенные первым разработчиком, могут быть затерты. И хотя эти изменения содержатся в хранилище, правки, внесенные

первым разработчиком, будут отсутствовать в последней ревизии файла. Чтобы избежать подобной проблемы, используются **модели версионирования**.

Типы моделей версионирования:

- Модель "Блокирование-Изменение-Разблокирование".
- Модель "Копирование-Изменение-Слияние".

Модель "Блокирование-Изменение-Разблокирование"

Хранилище разрешает вносить изменения в файл только одному пользователю за раз. До того как первый пользователь сможет внести изменения в файл, он должен сначала заблокировать этот файл. Второй пользователь не сможет зафиксировать изменения до тех пор, пока первый не внесет изменения в хранилище и не снимет блокировку.

Особенности модели:

- **Блокирование может вызвать проблемы администрирования.**

Первый разработчик может забыть снять блокировку, что приведет к потере времени вторым разработчиком.

- **Блокирование может вызвать излишнюю пошаговость.**

Если разработчики работают с непересекающимися частями файла, то можно было бы работать с файлом одновременно, предполагая корректное слияние изменений.

- **Блокирование может вызвать ложное чувство безопасности.**

Разработчики могут одновременно работать с разными файлами, содержащими зависящую друг от друга функциональность. Каждый разработчик заблокировал свой файл и считает, что начинает безопасную изолированную задачу. Это препятствует заблаговременному обсуждению изменений, которые могут быть несовместимы друг с другом, что приведет к неработоспособности разрабатываемого решения.

Эту модель необходимо использовать, если выполняется работа над файлами, не поддающимися слиянию. Например, если хранилище содержит изображения, и пользователи изменяют их в одно и то же время, то нет возможности выполнить слияние эти изменения.

Модель "Копирование-Изменение-Слияние"

Клиентское приложение каждого пользователя считывает из хранилища проект и создает персональную **рабочую копию** — локальную копию файлов и каталогов хранилища. После этого пользователи работают, одновременно изменяя свои личные копии. В результате работ, личные копии сливаются в новую, финальную версию. Обычно SVN выполняет слияние автоматически, но выполнение слияния необходимо подтвердить.

Если при одновременной работе двух пользователей изменения пересекаются, то возникает **конфликт**.

Чтобы **разрешить конфликт** необходимо:

1. Обсудить изменения, которые вызвали конфликт, с пользователем, выполняющим предыдущую фиксацию файлов.
2. Вручную выбрать, какие изменения из набора конфликтующих изменений необходимо зафиксировать.

3. Зафиксировать в хранилище объединенный файл.

Решающим фактором при использовании этой модели является взаимодействие между пользователями.

Работа с файлами в SVN

В служебный каталог `.svn` рабочей копии для каждого файла SVN записывает свойства.

Свойства файла:

- Рабочая ревизия файла — номер ревизии, на которой основан файл в рабочей копии.
- Дата и время последнего обновления локальной копии файла из хранилища.

Назначение свойств — определить состояние файла рабочей копии.

Состояния файла рабочей копии:

- **Не изменялся и не устарел.**

В хранилище не фиксировались изменения файла со времени его рабочей ревизии. При попытке его обновить или зафиксировать не будет выполнено никаких действий.

- **Изменен локально и не устарел.**

В хранилище не фиксировались изменения этого файла со времени его базовой ревизии. Обновление выполняться не будет. Фиксация в хранилище выполнится успешно.

- **Не изменялся и устарел.**

Файл в рабочей папке не изменялся, но был изменен в хранилище. Файл необходимо обновить для соответствия текущей публичной ревизии. Фиксация выполняться не будет. Обновление выполнится успешно.

- **Изменен локально и устарел.**

Файл был изменен как в рабочей папке, так и в хранилище. Попытка фиксации потерпит неудачу. Файл необходимо сначала обновить, попытавшись объединить опубликованные другим разработчиком изменения с локальными. Если SVN не сможет выполнить объединение самостоятельно, решение конфликта будет выполнять пользователь.

Рабочая копия, используемая приложением Creatio

В Creatio по умолчанию включен режим работы с SVN. При выключенном режиме [разработки в файловой системе](#) приложение Creatio использует собственную рабочую копию каждого пользовательского пакета, для которого подключена версияность. Эти рабочие копии размещаются в каталоге, указанном в элементе `defPackagesWorkingCopyPath` конфигурационного файла `ConnectionStrings.config`.

Если включен режим разработки в файловой системе, то рабочая копия может быть [создана вручную](#) в каталоге `[Путь к приложению]\Terrasoft.WebApp\Terrasoft.Configuration\Pkg\НазваниеПакета`.

Клиентское приложение для работы с SVN

Для работы с SVN в файловой системе рекомендуется использовать клиентское приложение [TortoiseSVN](#) версии не ниже 1.9. Оно реализовано как расширение оболочки Windows и встраивается в контекстное меню проводника Windows. Использование TortoiseSVN описано в [документации по TortoiseSVN](#).

Создать пакет в режиме разработки в файловой системе



Если не предполагается разработка с использованием SVN, то при включенном режиме разработки в файловой системе [последовательность создания пакета](#) ничем не отличается от обычного режима.

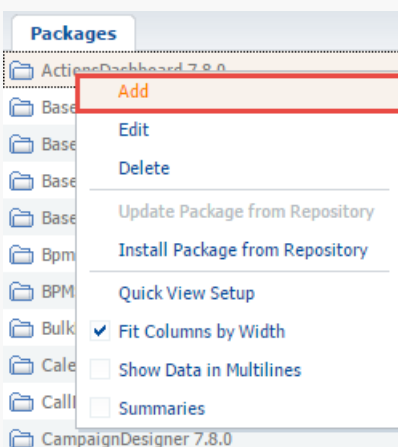
Важно. В Creatio по умолчанию включен режим работы с SVN. Однако, если при создании пакета не заполнять поле [*Хранилище системы контроля версий*], то пакет не будет привязан к хранилищу. Вести версиюнную разработку этого пакета можно будет только подключив его вручную из файловой системы.

При включенном [режиме разработки в файловой системе](#) механизм интеграции с системой хранения версий (SVN) отключен. Встроенными средствами можно только [установить](#) либо [обновить](#) пакеты из хранилища SVN. Поэтому рекомендуется создавать пакет с помощью встроенных средств, а привязывать его к хранилищу с помощью сторонних утилит, например, [TortoiseSVN](#).

Важно. Прежде чем привязывать пакет к хранилищу SVN при включенном режиме разработки в файловой системе необходимо удостовериться, что приложение настроено для доступа к нужному хранилищу SVN.

1. Создать пакет в приложении

На вкладке [*Пакеты*] ([*Packages*]) раздела [*Конфигурация*] ([*Configuration*]) в контекстном меню выбрать действие [*Добавить*] ([*Add*])).



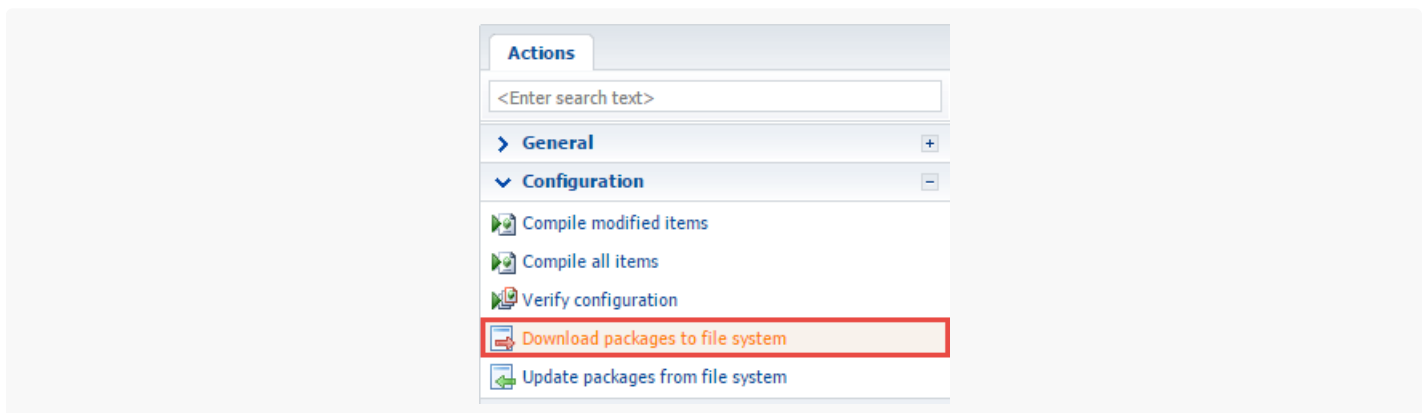
В появившейся карточке пакета заполнить [основные поля свойств пакета](#). Необходимо указать название репозитория, к которому будет привязан пакет.

Важно. Название репозитория в карточке пакета указывает только на то, что пакет будет создан сторонними средствами в этом репозитории. Это позволит в дальнейшем выполнять обновление пакета из раздела [Конфигурация].

Карточка пакета

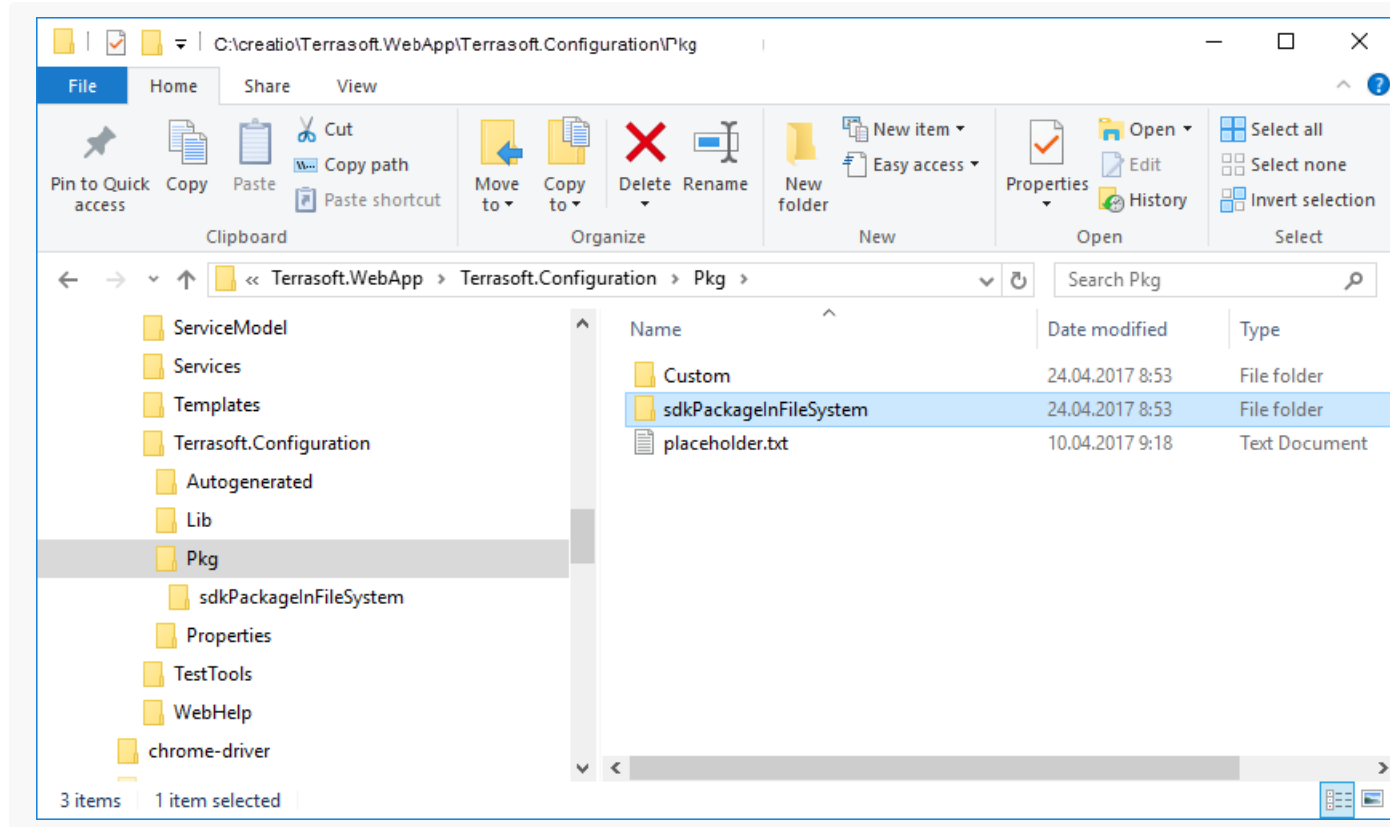
2. Выгрузить созданный пакет в файловую систему

Выполнить действие [*Выгрузить пакеты в файловую систему*] ([*Download packages to file system*]).



В результате пустой пакет будет выгружен в каталог

Путь к установленному приложению\Terrasoft.WebApp\Terrasoft.Configuration\Pkg\sdkPackageInFileSystem.

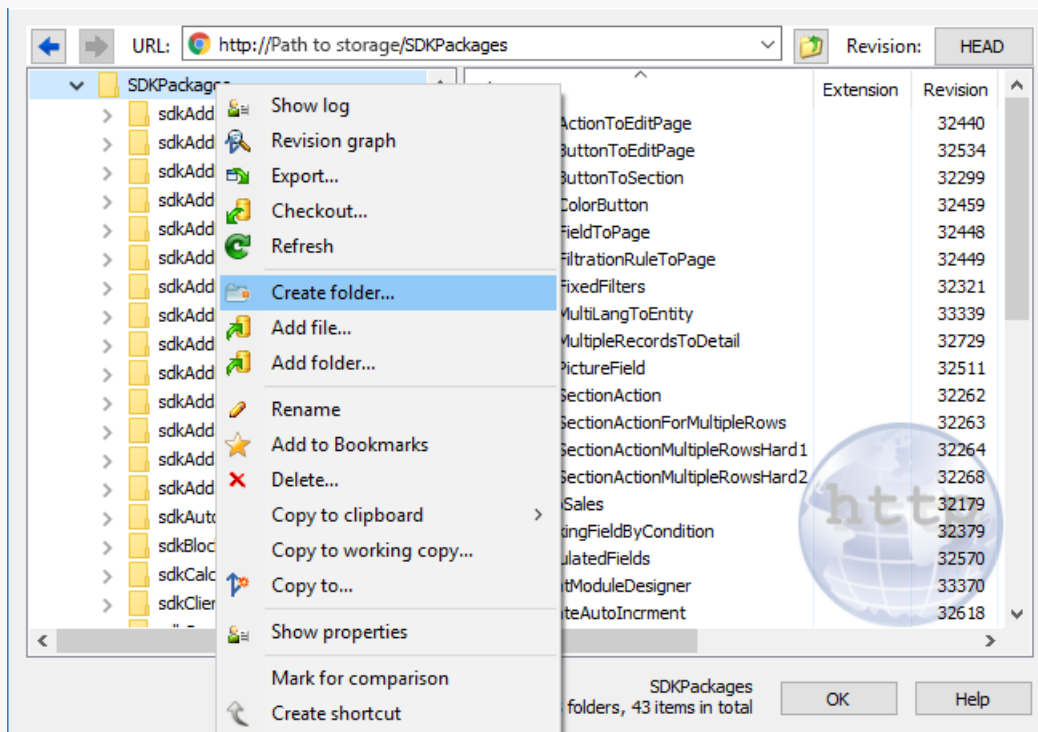


На заметку. Несмотря на то, что в карточке пакета было указано хранилище SVN, при включенном режиме разработки в файловой системе пакет нужно добавлять в хранилище вручную.

3. Создать необходимые каталоги для пакета в хранилище SVN

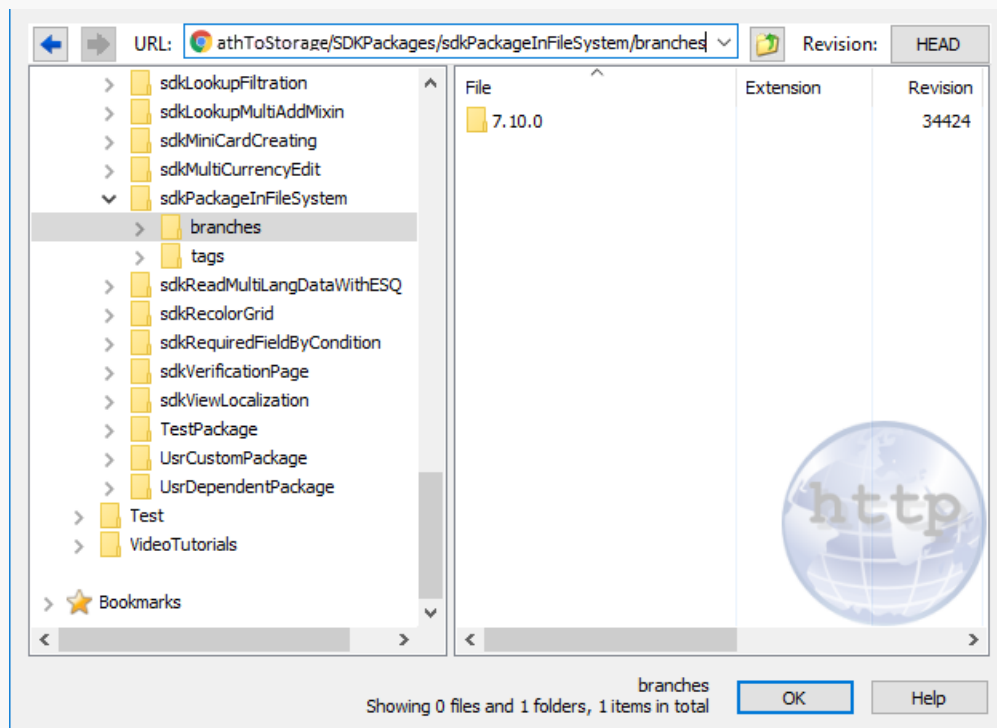
Чтобы создать каталоги для пакета, используя клиентское приложение для работы с SVN (например, [TortoiseSvn](#)), необходимо перейти в репозиторий, указанный в карточке пакета. Затем в репозитории создать каталог, название которого совпадает с названием созданного в приложении пакета.

Важно. В данной статье пример работы с SVN с помощью TortoiseSvn рассмотрен сокращенно. Подробные инструкции о работе с хранилищем с SVN при помощи TortoiseSvn доступны в [документации TortoiseSvn](#).



В созданном каталоге необходимо создать подкаталоги `branches` и `tags`, т.е. повторить [плоскую структуру пакетов](#) Creatio. В завершение в каталоге `branches` необходимо создать каталог, название которого совпадает с номером версии пакета — `7.10.0`.

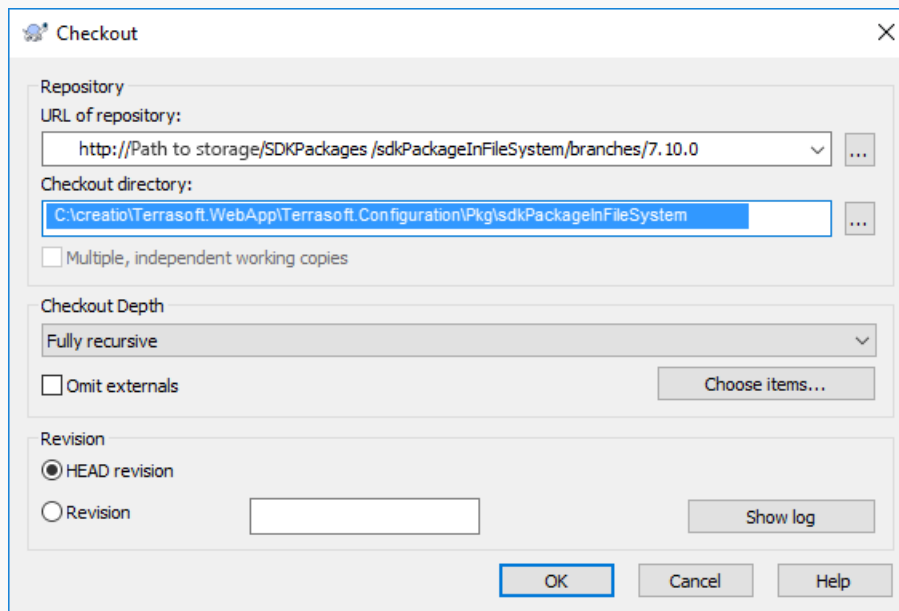
Плоская структура пакета в хранилище



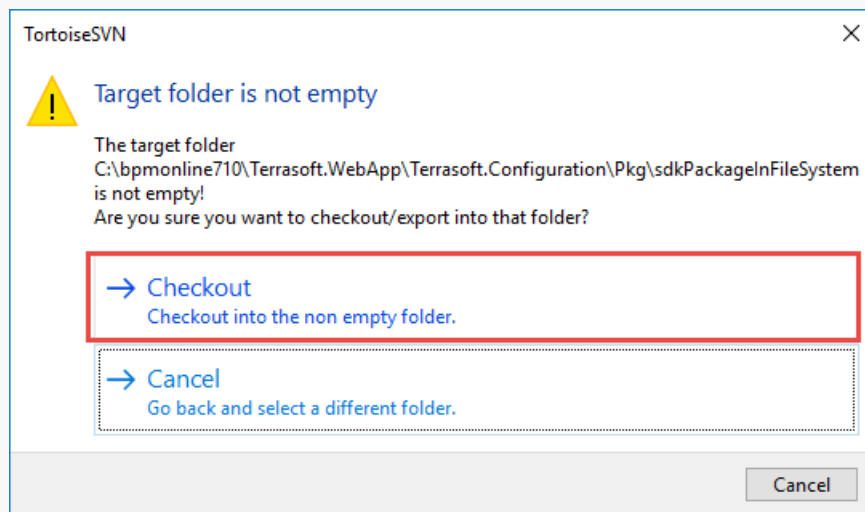
4. Создать рабочую копию версионной ветки пакета

Чтобы создать рабочую копию версионной ветки пакета необходимо выполнить выгрузку (checkout) из хранилища каталога, имя которого совпадает с номером версии пакета, в каталог пакета в файловой системе и подтвердить выгрузку в существующий каталог.

Выгрузка из хранилища рабочей копии версионной ветки пакета



Подтверждение выгрузки в существующий каталог



В результате каталог пакета в файловой системе

Путь к установленному приложению\Terrasoft.WebApp\Terrasoft.Configuration\Pkg\sdkPackageInFileSystem СТАНЕТ связанным с веткой версии 7.10.0 пакета в хранилище.

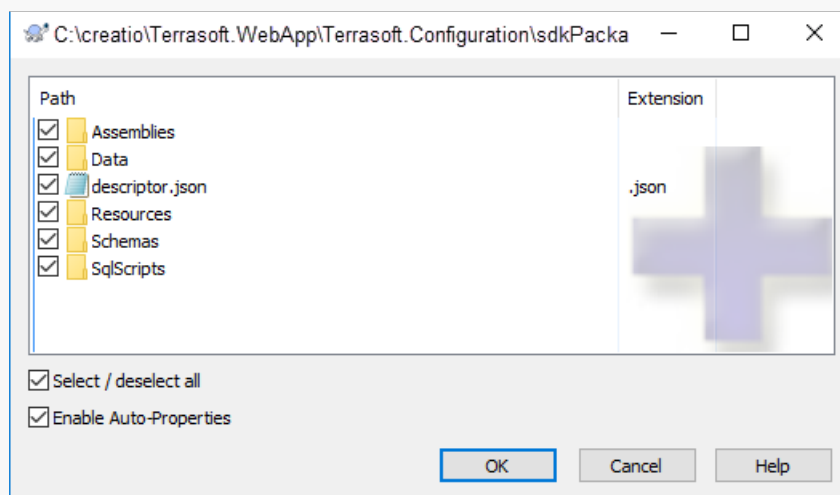
Name	Date modified	Type
Custom	24.04.2017 8:53	File folder
sdkPackageInFileSystem	24.04.2017 10:31	File folder
placeholder.txt	10.04.2017 9:18	Text Document

5. Зафиксировать в хранилище каталог пакета

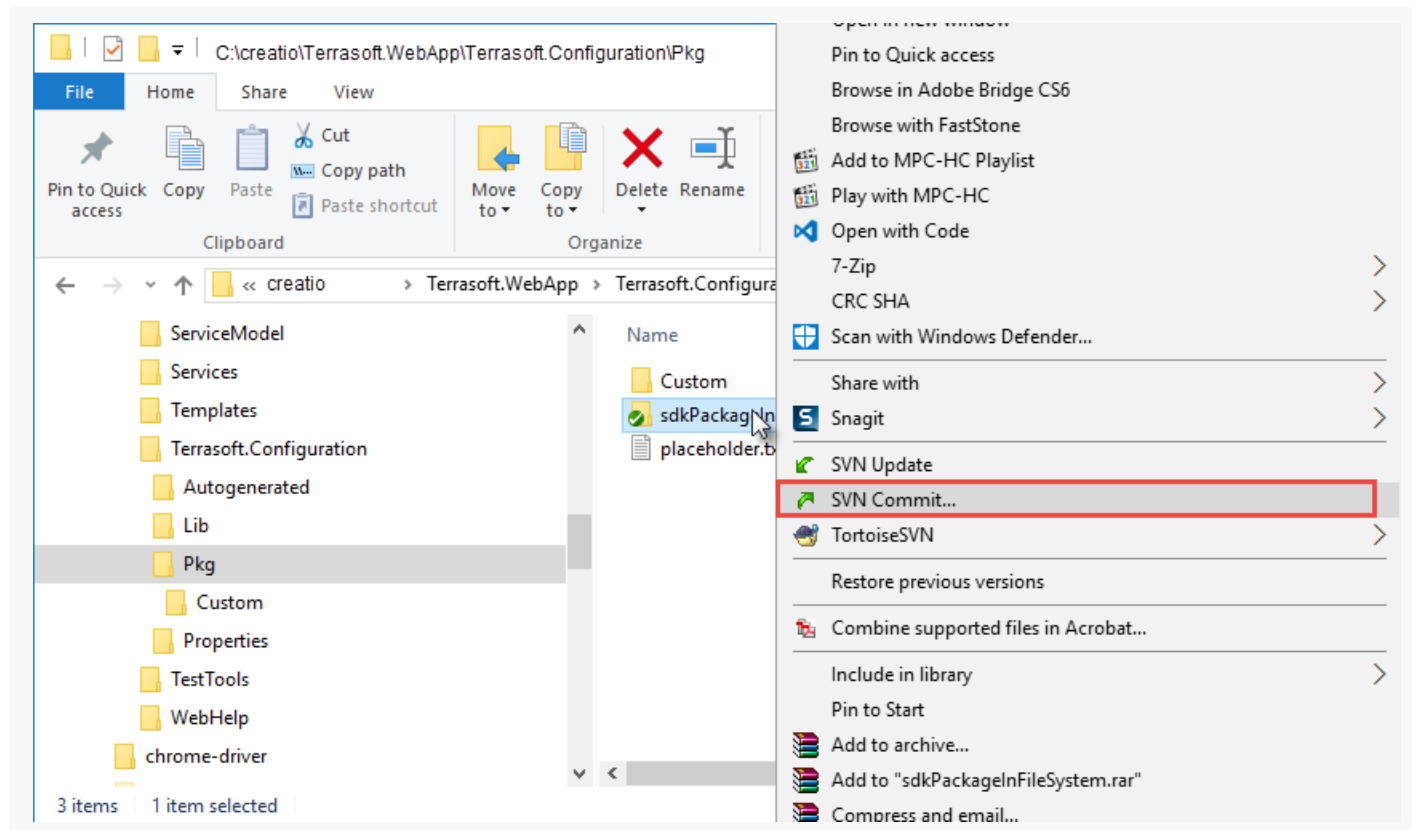
Чтобы зафиксировать каталог пакета необходимо добавить в хранилище все содержимое

Путь к установленному приложению\Terrasoft.WebApp\Terrasoft.Configuration\Pkg\sdkPackageInFileSystem и выполнить фиксацию.

Добавление в хранилище



Фиксация в хранилище



Установить существующий пакет из SVN в режиме разработки в файловой системе

 Средний

Последовательность установки пакета из хранилища SVN в режиме разработки в файловой системе

1. Установить пакет в файловую систему.
2. Установить пакет в приложение.
3. Выполнить генерацию исходных кодов.
4. Скомпилировать изменения.
5. Обновить структуру базы данных.
6. При необходимости установить SQL-сценарии и привязанные данные.

Важно. Чтобы необходимые изменения применились после установки пакета автоматически, нужно включить механизмы автоматического применения изменений. Для этого необходимо в файле `..\Terrasoft.WebApp\Web.config` установить значение `true` для следующих ключей элемента `appSettings`.

```
..\Terrasoft.WebApp\Web.config
```

```
<add key="AutoUpdateOnCommit" value="true" />
<add key="AutoUpdateDBStructure" value="true" />
<add key="AutoInstallSqlScript" value="true" />
<add key="AutoInstallPackageData" value="true" />
```

Ключ `AutoUpdateOnCommit` отвечает за автоматическое обновление пакетов из SVN перед их заливкой. Если для этого ключа установлено значение `false`, то перед заливкой в SVN приложение предупредит пользователя о необходимости обновления в случае, если схемы пакета были изменены. Ключи `AutoUpdateDBStructure`, `AutoInstallSqlScript`, `AutoInstallPackageData` отвечают соответственно за автоматическое обновление структуры базы данных, автоматическую установку SQL-сценариев и установку привязанных данных.

После включения режима автоматического применения изменений выполнять шаги 3—6 нет необходимости. Включить этот режим необходимо до установки пакета.

Важно. Если нужно взаимодействовать с хранилищем SVN как из раздела [*Конфигурация*], так и из файловой системы, то необходимо:

1. [Установить](#) пакет из хранилища в разделе [*Конфигурация*].
2. Выгрузить пакет в файловую систему по действию [*Выгрузить пакеты в файловую систему*] ([*Download packages to file system*]).

Далее следует повторить шаги 3—6 приведенной выше последовательности установки пакета.

Пример. В приложение Creatio в режиме разработки в файловой системе установить пакет из хранилища SVN.

URL пакета в хранилище SVN

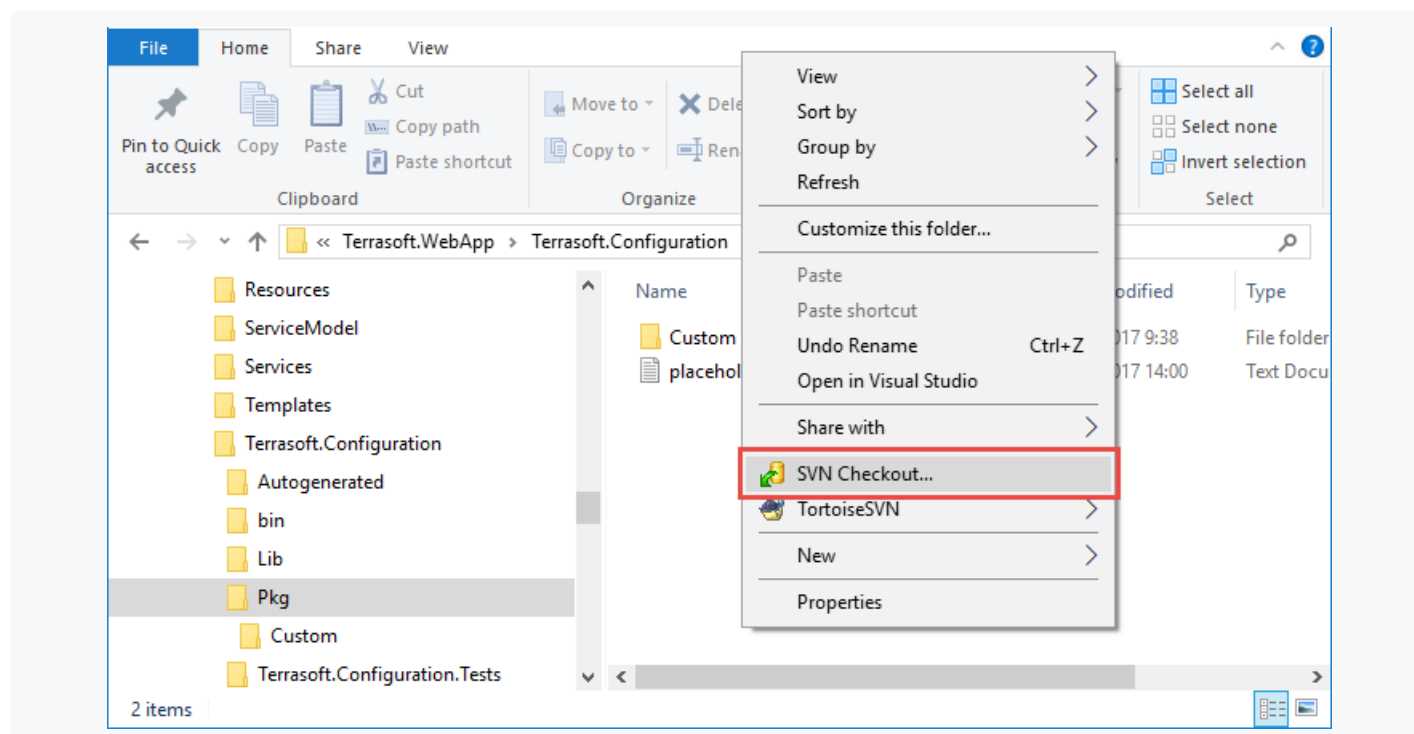
```
http://svn-server:8050/SDKPackages/sdkCreateDetailWithEditableGrid/branches/7.8.0
```

На заметку. В пакете, рассматриваемом в данном примере, содержится функциональность [детали с редактируемым реестром](#).

Алгоритм реализации примера

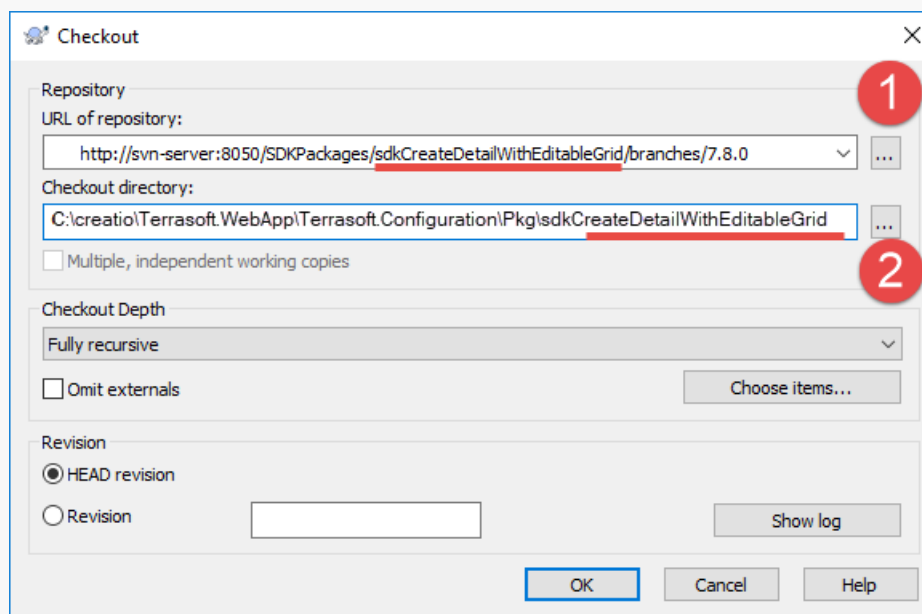
1. Установить пакет в файловую систему

В каталоге приложения `...\Terrasoft.WebApp\Terrasoft.Configuration\Pkg`, открытом в Проводнике, необходимо выполнить действие [*SVN Checkout*].



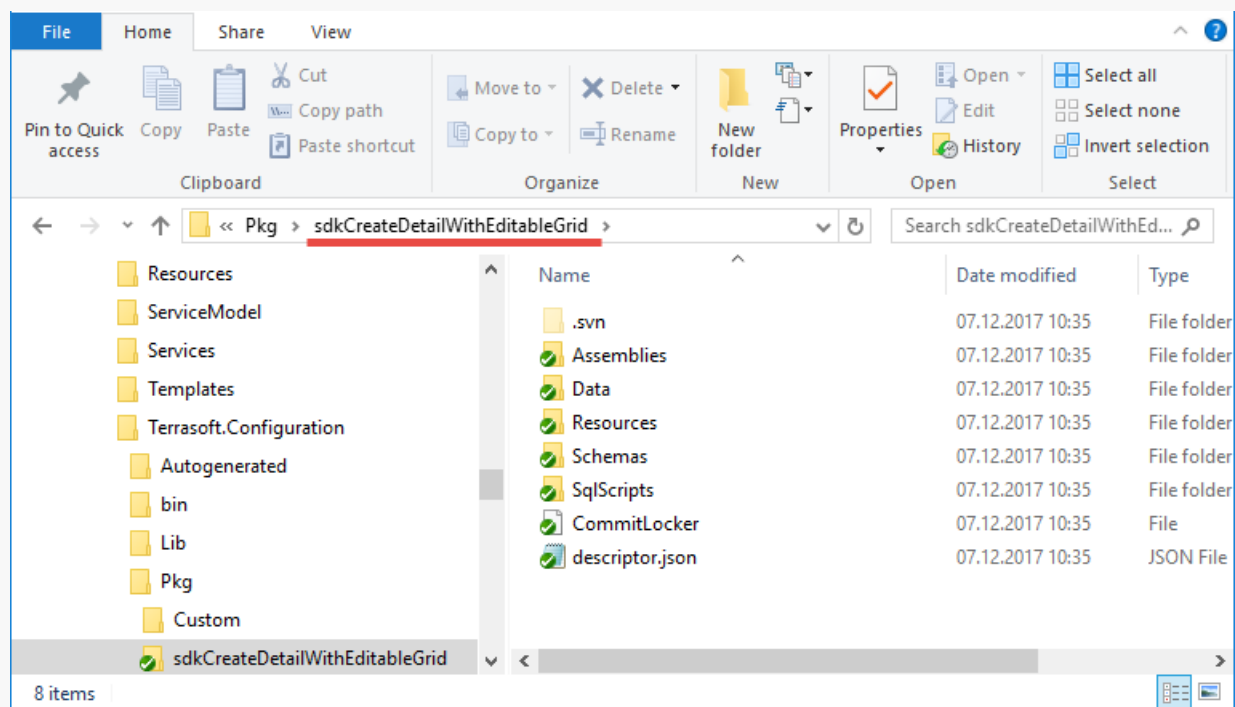
На заметку. Для работы с Subversion (SVN) в файловой системе рекомендуется использовать клиентское приложение [TortoiseSVN](#) версии не ниже 1.8. Оно реализовано как расширение оболочки Windows и встраивается в контекстное меню Проводника. Подробная документация по использованию TortoiseSVN доступна [здесь](#).

В открывшемся диалоговом окне необходимо указать адрес хранилища, по которому размещено содержимое пакета (1), и каталог для выгрузки содержимого пакета.



Важно. Название каталога для выгрузки содержимого пакета должно совпадать с названием пакета.

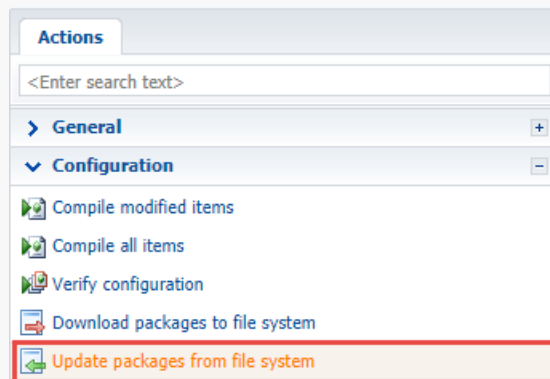
После успешной выгрузки в каталоге `..\Terrasoft.WebApp\Terrasoft.Configuration\Pkg` будет создана рабочая копия пакета.



2. Установить пакет в приложение

Для установки пакета из файловой системы в приложение в разделе [*Конфигурация*] ([*Configuration*])

необходимо выполнить действие [*Обновить пакеты из файловой системы*] ([*Update packages from file system*]).

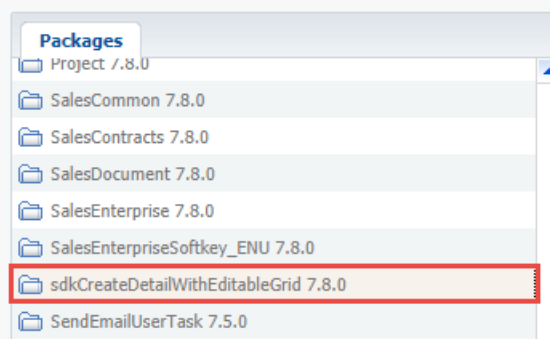


В результате пакет будет добавлен в приложение.

Информационное окно статуса загрузки пакета в приложение

Name	Type	Status
[-] sdkCreateDetailWithEditableGrid	Package	Added
[-] OrderPageV2	Schema	Added
[-] UsrCourierService	Schema	Added
[-] UsrCourierServiceDetail	Schema	Added

Пакет на вкладке [*Пакеты*] ([*Packages*])



Важно. Отсутствие названия репозитория в названии пакета свидетельствует о том, что все изменения могут быть зафиксированы в репозитории только из файловой системы.

3. Выполнить генерацию исходных кодов

Для этого в разделе [*Конфигурация*] ([*Configuration*]) необходимо выполнить действие [*Сгенерировать для требующих генерации*] ([*Generate where it is needed*]).

4. Скомпилировать изменения

Чтобы скомпилировать изменения, необходимо выполнить действие [*Компилировать измененное*] ([*Compile modified items*]).

На заметку. Необходимость действий обновления структуры базы данных, установки SQL-скриптов и привязанных данных отображается в колонках [*Требуется обновление БД*] и [*Требуется установки в БД*] дизайнеров. В случае возникновения ошибок после этих действий, текст последней ошибки можно увидеть в колонке [*Текст последней ошибки*].

Не все эти колонки отображаются по умолчанию в реестре вкладок [*Схемы*], [*SQL сценарии*] и [*Данные*] раздела [*Конфигурация*]. При необходимости их можно добавить при помощи команды контекстного меню [*Настроить колонки*].

5. Обновить структуру базы данных

После компиляции изменений нужно обновить структуру базы данных действием [*Обновить для требующих обновления*] ([*Update where it is needed*]).

6. Установить SQL-сценарии и привязанные данные (при необходимости)

Если пакет содержит привязанные SQL-сценарии или данные, то необходимо выполнить соответствующие действия для их выполнения или установки.

После успешной установки в приложении станет доступной реализованная в пакете функциональность. В приведенном примере это функциональность детали с редактируемым реестром.

ORD-1 (sample) What can I do for you? Creatio

CLOSE ACTIONS VIEW

Status 1. Draft Payment amount, \$ 0.00

< PRODUCTS ORDER DETAILS DELIVERY SUMMARY HISTORY GENERAL INFO >

Delivery type Courier Payment type Non-cash payment

Delivery address

Recipient information

Courier Service + :

Order Account

ORD-1 (sample) Accom (sample) Enter a value

На заметку. Для отображения примененных изменений может понадобиться обновление страницы с очисткой кэша.

Привязать к SVN не связанный с хранилищем пакет

Средний

Важно. Привязать не связанный с хранилищем существующий пакет к SVN можно только в приложении, установленном on-site. Работать с таким пакетом можно только в режиме разработки в файловой системе.

На заметку. После привязки пакета к SVN в файловой системе его можно [установить](#) в другое приложение, используя встроенные средства Creatio.

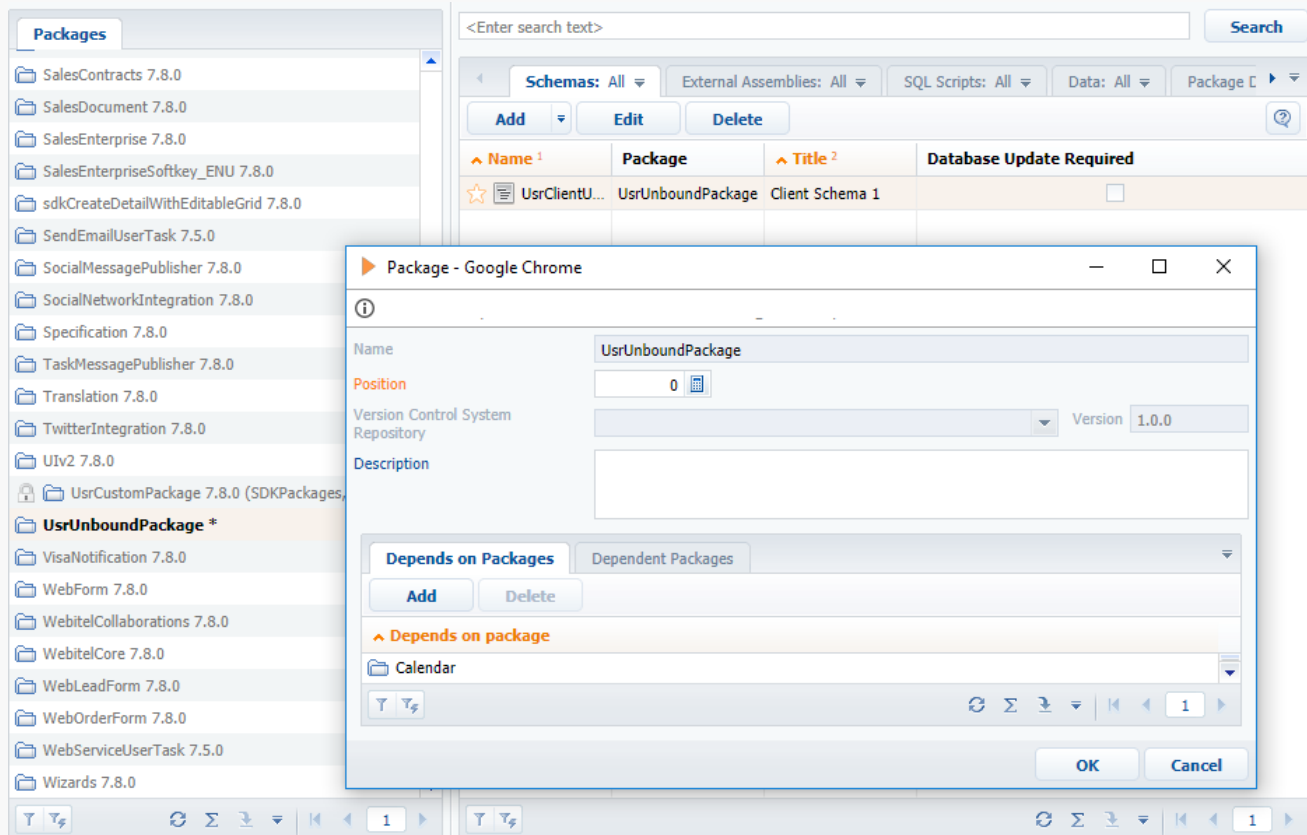
Последовательность привязки существующего пакета к хранилищу

1. Перейти в режим разработки в файловой системе.
2. Выгрузить пакет в файловую систему.
3. Создать необходимые каталоги для пакета в хранилище SVN.
4. Создать рабочую копию версионной ветки пакета.
5. Зафиксировать в хранилище каталог пакета.

Альтернативный вариант привязки пакета к хранилищу — через прямой запрос в базу данных. Для этого необходимо:

1. В разделе [*Конфигурация*] добавить в приложение информацию о хранилище SVN.
2. Привязать хранилище к пакету. Для этого:
 - В таблице `SysRepository` считать идентификатор записи, содержащей адрес нужного хранилища SVN.
 - В таблице `SysPackage` в запись, содержащую имя не привязанного к хранилищу SVN пакета, в колонку [`SysRepositoryId`] добавить полученный идентификатор.

Пример. Привязать к хранилищу не связанный с SVN существующий пользовательский пакет `UsrUnboundPackage`.



URL для доступа к хранилищу SVN

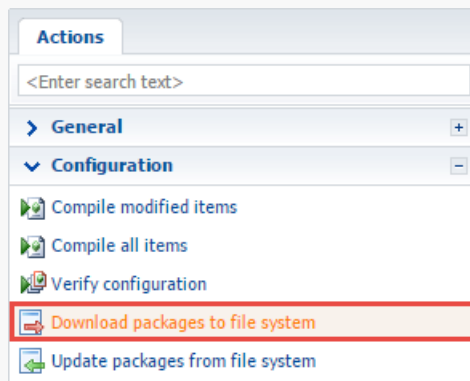
```
http://svn-server:8050/SDKPackages
```

Алгоритм реализации примера

1. Перейти в [режим разработки в файловой системе](#)

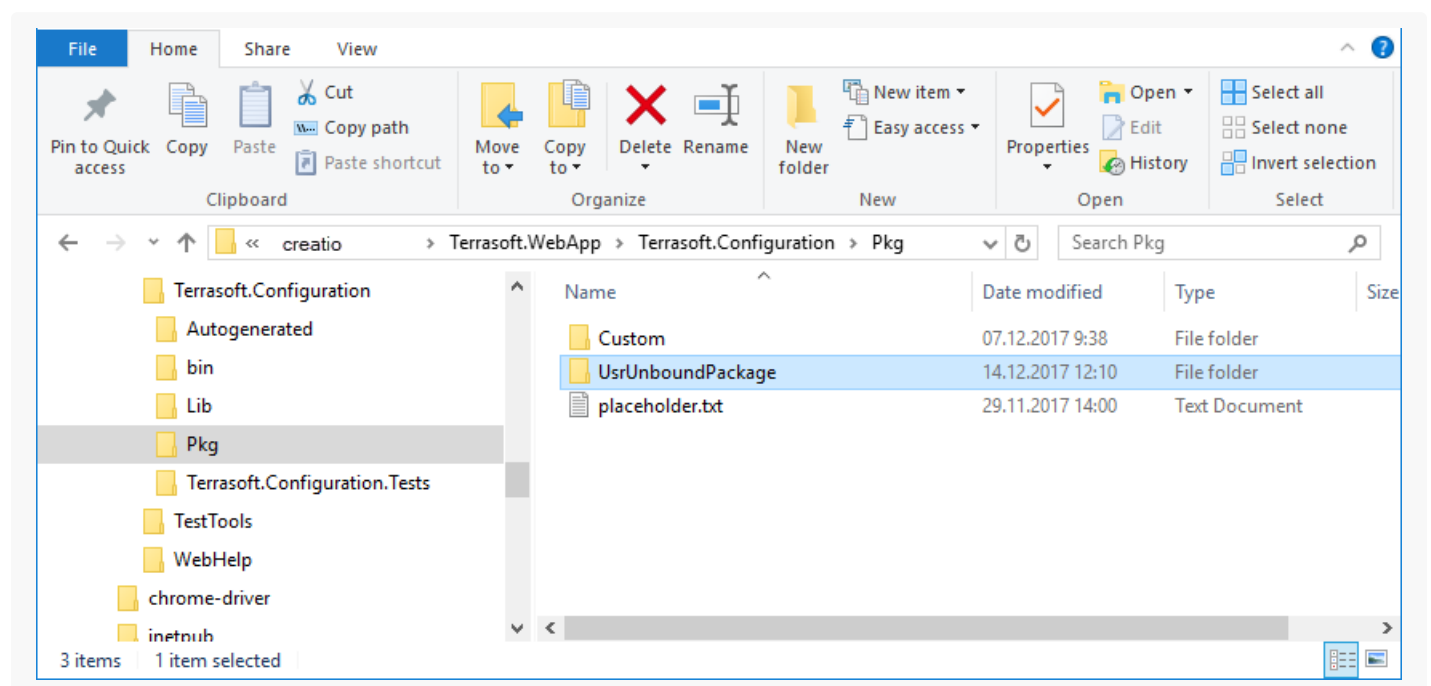
2. Выгрузить пакет в файловую систему

В разделе [Конфигурация] ([Configuration]) выполнить действие [Выгрузить пакеты в файловую систему] ([Download packages to file system]).



В результате пакет будет выгружен в каталог

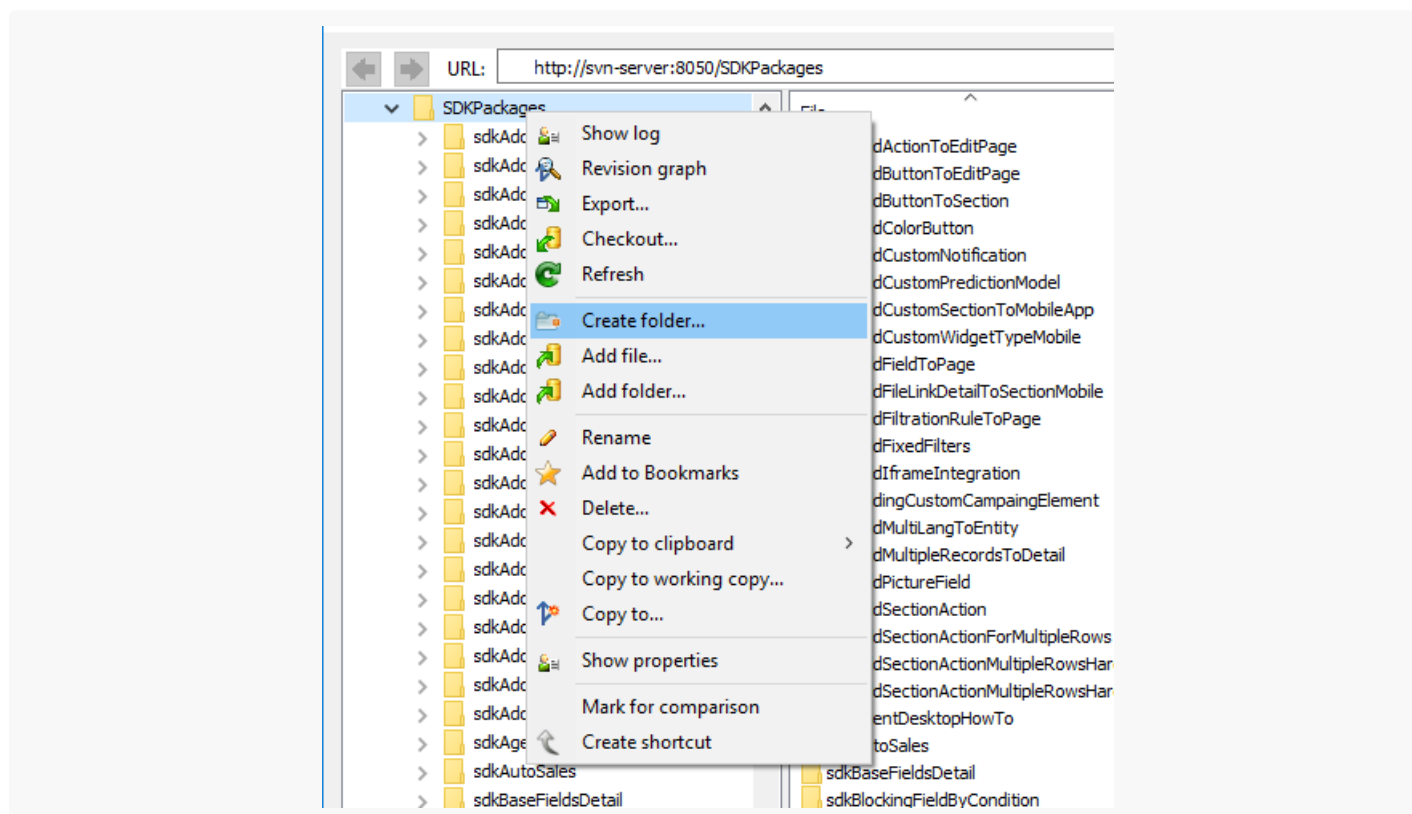
```
..\Terrasoft.WebApp\Terrasoft.Configuration\Pkg\UsrUnboundPackage .
```



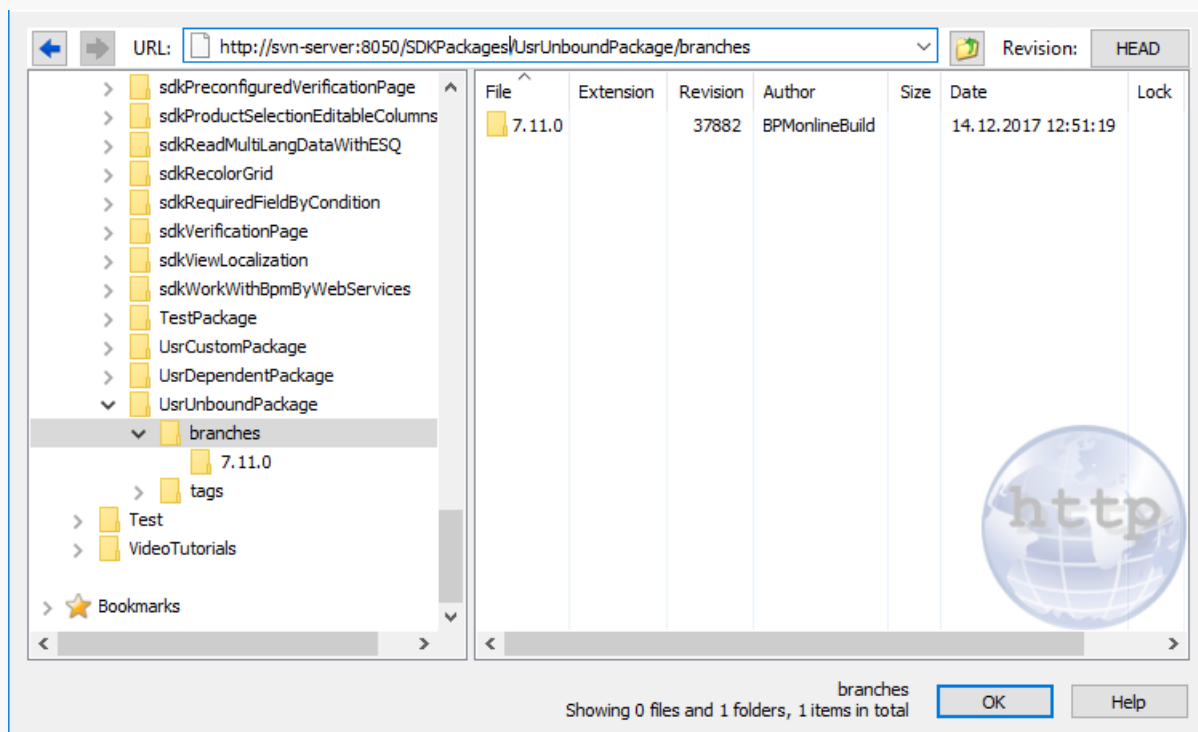
3. Создать необходимые каталоги для пакета в хранилище SVN

Чтобы создать каталоги для пакета, используя клиентское приложение для работы с SVN (например, [TortoiseSvn](#)), необходимо перейти в репозиторий и создать каталог, название которого совпадает с названием пакета.

Важно. Работа с SVN с помощью TortoiseSvn рассмотрена сокращенно. Подробные инструкции о работе с хранилищем при помощи TortoiseSvn доступны в [документации TortoiseSvn](#).



В созданном каталоге необходимо создать подкаталоги `branches` и `tags`, т.е. повторить [плоскую структуру пакетов](#) Creatio. В завершение в каталоге `branches` необходимо создать каталог, название которого совпадает с номером версии пакета — `7.11.0`.

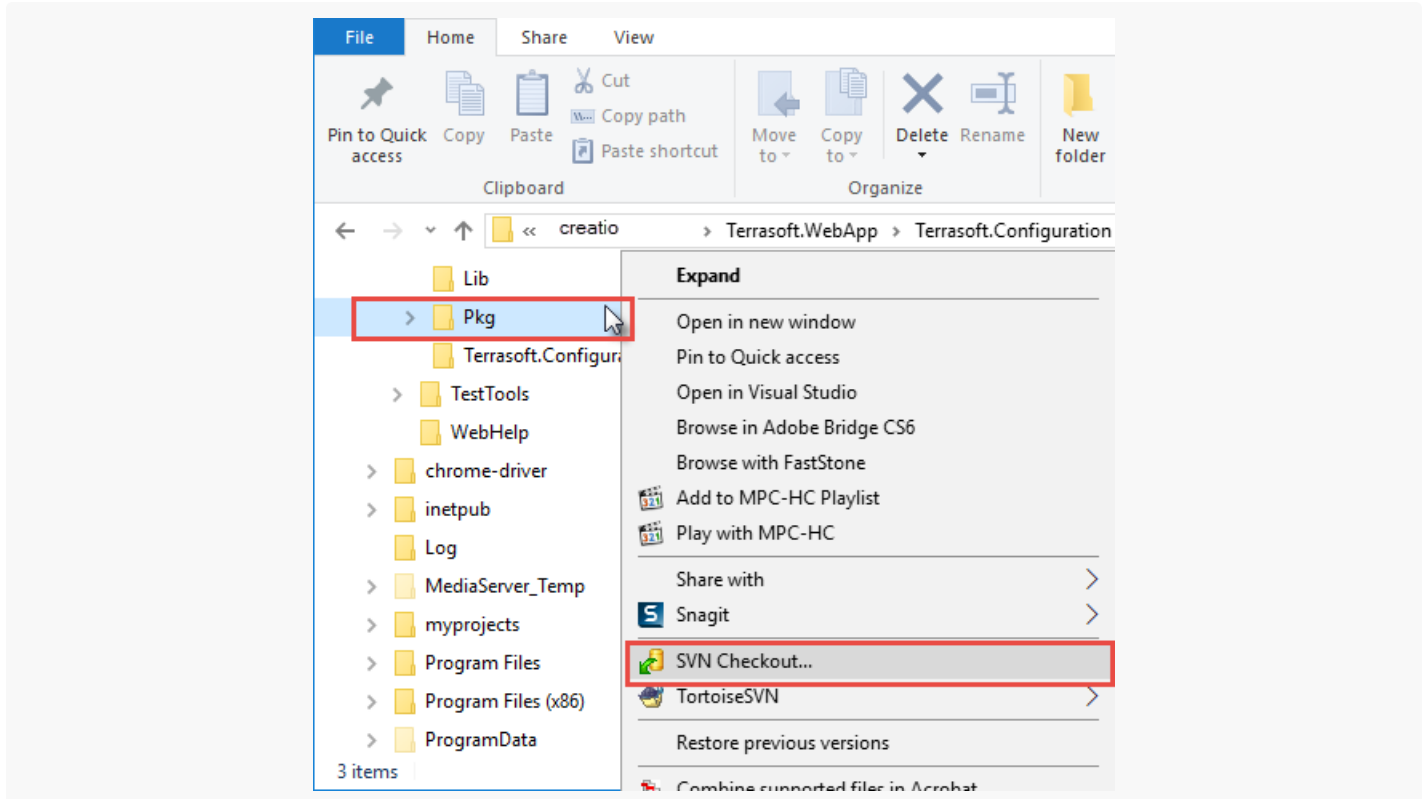


На заметку. Повторять плоскую структуру пакета в хранилище необходимо только в том случае, если в дальнейшем планируется использовать встроенные средства Creatio для работы с SVN.

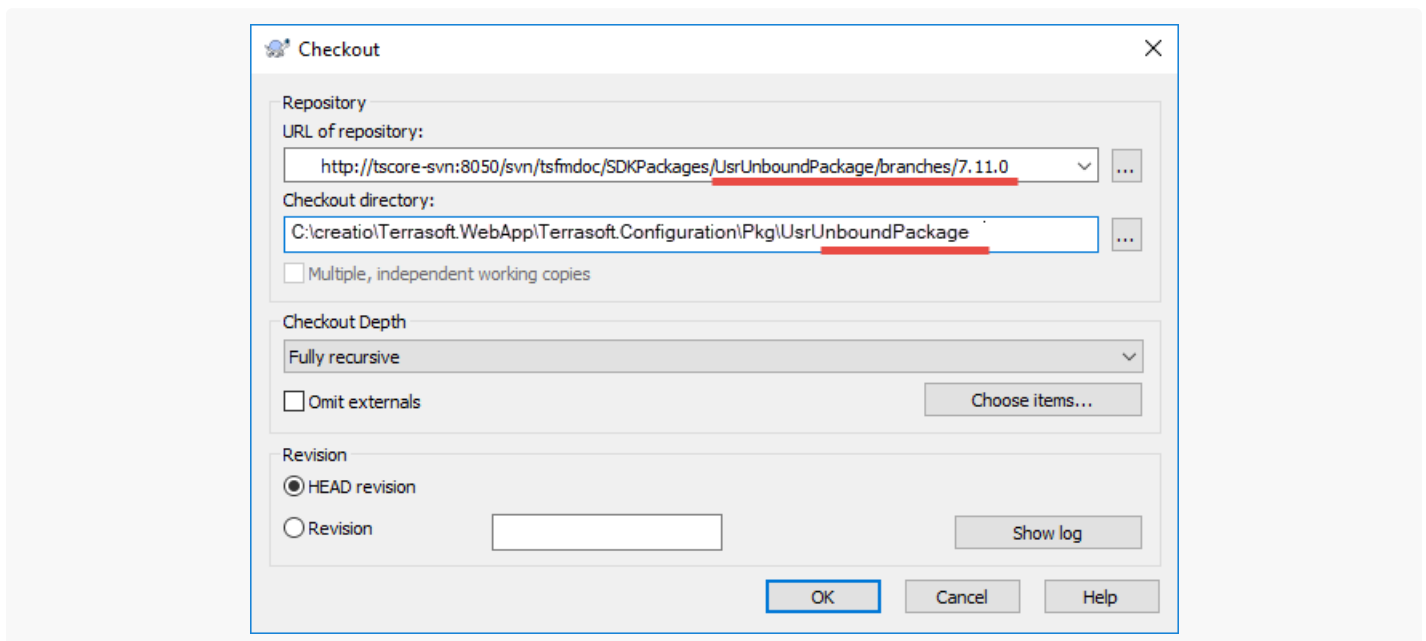
4. Создать рабочую копию версионной ветки пакета

Чтобы создать рабочую копию версионной ветки пакета, необходимо выгрузить из хранилища каталог, имя которого совпадает с номером версии пакета ([*SVN Checkout*]), в каталог пакета в файловой системе и подтвердить выгрузку в существующий каталог.

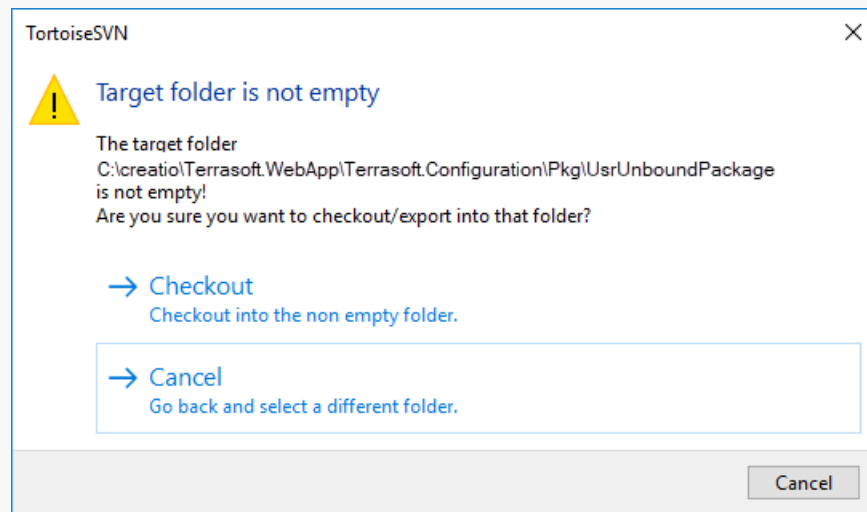
Команда [*SVN Checkout*]



Выгрузка из хранилища рабочей копии версионной ветки пакета



Подтверждение выгрузки в существующий каталог



В результате каталог пакета в файловой системе

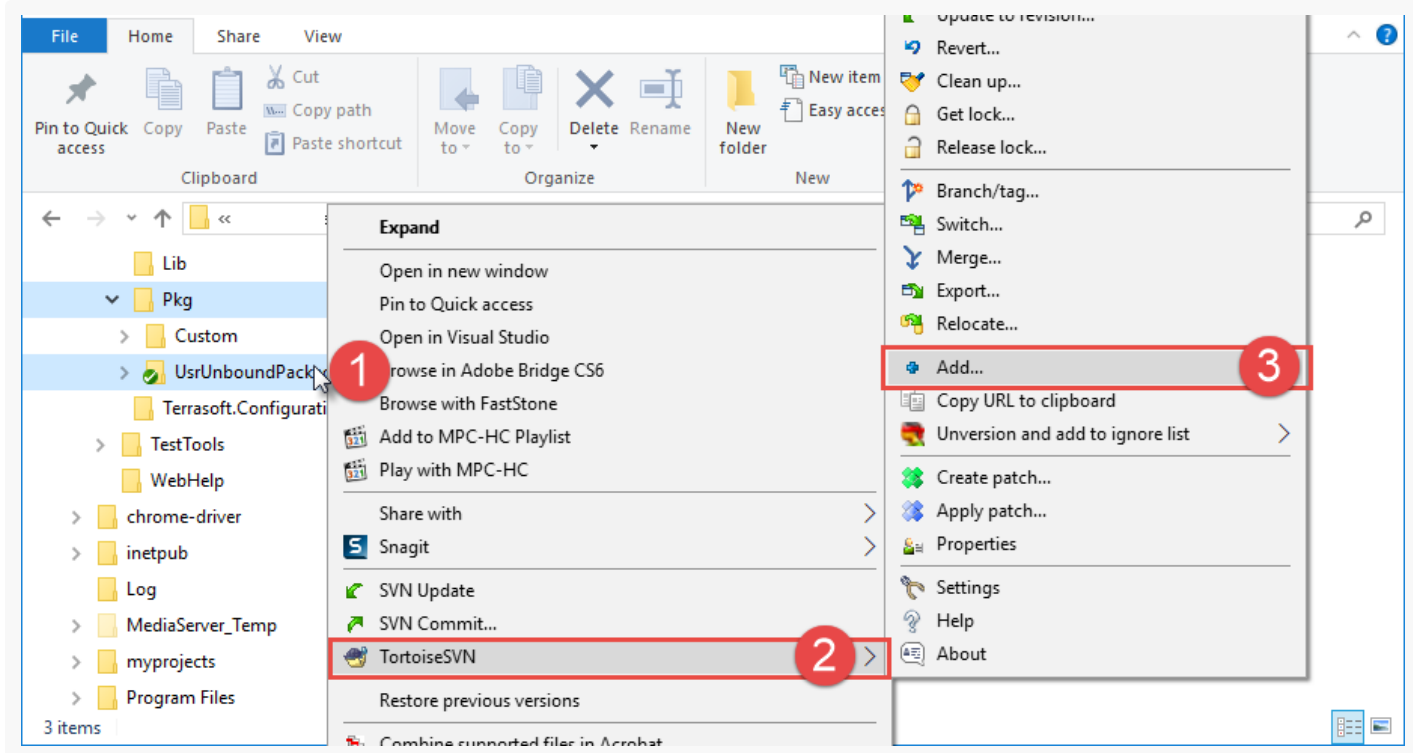
`..\Terrasoft.WebApp\Terrasoft.Configuration\Pkg\UsrUnboundPackage` станет рабочей копией пакета версии 7.11.0 в хранилище.

Custom	07.12.2017 9:38	File folder
UsrUnboundPackage	14.12.2017 12:55	File folder
placeholder.txt	29.11.2017 14:00	Text Document

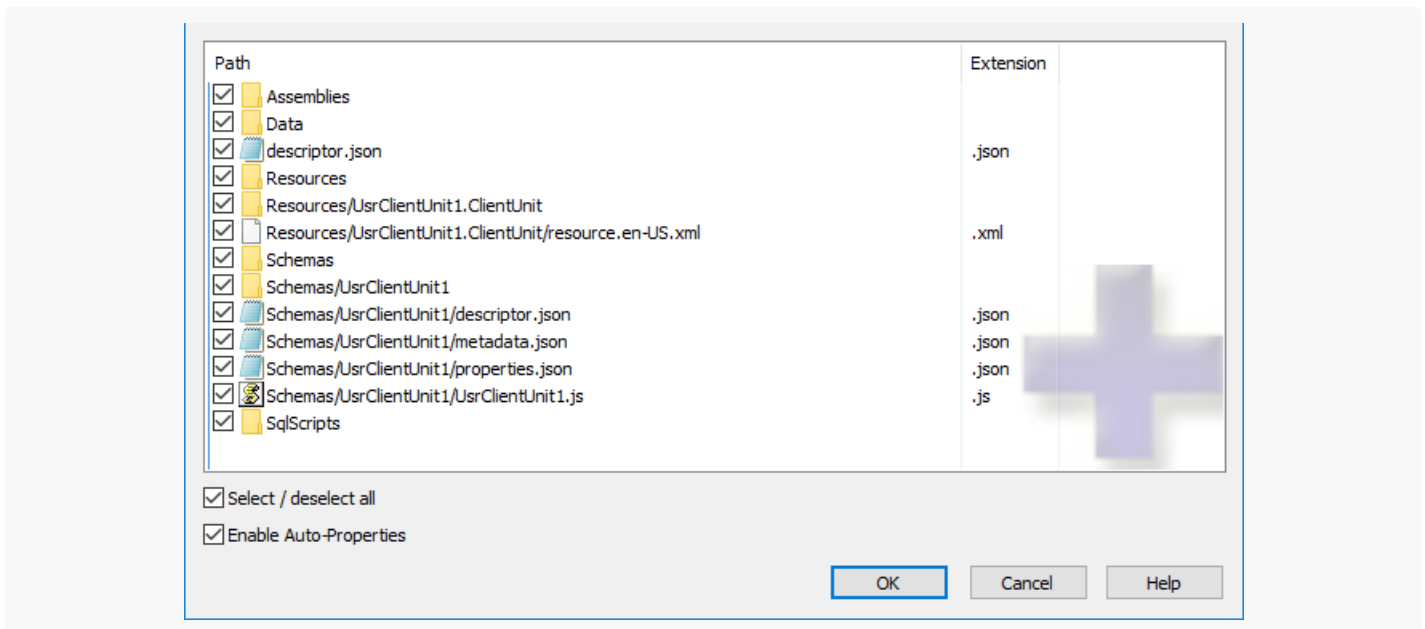
5. Зафиксировать в хранилище каталог пакета

Чтобы зафиксировать в хранилище все содержимое каталога пакета, необходимо выполнить команду [*Add...*] приложения TortoiseSVN, а затем выполнить команду [*SVN Commit...*].

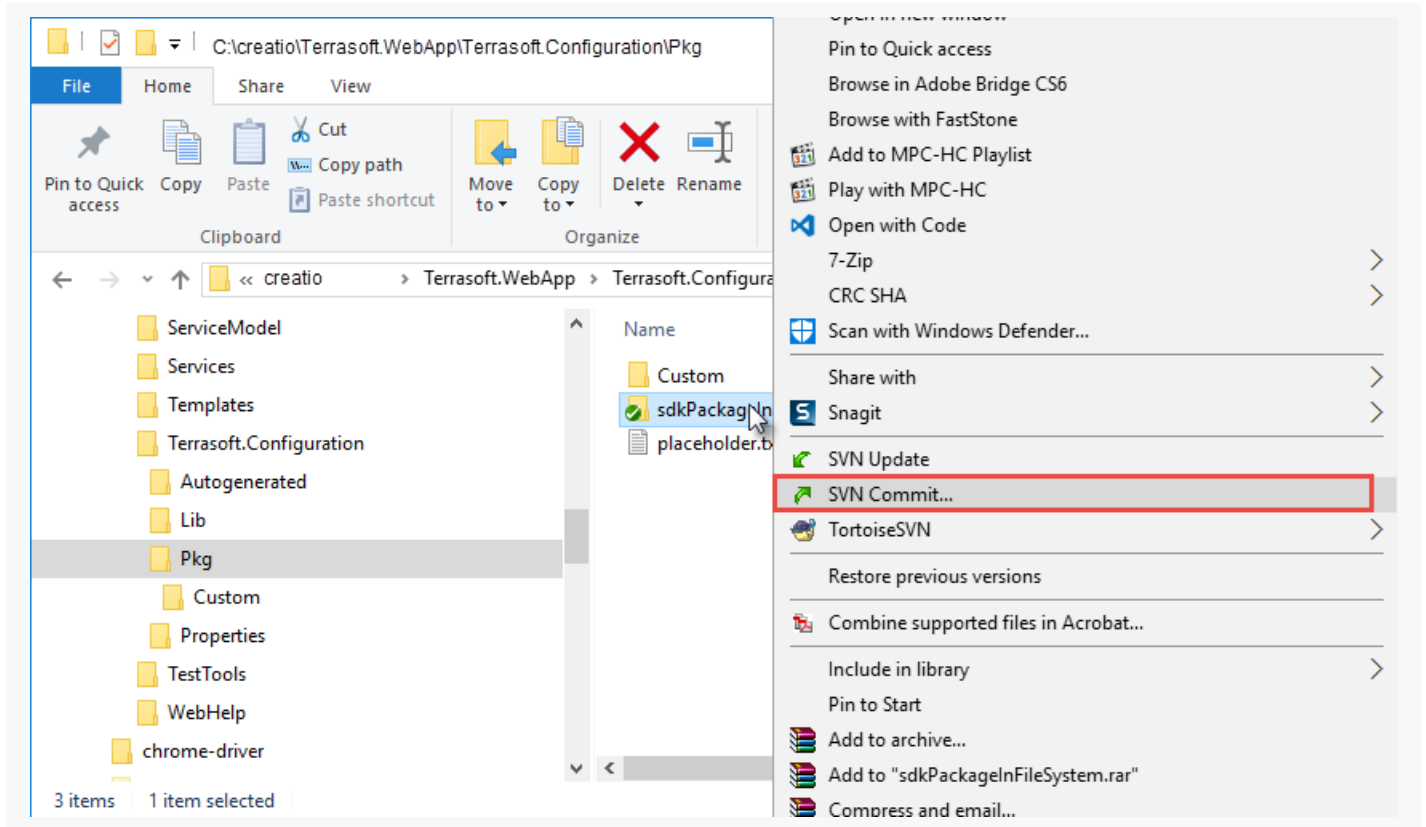
Команда [*Add*]



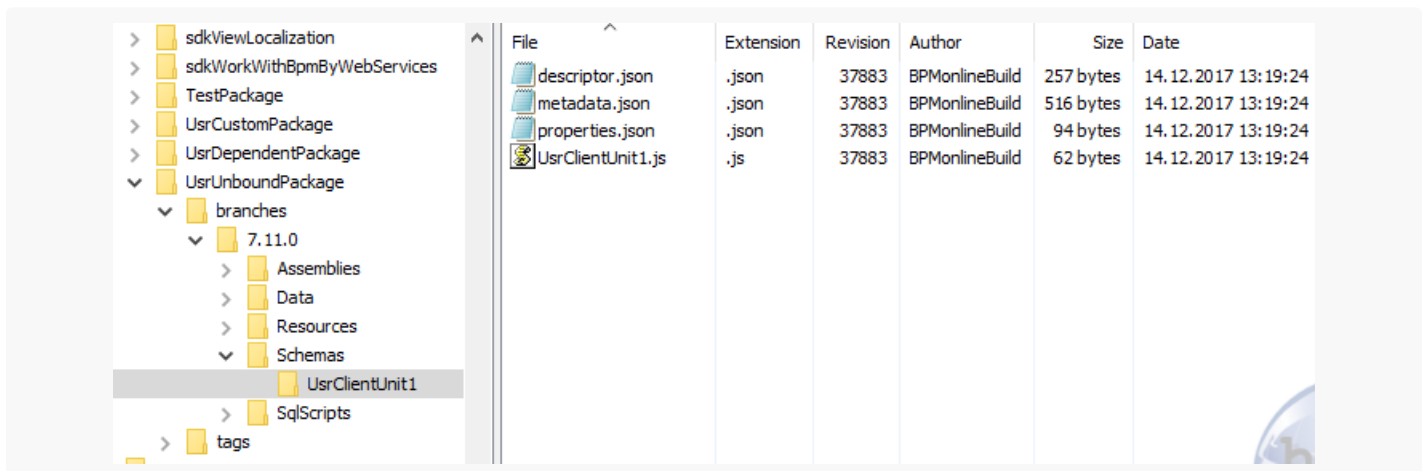
Диалог выбора элементов для добавления в хранилище



Фиксация в хранилище



В результате все содержимое пакета будет привязано к хранилищу SVN.



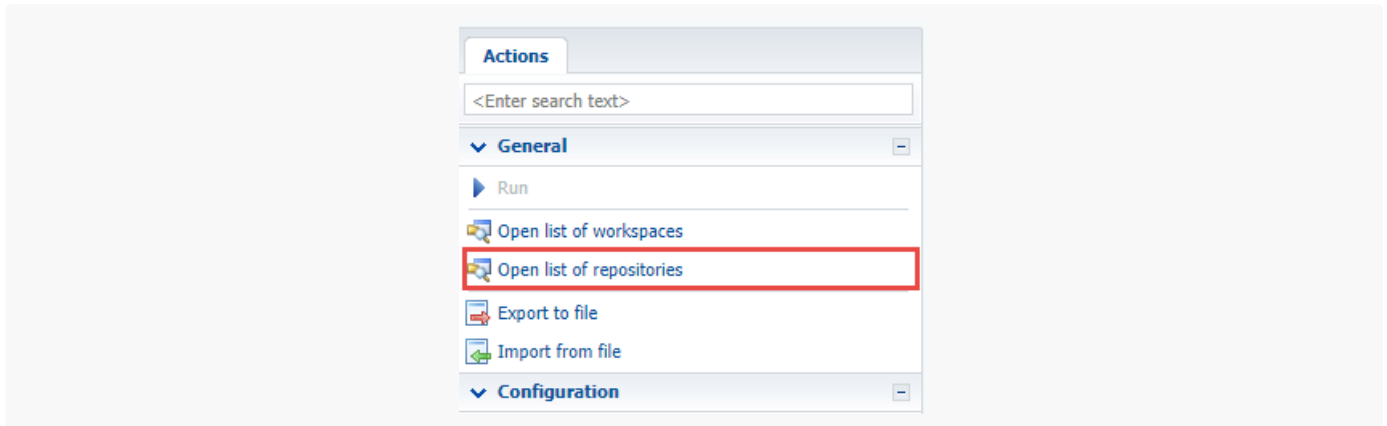
На заметку. Чтобы зафиксировать в хранилище SVN новые схемы пакета, необходимо повторить действия шага 5.

Альтернативный вариант реализации примера

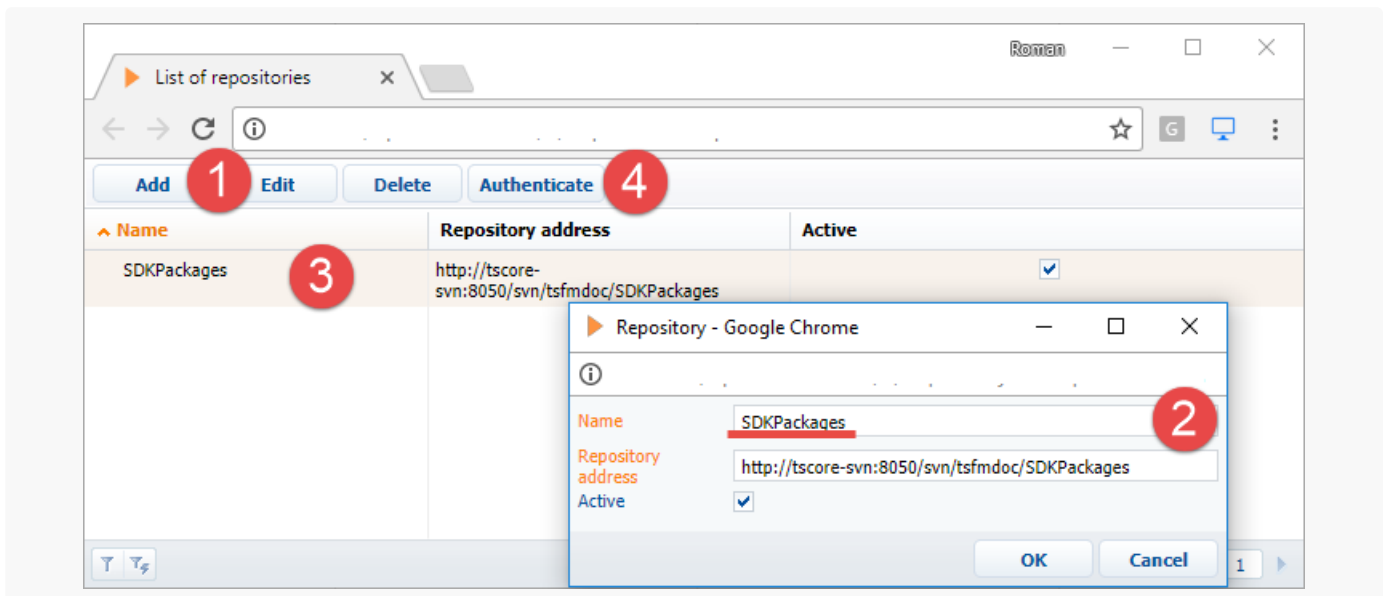
1. В разделе [*Конфигурация*] добавить в приложение информацию о хранилище SVN

Если информация о нужном хранилище не добавлена в разделе [*Конфигурация*] приложения, то необходимо:

1. Выполнить команду [*Открыть список хранилищ*] ([*Open list of repositories*]).



2. В открывшемся окне [*Список хранилищ*] ([*List of repositories*]) при помощи команды [*Добавить*] (1) добавить необходимый репозиторий (2). После этого информация о репозитории отобразится в окне [*Список хранилищ*] (3). Затем необходимо выделить строку с информацией о хранилище и выполнить аутентификацию (4).



2. Привязать хранилище к пакету

Для этого необходимо выполнить SQL-запрос в базу данных приложения.

Пример SQL-запроса

```
UPDATE SysPackage
SET
    [SysRepositoryId] =
    (
```

```
select top 1 Id from SysRepository
where Name = 'SDKPackages'-- Название хранилища.
)
where [Name]='UsrUnboundPackage'-- Название пользовательского пакета.
```

Здесь `SDKPackages` — название хранилища, а `UsrUnboundPackage` — название пользовательского пакета.

Важно. Чтобы изменения применились в приложении, необходимо выйти из приложения и войти в него повторно.

3. Выгрузить пакет в файловую систему

В разделе [*Конфигурация*] ([*Configuration*]) выполнить действие [*Выгрузить пакеты в файловую систему*] ([*Download packages to file system*]). В результате привязанный к хранилищу пакет будет выгружен в каталог `..\Terrasoft.WebApp\Terrasoft.Configuration\Pkg\UsrUnboundPackage`.

Обновить и зафиксировать пакет в SVN в режиме разработки в файловой системе

 Средний

Пример. В конфигурацию Creatio установлен пользовательский пакет `sdkPackageInFileSystem`. В режиме разработки в файловой системе необходимо выполнить его обновление, а после изменения содержимого — фиксацию в хранилище.

1. Обновить пакет из хранилища SVN

Для получения последней ревизии пакета необходимо использовать клиентское приложение для работы с SVN (например, [TortoiseSvn](#)).

Чтобы **обновить пакет из хранилища SVN**:


1. В каталоге `...\Terrasoft.WebApp\Terrasoft.Configuration\Pkg` выберите необходимый пакет. В нашем примере это `sdkPackageInFileSystem`.
2. Обновите пакет (команда [*SVN Update*]).
После выполнения команды будут обновлены дата редактирования пакета в файле дескриптора пакета `descriptor.json` и исходный код схемы типа [*Исходный код*] ([*Source code*]) `UsrGreetingService`.

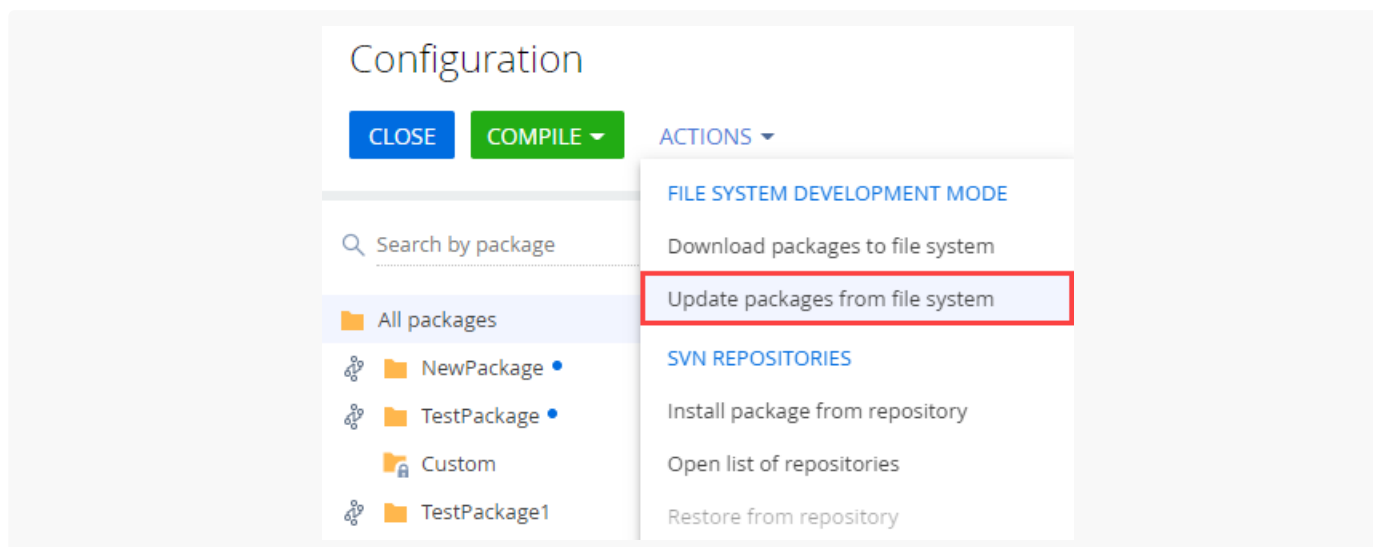
Схема `UsrGreetingService`

```

namespace Terrasoft.Configuration
{
    using System.ServiceModel;
    using System.ServiceModel.Activation;
    using System.ServiceModel.Web;
    [ServiceContract]
    [AspNetCompatibilityRequirements(RequirementsMode = AspNetCompatibilityRequirementsMode.Required)]
    public class UsgreetingService : System.Web.SessionState.IReadOnlySessionState
    {
        [OperationContract]
        [WebInvoke(Method = "GET", UriTemplate = "Hello")]
        public string TestHello()
        {
            return "Hello!";
        }
    }
}

```

3. Перейдите в дизайнер системы по кнопке .
4. В блоке [Конфигурирование разработчиком] ([Admin area]) перейдите по ссылке [Управление конфигурацией] ([Advanced settings]).
5. На панели инструментов в группе действий [Разработка в файловой системе] ([File system development mode]) выберите [Обновить пакеты из файловой системы] ([Update packages from file system]).



6. Если были изменены схемы объектов или схемы исходного кода, то для применения изменений также необходимо выполнить шаги 3—6 статьи [Установить пакет из SVN в режиме разработки в файловой системе](#).

2. Изменить содержимое пакета

Чтобы **изменить содержимое пакета**, добавьте в схему типа [*Исходный код*] ([*Source code*])

UsrGreetingService МЕТОД TestHelloWorld() .

Схема UsrGreetingService

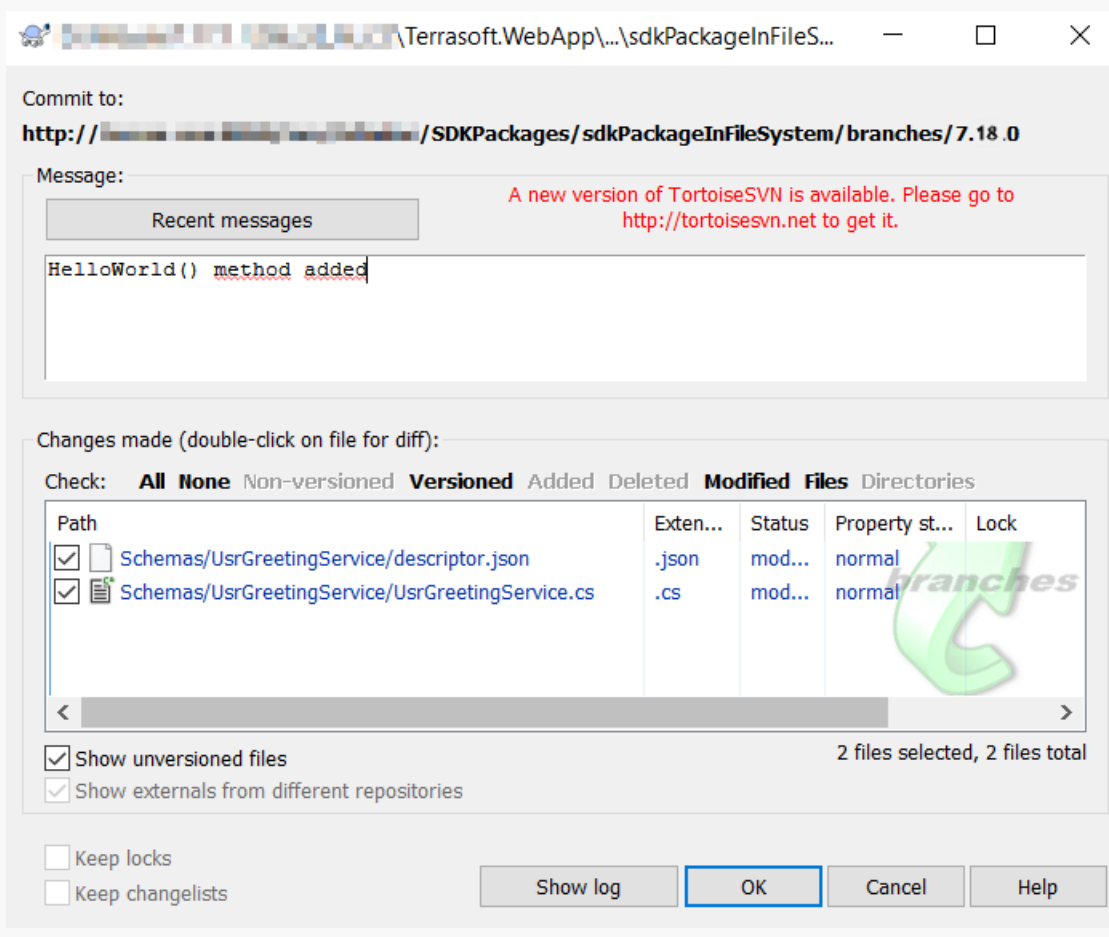
```
namespace Terrasoft.Configuration
{
    using System.ServiceModel;
    using System.ServiceModel.Activation;
    using System.ServiceModel.Web;
    [ServiceContract]
    [AspNetCompatibilityRequirements(RequirementsMode = AspNetCompatibilityRequirementsMode.RequirementsModeNone)]
    public class UsrGreetingService : System.Web.SessionState.IReadOnlySessionState
    {
        [OperationContract]
        [WebInvoke(Method = "GET", UriTemplate = "Hello")]
        public string TestHello()
        {
            return "Hello!";
        }

        [OperationContract]
        [WebInvoke(Method = "GET", UriTemplate = "HelloWorld")]
        public string TestHelloWorld()
        {
            return "Hello world!";
        }
    }
}
```

3. Зафиксировать пакет в хранилище

Чтобы **зафиксировать пакет в хранилище**:

1. В каталоге `..\Terrasoft.WebApp\Terrasoft.Configuration\Pkg` выберите необходимый пакет. В нашем примере это `sdkPackageInFileSystem` .
2. Выполните фиксацию (команда `SVN Commit`) каталога в хранилище.



Создать пакет при переходе в режим разработки в файловой системе

 Средний

Последовательность создания пакета при переходе в режим разработки в файловой системе

В режиме разработки в файловой системе после выполнения действия [*Выгрузить пакеты в файловую систему*] ([*Download packages to file system*]) все пользовательские пакеты будут выгружены в каталог `..\Terrasoft.WebApp\Terrasoft.Configuration\Pkg`. При этом выгруженное в файловую систему содержимое пользовательского пакета не будет привязано к хранилищу SVN даже в том случае, если сам пакет был привязан к хранилищу в разделе [*Конфигурация*].

Если при [создании пакета](#) заполнить поле [*Хранилище системы контроля версий*] с помощью встроенных средств, то пакет будет привязан к хранилищу SVN. При этом в файловой системе будет создана рабочая копия пакета. Путь к каталогу, в котором создаются рабочие копии пакетов, задается настройкой `defPackagesWorkingCopyPath` в файле `ConnectionStrings.config`.

Эту особенность можно использовать для создания пакета, привязанного к SVN и предназначенного для разработки в файловой системе. Если в настройке `defPackagesWorkingCopyPath` указать путь к каталогу `..\Terrasoft.WebApp\Terrasoft.Configuration\Pkg`, то после выгрузки пакета в файловую систему он будет

автоматически привязан к нужному хранилищу SVN.

Общая последовательность действий:

1. В настройке `defPackagesWorkingCopyPath` указать путь к каталогу `..\Terrasoft.WebApp\Terrasoft.Configuration\Pkg.`
2. В режиме разработки с помощью встроенных средств в разделе [*Конфигурация*] создать пакет, привязанный к хранилищу SVN.
3. В разделе [*Конфигурация*] выполнить фиксацию пакета в хранилище.
4. Перейти в режим разработки в файловой системе.
5. Выгрузить пакет в файловую систему.
6. Добавить новые элементы пакета в хранилище SVN.

Пример. В режиме разработки с помощью встроенных средств в разделе [*Конфигурация*] создать пользовательский пакет, привязанный к хранилищу SVN. Выполнить настройку Creatio таким образом, чтобы в режиме разработки в файловой системе после выгрузки пакета его содержимое в файловой системе также было привязано к хранилищу SVN.

Важно. Приведенный в этой статье пример требует четкого понимания разницы между режимами разработки. Общие рекомендации следующие: в режиме разработки в файловой системе работать с хранилищем SVN следует только из файловой системы, а в режиме разработки с помощью встроенных средств работать с SVN нужно только встроенными средствами раздела [*Конфигурация*].

Последовательность реализации примера

1. Изменить настройку `defPackagesWorkingCopyPath`

В файле `ConnectionStrings.config` в настройке `defPackagesWorkingCopyPath` необходимо указать полный путь к каталогу `..\Terrasoft.WebApp\Terrasoft.Configuration\Pkg.`

Пример

```
<?xml version="1.0" encoding="utf-8"?>
<connectionStrings>
  ...
  <add name="defPackagesWorkingCopyPath" connectionString="C:\creatio\Terrasoft.WebApp\Terrasoft
  ...
</connectionStrings>
```

Это изменение позволит совместить каталог, в котором будут содержаться рабочие копии пользовательских пакетов, с каталогом, в который будут выгружаться пакеты в режиме разработки в

файловой системе.

2. Создать пользовательский пакет

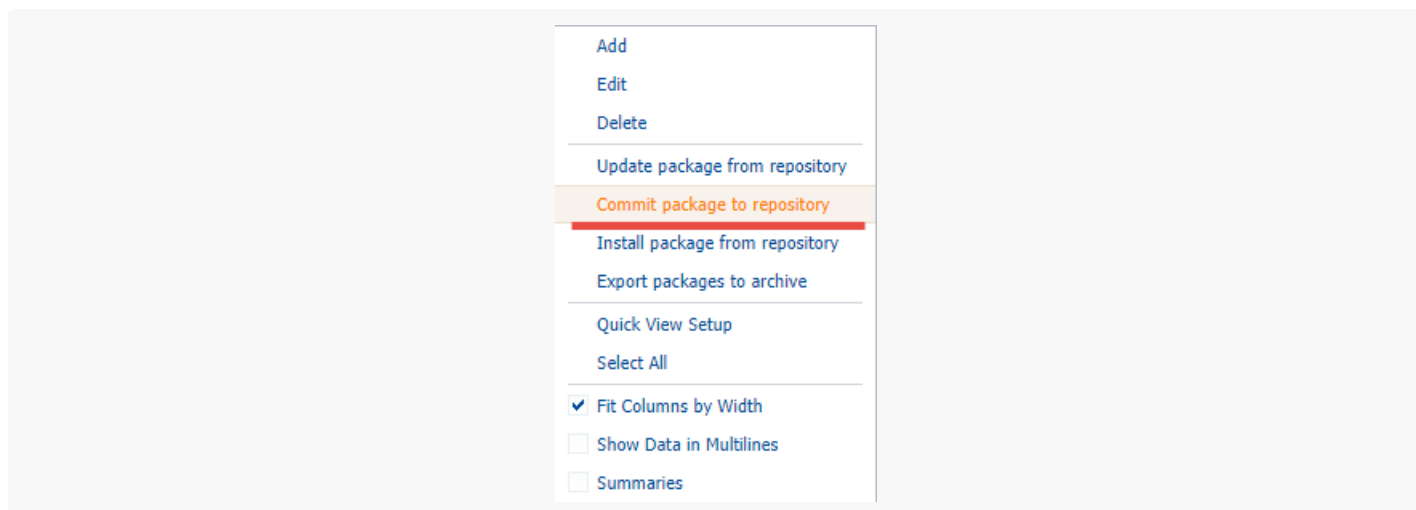
В режиме разработки с помощью встроенных средств в разделе [*Конфигурация*] необходимо [создать пользовательский пакет](#), привязанный к хранилищу SVN. Для создаваемого пакета указать название, репозиторий и версию.

Важно. После создания пакета необходимо добавить нужные зависимости от базовых пакетов.

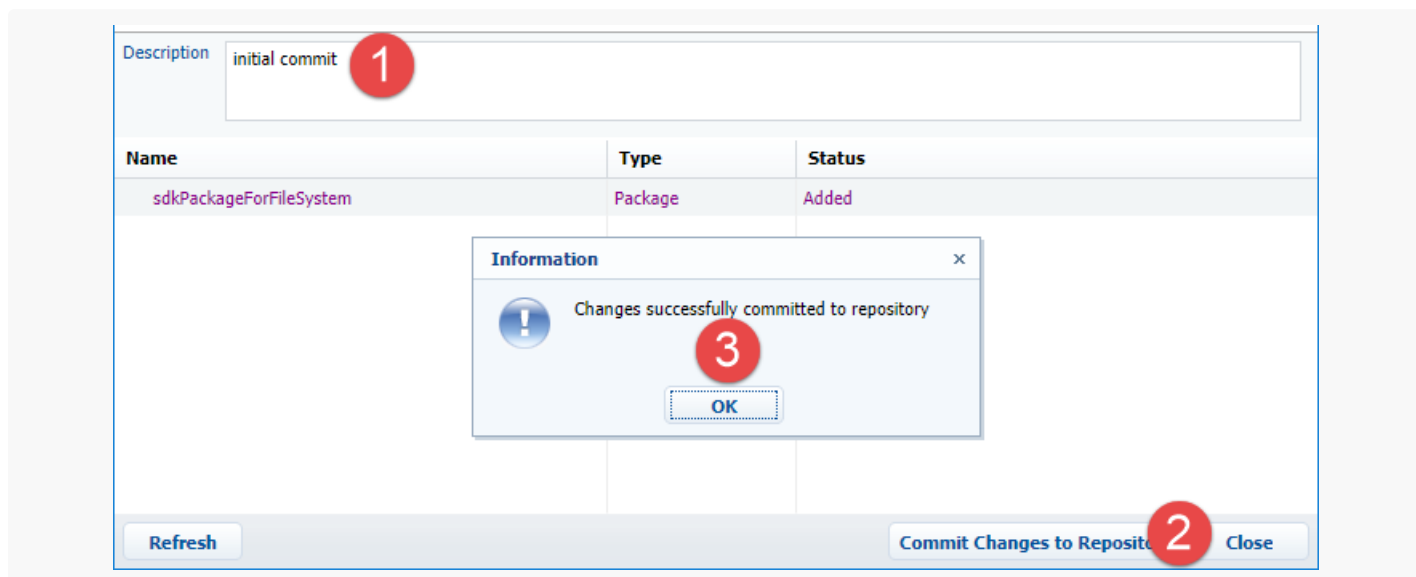
3. Выполнить фиксацию пакета в хранилище

Чтобы зафиксировать пакет в хранилище, необходимо выполнить действие [*Зафиксировать пакет в хранилище*] ([*Commit package to repository*]). В появившемся диалоговом окне нужно указать описание изменений (1), а затем нажать кнопку [*OK*]. После завершения фиксации появится соответствующее сообщение (3).

Действие [*Зафиксировать пакет в хранилище*]

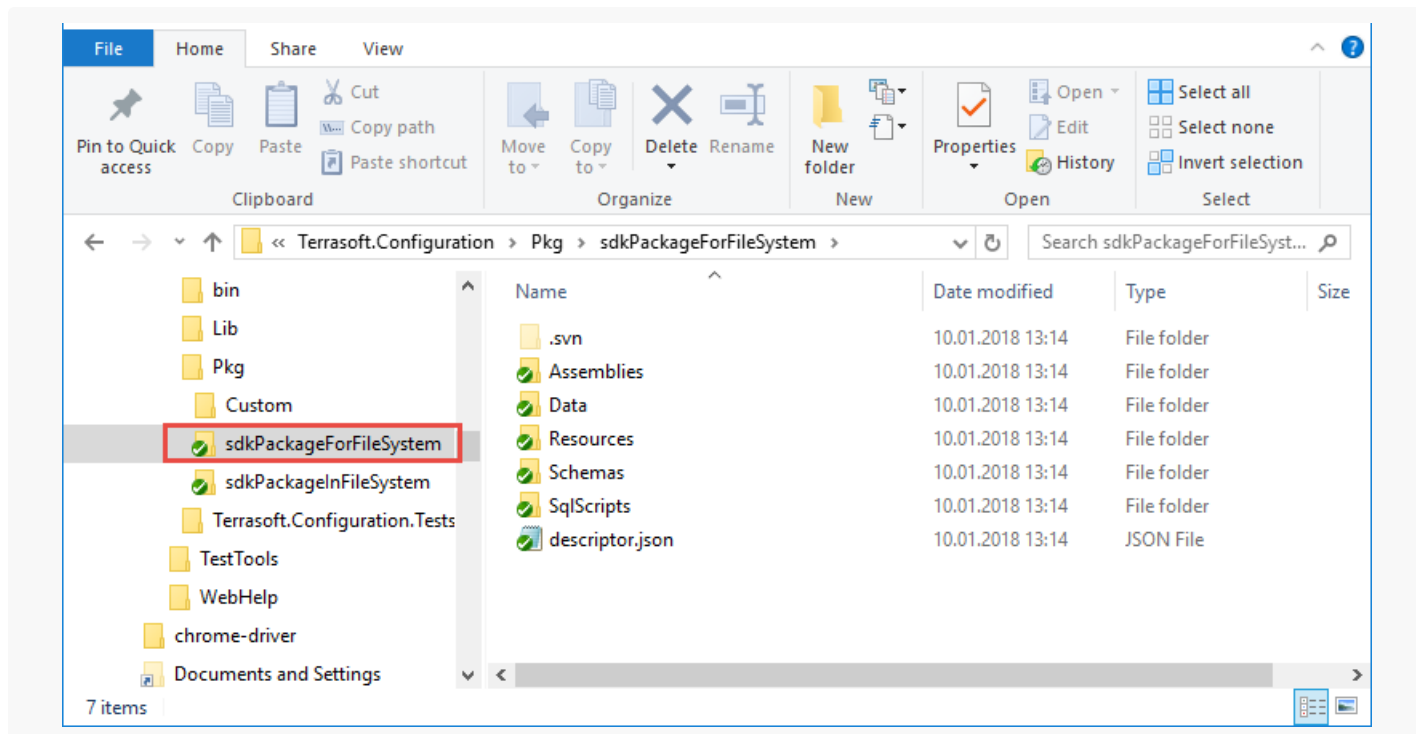


Диалоговое окно свойств заливки



В результате пакет будет зафиксирован в SVN, а в каталоге

`..\Terrasoft.WebApp\Terrasoft.Configuration\Pkg` будет создана рабочая копия пакета.



4. Перейти в режим разработки в файловой системе

Чтобы включить режим разработки конфигурации в файловой системе, необходимо в файле `Web.config`, который находится в корневом каталоге с установленным приложением, установить значение `true` для атрибута `enabled` элемента `fileDesignMode`.

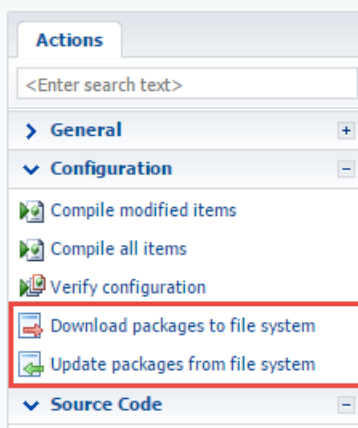
Включение режима разработки конфигурации в файловой системе

```
<fileDesignMode enabled="true"/>
```

Важно. Также необходимо отключить использование [статического контента](#).

После включения режима разработки в файловой системе в разделе [Конфигурация] на вкладке [Действия] появятся две кнопки:

- [Выгрузить пакеты в файловую систему] ([Download packages to file system]) — выгружает пакеты из базы данных приложения в каталог `..\TerraSoft.WebApp\TerraSoft.Configuration\Pkg`.
- [Обновить пакеты из файловой системы] ([Update packages from file system]) — загружает пакеты из каталога `..\TerraSoft.WebApp\TerraSoft.Configuration\Pkg` в базу данных.



5. Выгрузить пакет в файловую систему

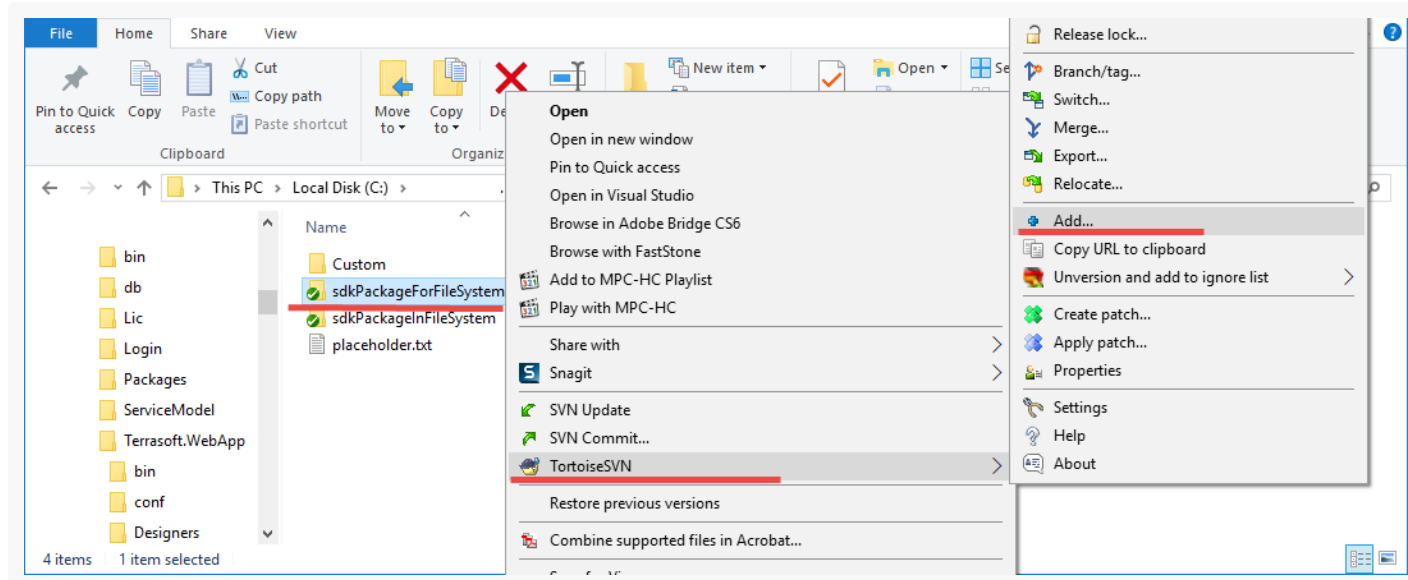
На заметку. Если после фиксации в хранилище содержимое пакета не было изменено, то это действие необязательно.

Для выгрузки пакета в файловую систему необходимо выполнить действие [*Выгрузить пакеты в файловую систему*]. В результате все элементы пакета, измененные и созданные с помощью встроенных средств разработки в разделе [*Конфигурация*], будут выгружены в файловую систему в каталог `..\Terrasoft.WebApp\Terrasoft.Configuration\Pkg`.

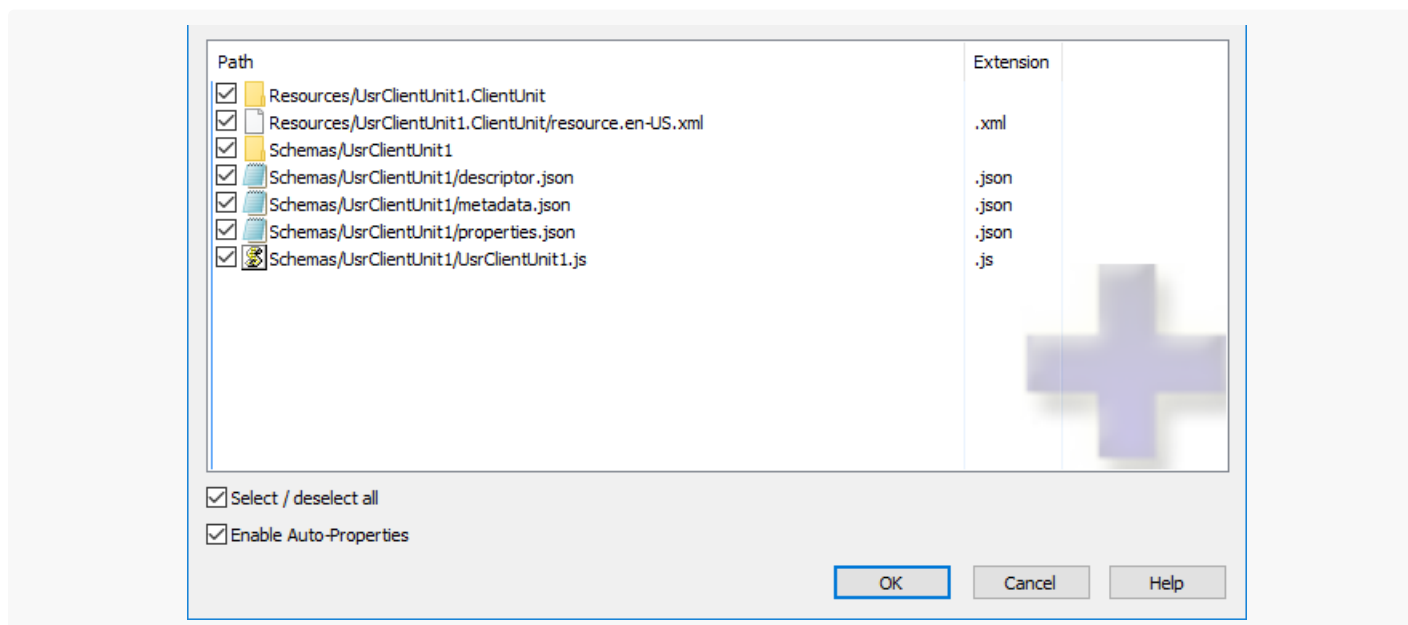
Важно. Поскольку в режиме разработки в файловой системе встроенные средства работы с SVN отключены, то новые элементы пакета не будут привязаны к хранилищу.

6. Добавить новые элементы пакета в хранилище SVN

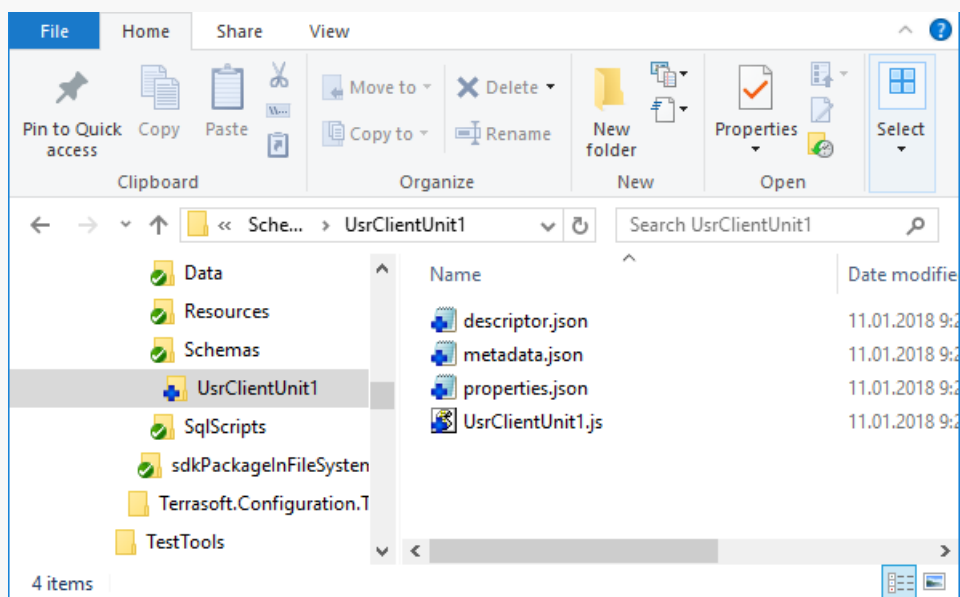
Чтобы в режиме разработки добавить новые элементы пакета в хранилище, необходимо в файловой системе выделить каталог рабочей копии пакета и выполнить команду [*Add...*] клиентского приложения для работы с хранилищем SVN, например, `TortoiseSVN`.



После этого появится диалоговое окно выбора добавляемых элементов. После выбора нужных элементов необходимо нажать на кнопку [OK], после чего отобразится информационное окно о завершении команды добавления новых элементов.



Добавленные элементы будут отмечены, как связанные с хранилищем SVN, но не зафиксированные в нем.



Для фиксации всех измененных и новых элементов пакета в хранилище необходимо выполнить команду [*SVN Commit...*].

