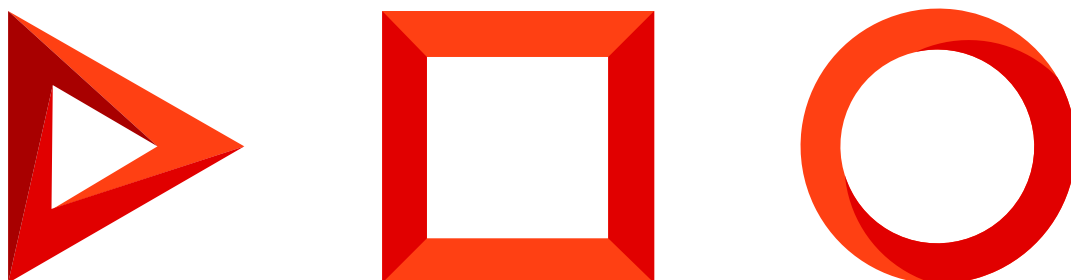


# Контроль версий в Subversion

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

<b>Контроль версий в Subversion</b>	<b>4</b>
Основные понятия	5
Модели версионирования	6
Работа с файлами в SVN	7
Рабочая копия, используемая приложением Creatio	8
Клиентское приложение для работы с SVN	8
<b>Создать пакет в режиме разработки в файловой системе</b>	<b>8</b>
1. Создать пакет	8
2. Выгрузить пакет в файловую систему	9
3. Создать каталоги для пакета в хранилище SVN	10
4. Создать рабочую копию версионной ветки пакета	11
5. Зафиксировать пакет в хранилище	13
<b>Установить пакет из SVN в режиме разработки в файловой системе</b>	<b>14</b>
Вручную установить пакет из SVN в режиме разработки в файловой системе	15
Автоматически установить пакет из SVN в режиме разработки в файловой системе	20
Настроить взаимодействие с хранилищем SVN (опционально)	21
<b>Привязать к SVN не связанный с хранилищем пакет</b>	<b>21</b>
1. Выгрузить пакет в файловую систему	22
2. Создать каталоги для пакета в хранилище SVN	22
3. Создать рабочую копию версионной ветки пакета	23
4. Зафиксировать в хранилище каталог пакета	25
<b>Привязать к SVN не связанный с хранилищем пакет через запрос к базе данных</b>	<b>26</b>
1. Подключите к приложению хранилище SVN	26
2. Привязать хранилище SVN к пакету	27
3. Выгрузить пакет в файловую систему	28
<b>Обновить и зафиксировать пакет в SVN в режиме разработки в файловой системе</b>	<b>29</b>
1. Обновить пакет из хранилища SVN	29
2. Изменить содержимое пакета	30
3. Зафиксировать пакет в хранилище	31
<b>Создать пакет при переходе в режим разработки в файловой системе</b>	<b>32</b>
1. Задать путь к каталогу для рабочих копий пакетов	33
2. Создать пакет	33
3. Выгрузить пакет в файловую систему	34
4. Зафиксировать пакет в хранилище	35

# 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](#).

**Важно.** Систему контроля версий SVN разрешено использовать для переноса изменений только между [средами разработки](#). Запрещено использовать SVN на [предпромышленной](#) и [промышленной](#) среде, поскольку это может привести к неработоспособности или ухудшению производительности приложения.

Неработоспособность приложения при использовании SVN на предпромышленной и промышленной среде может быть вызвана ошибками, которые получены при переносе изменений или при сохранении конфигурационных элементов (например, если в качестве текущего указан пакет, который привязан к SVN). Восстановить работоспособность приложения можно из резервной копии баз данных. Это может

привести к потере изменений, которые были выполнены в приложении с момента создания последней резервной копии.

Также при использовании SVN на предпромышленной и промышленной среде изменения могут поставляться без предварительного тестирования на среде разработки, что категорически запрещено.

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.

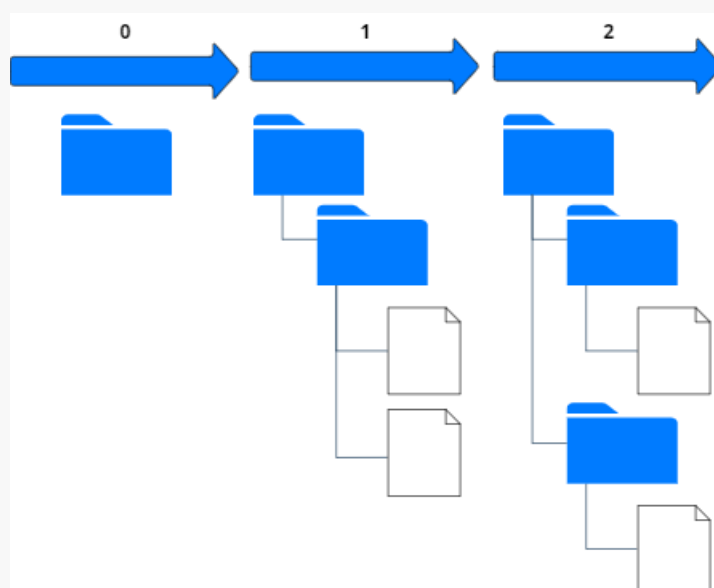
**Рабочая копия** — каталог на локальном компьютере, с которым работает пользователь. Рабочая копия содержит копию файлов, которые были в хранилище до того, как их начал изменять пользователь. Таким образом можно узнать, какие конкретно изменения были выполнены.

**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`.

Для настройки работы с SVN необходимо изменить настройку `sourceControlAuthPath` конфигурационного файла `ConnectionStrings.config`. Эта настройка содержит путь к каталогу на диске, который хранит аутентификационные данные (логин и зашифрованный пароль) подключения пользователя к SVN-репозиторию. Аутентификационные данные сохраняются SVN-клиентом, встроенным в Creatio. В качестве значения настройки `sourceControlAuthPath` рекомендуется установить путь на фиксированный каталог, поскольку временный каталог, установленный по умолчанию, может быть очищен операционной системой.

## 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



При включенном [режиме разработки в файловой системе](#) механизм интеграции с 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](#).

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.

**Пример.** Создать пакет в режиме разработки в файловой системе.

## 1. Создать пакет

Чтобы **создать пользовательский пакет**:

1. Перейдите в дизайнер системы по кнопке
2. В блоке [ *Конфигурирование разработчиком* ] ([ *Admin area* ]) перейдите по ссылке [ *Управление конфигурацией* ] ([ *Advanced settings* ]).



3. В области работы с пакетами нажмите кнопку .

4. Заполните **свойства пакета**:

## Package

Name\*  
sdkTestPackage

Description

Version control system repository

Version \*

SDKPackages

7.18.0

CANCEL

CREATE AND ADD DEPENDENCIES

SAVE

- [ *Название* ] ([ *Name* ]) — "sdkTestPackage".
- [ *Хранилище системы контроля версий* ] ([ *Version control system repository* ]) — "SDKPackages".

Если заполнить поле [ *Хранилище системы контроля версий* ] ([ *Version control system repository* ]), то пакет будет привязан к хранилищу. При включенном режиме разработки в файловой системе каталог с названием, соответствующим пакету, необходимо вручную добавить в хранилище.

Если не заполнять поле [ *Хранилище системы контроля версий* ] ([ *Version control system repository* ]), то пакет не будет привязан к хранилищу. Вести версиюнную разработку этого пакета можно будет только подключив его вручную из файловой системы.

- [ *Версия* ] ([ *Version* ]) — "7.18.0".

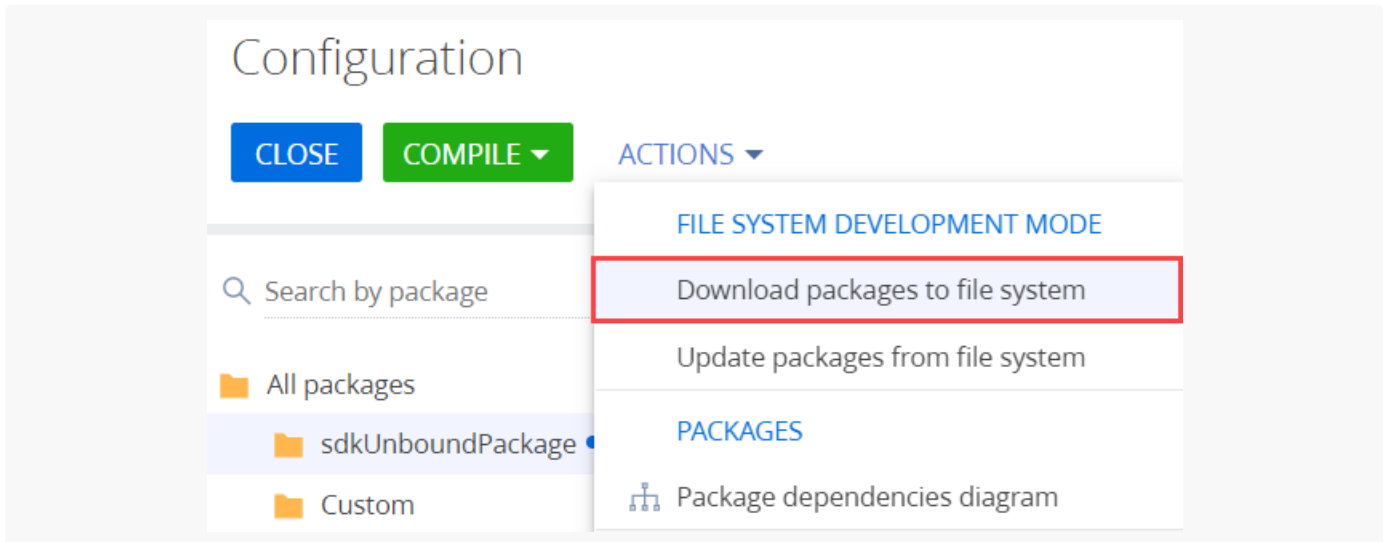
**Важно.** Название репозитория в свойствах пакета указывает только на то, что каталог для пакета будет создан сторонними средствами в этом репозитории. Это позволит в дальнейшем выполнять обновление пакета из раздела [ *Конфигурация* ] ([ *Configuration* ]).

5. Нажмите кнопку [ *Создать и добавить зависимости* ] ([ *Create and add dependencies* ]) и установите зависимости пакета.

## 2. Выгрузить пакет в файловую систему

Чтобы **выгрузить пакет** в файловую систему:

1. Настройте Creatio для работы в файловой системе. Настройка описана в статье [Внешние IDE](#).
2. На панели инструментов в группе действий [ *Разработка в файловой системе* ] ([ *File system development mode* ]) выберите [ *Выгрузить все пакеты в файловую систему* ] ([ *Download packages to file system* ]).



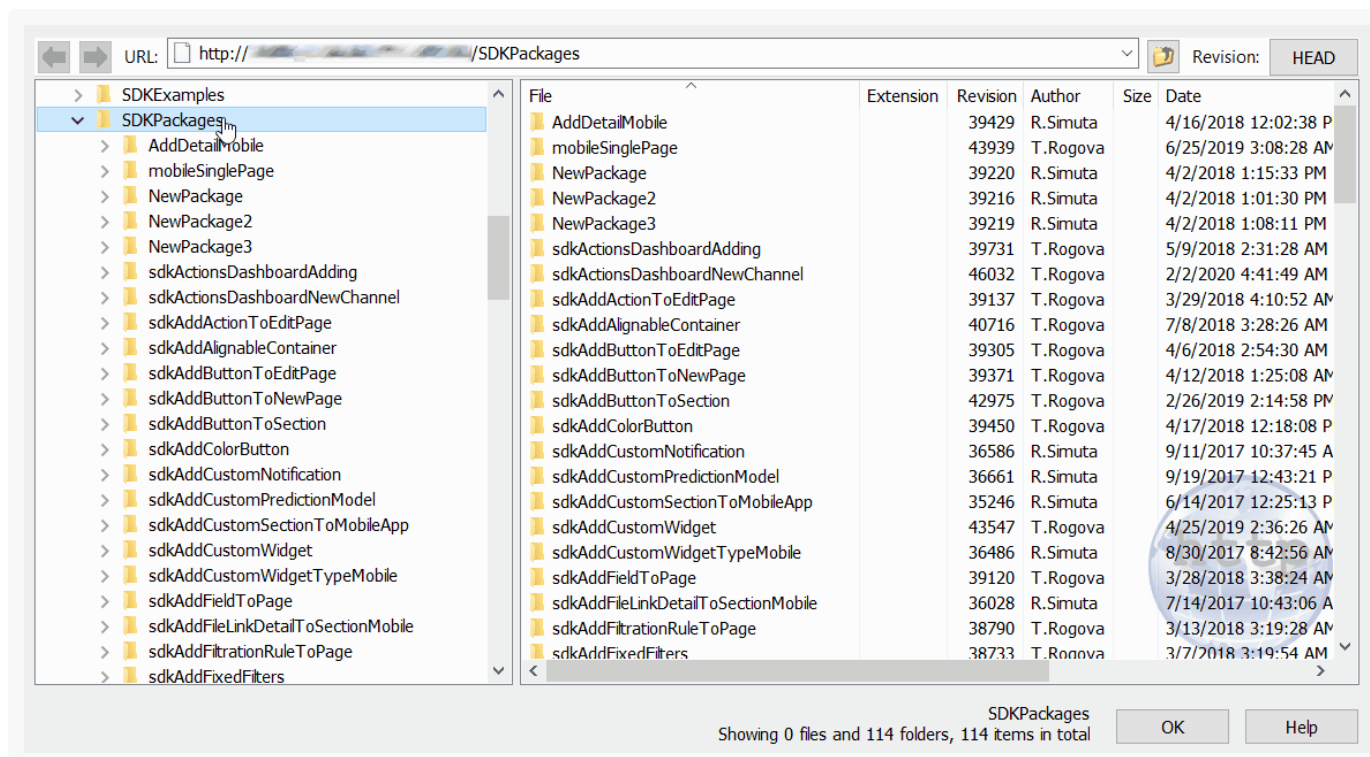
В результате все пакеты будут выгружены по пути `..\Terrasoft.WebApp\Terrasoft.Configuration\Pkg` в каталог с соответствующим названием пакета.

### 3. Создать каталоги для пакета в хранилище SVN

Чтобы создать каталоги для пакета в хранилище SVN необходимо использовать клиентское приложение для работы с SVN (например, [TortoiseSvn](#)).

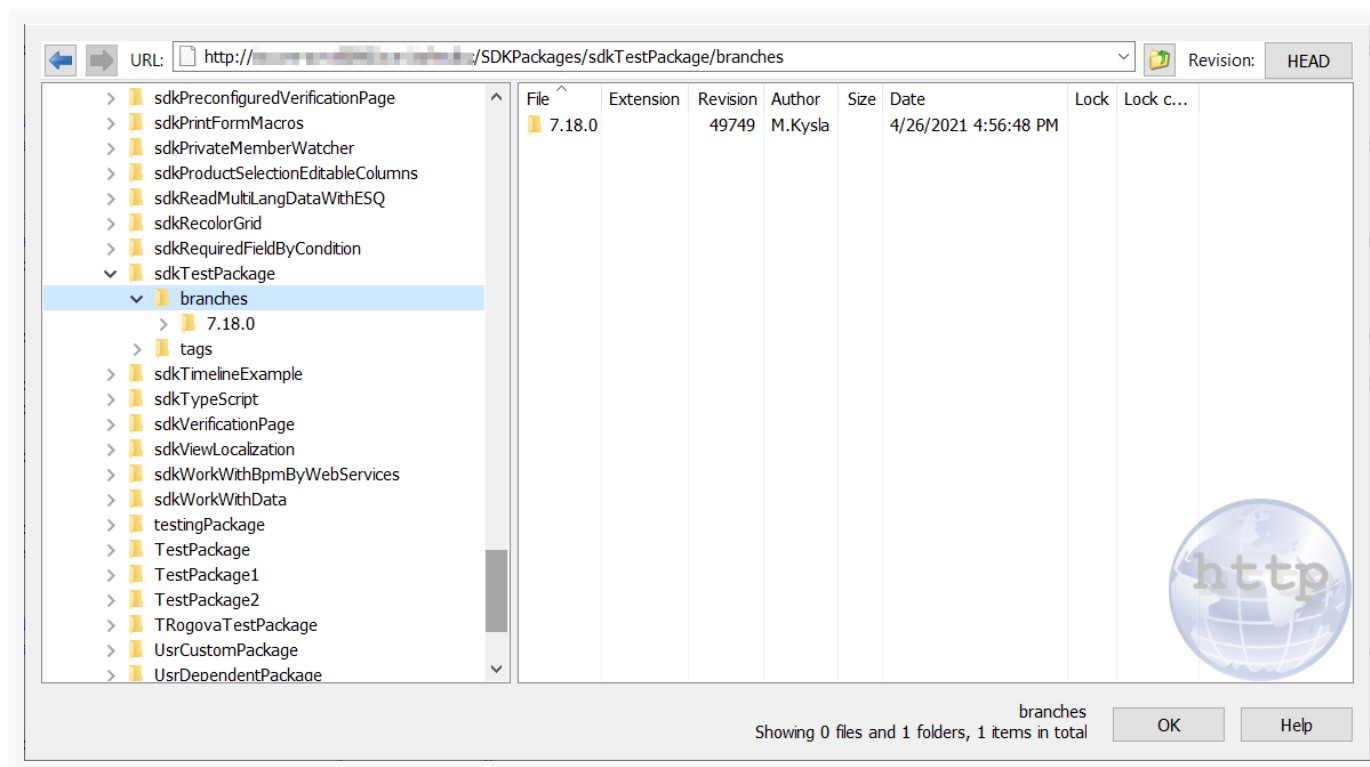
Чтобы **создать каталоги** для пакета в хранилище SVN:

1. Перейдите в репозиторий, указанный в свойствах пакета.
2. В репозитории создайте каталог, название которого совпадает с названием созданного в приложении пакета. В нашем примере это `sdkTestPackage`.



3. В созданном каталоге создайте подкаталоги `branches` и `tags`.

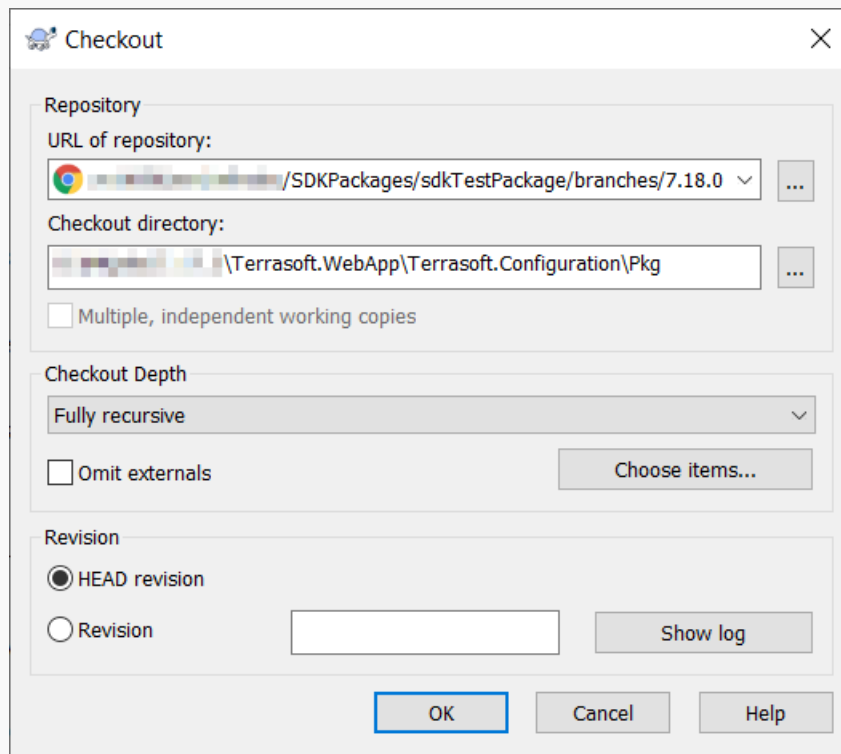
4. В каталоге `branches` создайте каталог, название которого совпадает с номером версии пакета — `7.18.0`.



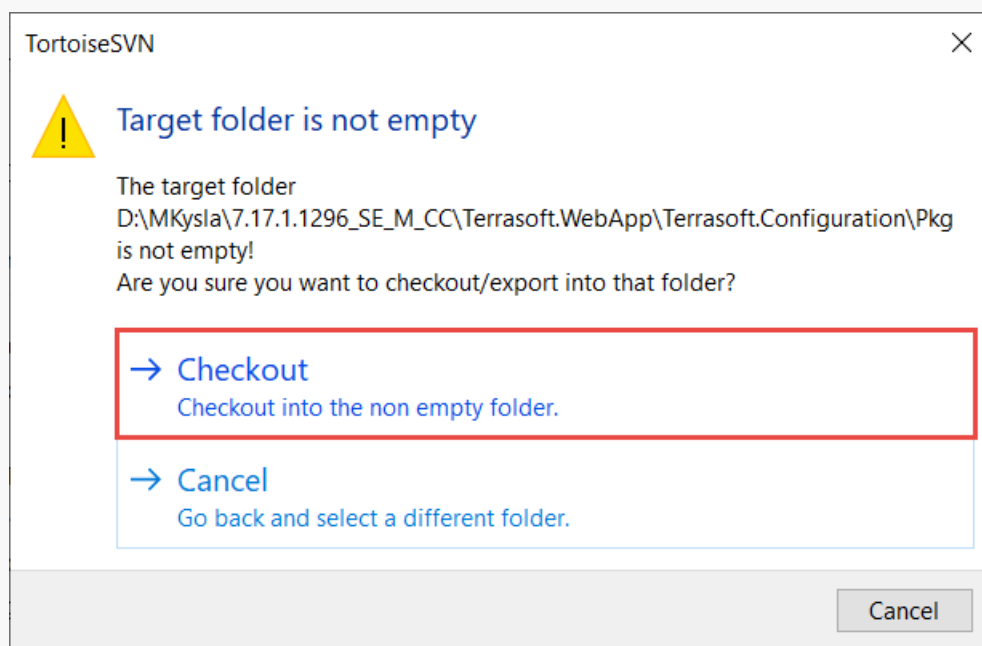
## 4. Создать рабочую копию версионной ветки пакета

Чтобы **создать рабочую копию версии ветки пакета**:

1. Выгрузите (команда `SVN Checkout...`) созданный на предыдущем шаге каталог из хранилища в каталог пакета в файловой системе. В нашем примере это каталог `7.18.0`.



2. Подтвердите выгрузку в существующий каталог.



В результате каталог пакета в файловой системе

`...\\Terrasoft.WebApp\\Terrasoft.Configuration\\Pkg\\sdkTestPackage` будет связан с веткой версии `7.18.0`

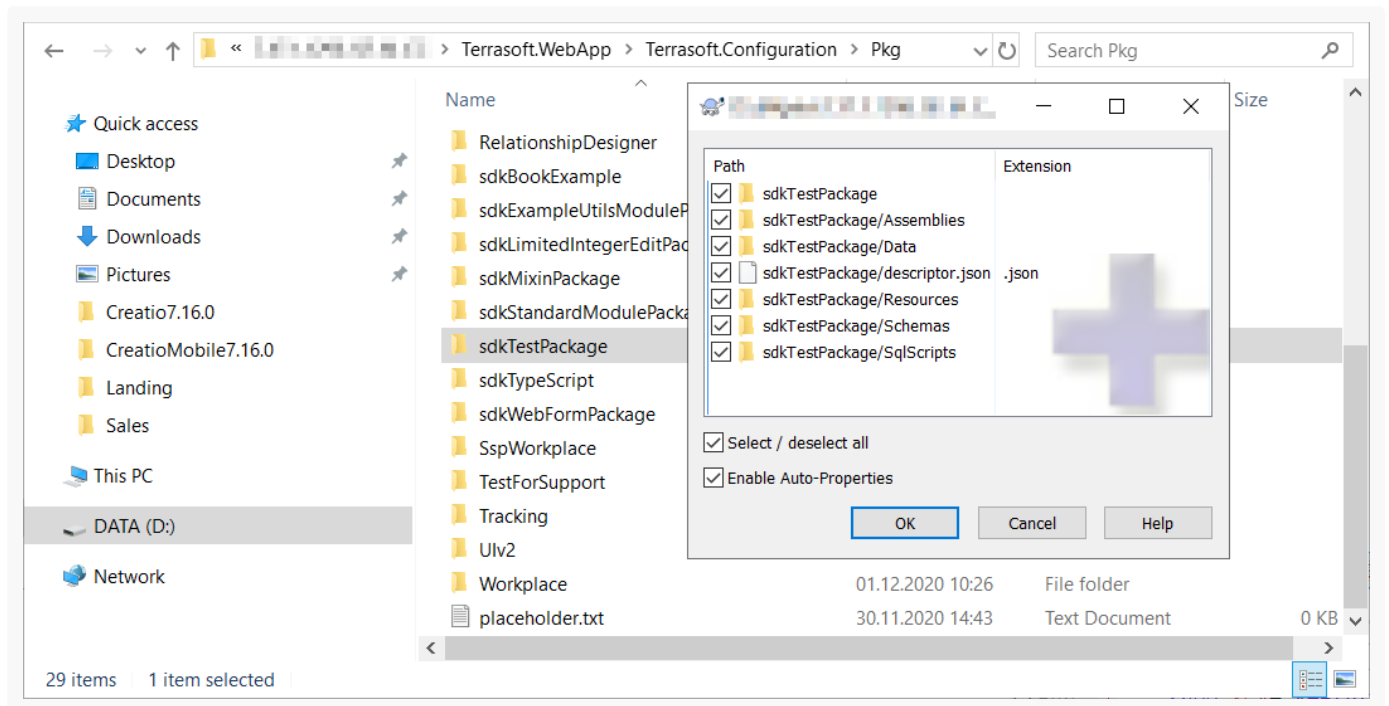
пакета в хранилище.

## 5. Зафиксировать пакет в хранилище

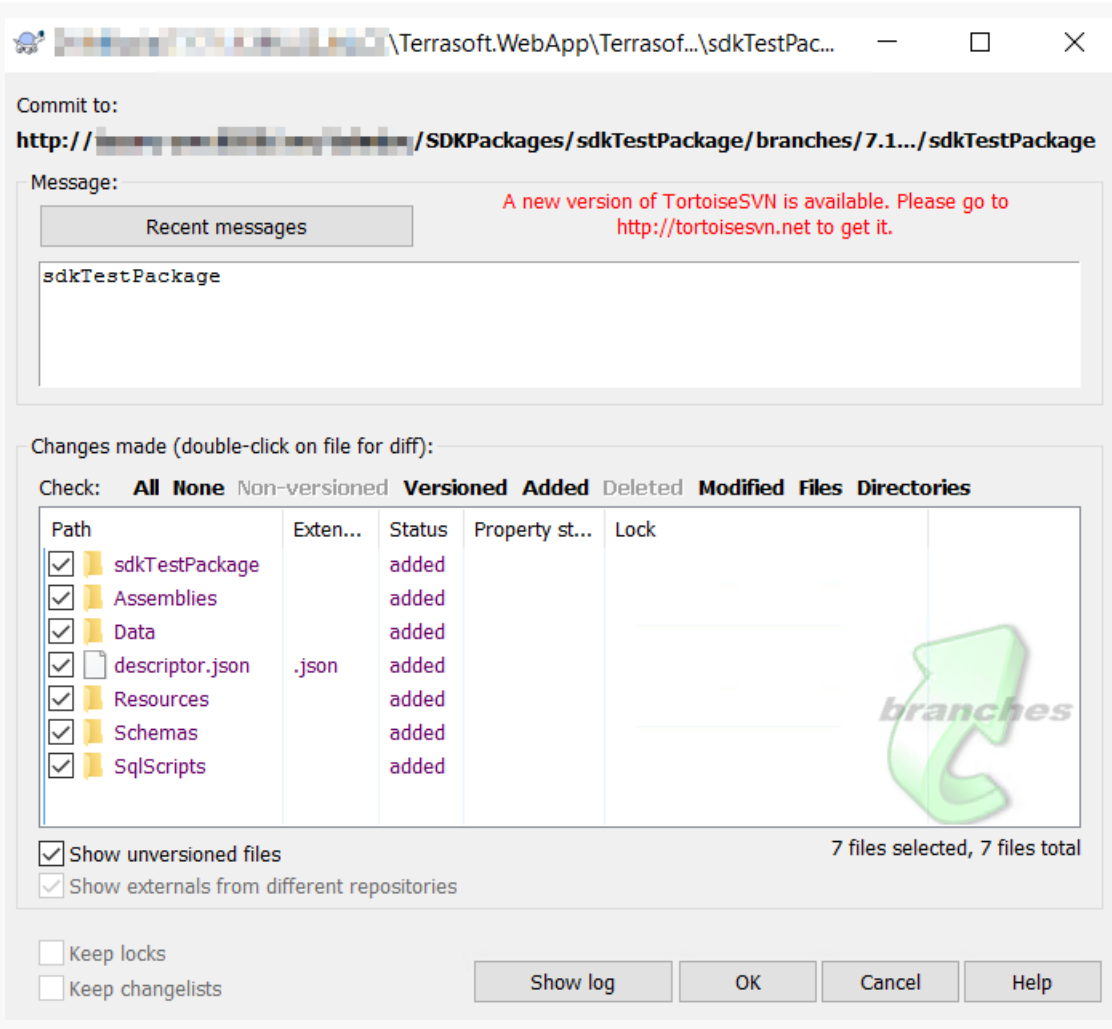
Чтобы **зафиксировать пакет в хранилище**:

1. Добавьте (команда `SVN Commit`) в хранилище содержимое каталога

`...\Terrasoft.WebApp\Terrasoft.Configuration\Pkg\sdkTestPackage .`



2. Выполните фиксацию каталога в хранилище.



## Установить пакет из SVN в режиме разработки в файловой системе

 Средний

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

Адрес пакета в хранилище SVN — `.../SDKPackages/sdkCreateDetailWithEditableGrid/branches/7.18.0`.

В пакете содержится функциональность [детали с редактируемым реестром](#).

Creatio предоставляет возможность установить существующий пакет из SVN в режиме разработки в файловой системе автоматически и вручную.

Последовательность действий при **ручной установке пакета**:

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. Установить SQL-сценарии и привязанные данные (опционально).

Последовательность действий при **автоматической установке пакета**:

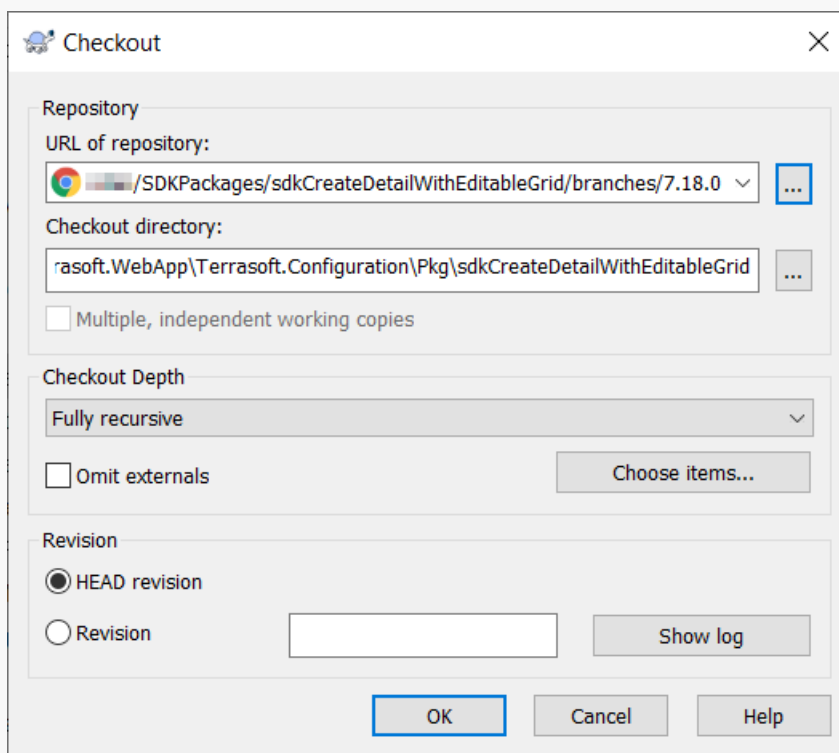
1. Включить автоматическое применение изменений.
2. Установить пакет в файловую систему.
3. Установить пакет в приложение.

## Вручную установить пакет из SVN в режиме разработки в файловой системе

### 1. Установить пакет в файловую систему

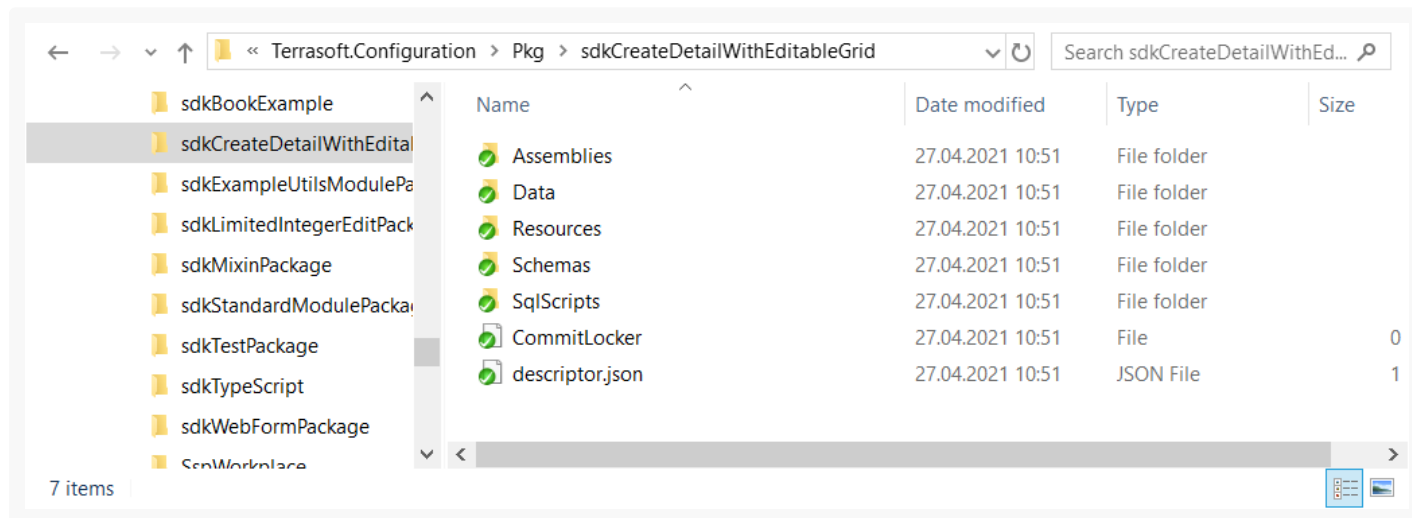
Чтобы **установить пакет в файловую систему**:

1. В каталоге приложения `...\Terrasoft.WebApp\Terrasoft.Configuration\Pkg` создайте каталог, название которого совпадает с названием пакета.
2. Выгрузите (команда `SVN Checkout...`) созданный на предыдущем шаге каталог из хранилища в каталог пакета в файловой системе.
3. Укажите адрес хранилища, по которому размещено содержимое пакета, и каталог для выгрузки содержимого пакета.




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

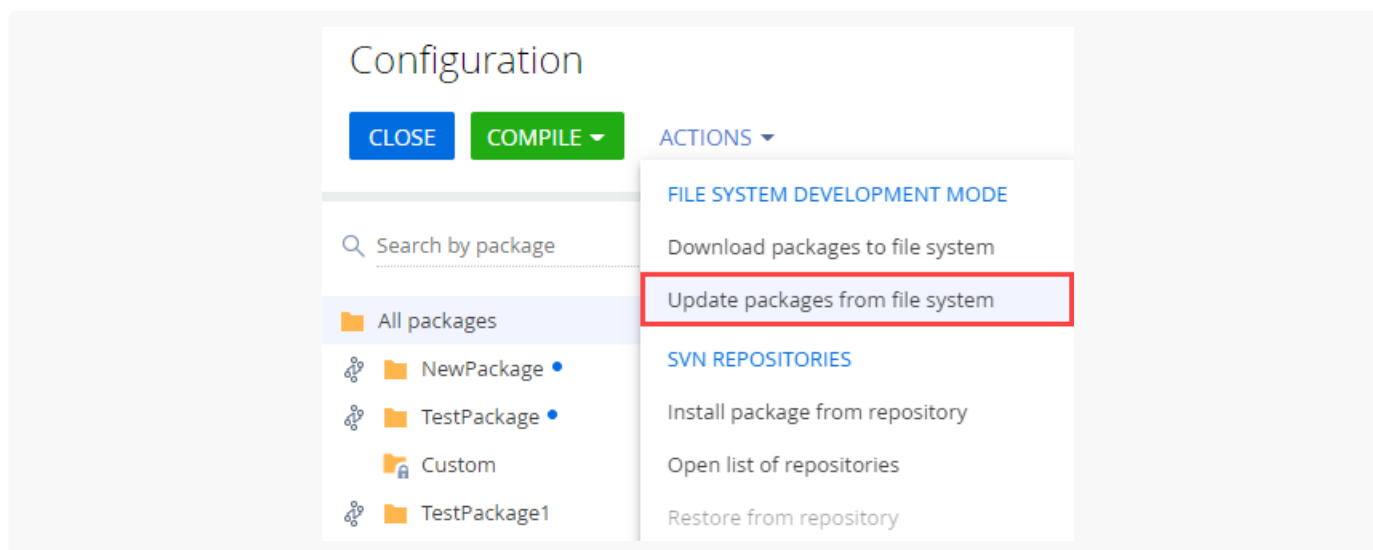
После выгрузки в каталоге `..\Terrasoft.WebApp\Terrasoft.Configuration\Pkg` будет создана рабочая копия пакета.



## 2. Установить пакет в приложение

Чтобы **установить пакет в приложение**:

1. Перейдите в дизайнер системы по кнопке .
2. В блоке [ *Конфигурирование разработчиком* ] ([ *Admin area* ]) перейдите по ссылке [ *Управление конфигурацией* ] ([ *Advanced settings* ]).
3. 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* ]).



В результате пакет будет добавлен в приложение.



Информационное окно статуса загрузки пакета в приложение

## Changes

Name	Type	Status
▼ sdkCreateDetailWithEditableGrid	Package	Added
OrderPageV2	Schema	Added
UsrCourierService	Schema	Added
UsrCourierServiceDetail	Schema	Added
OrderPageV2	Schema Resource	Added
OrderPageV2	Schema Resource	Added
OrderPageV2	Schema Resource	Added
OrderPageV2	Schema Resource	Added

CLOSE

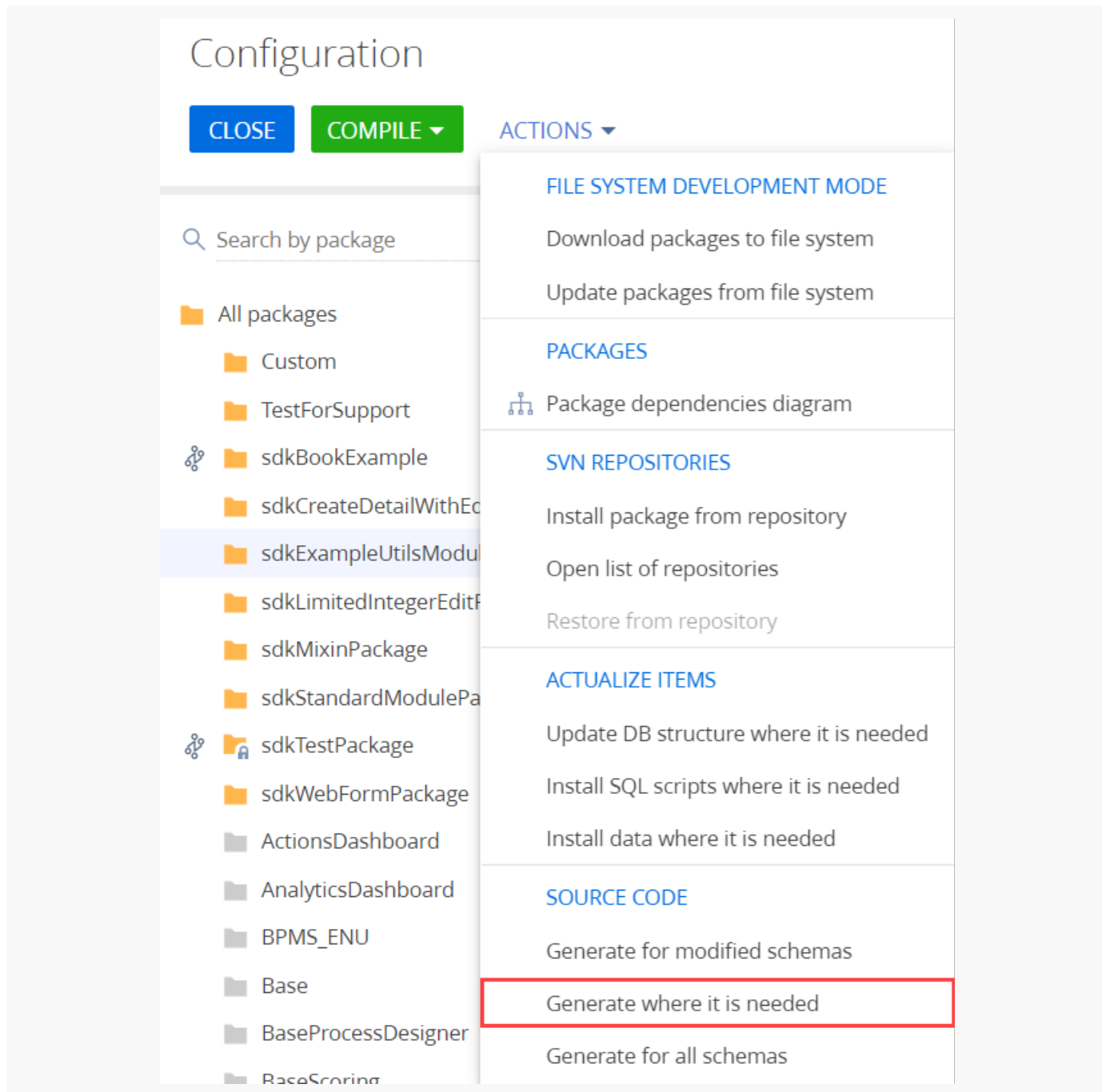
Пакет в области работы с пакетами

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

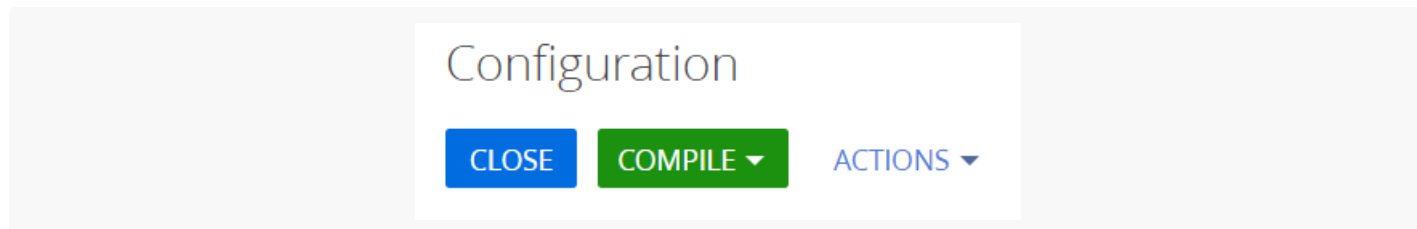
Чтобы **выполнить генерацию исходных кодов**:

1. На панели инструментов в группе действий [ *Исходный код* ] ([ *Source code* ]) выберите [ *Сгенерировать для требующих генерации* ] ([ *Generate where it is needed* ]).



#### 4. Скомпилировать изменения

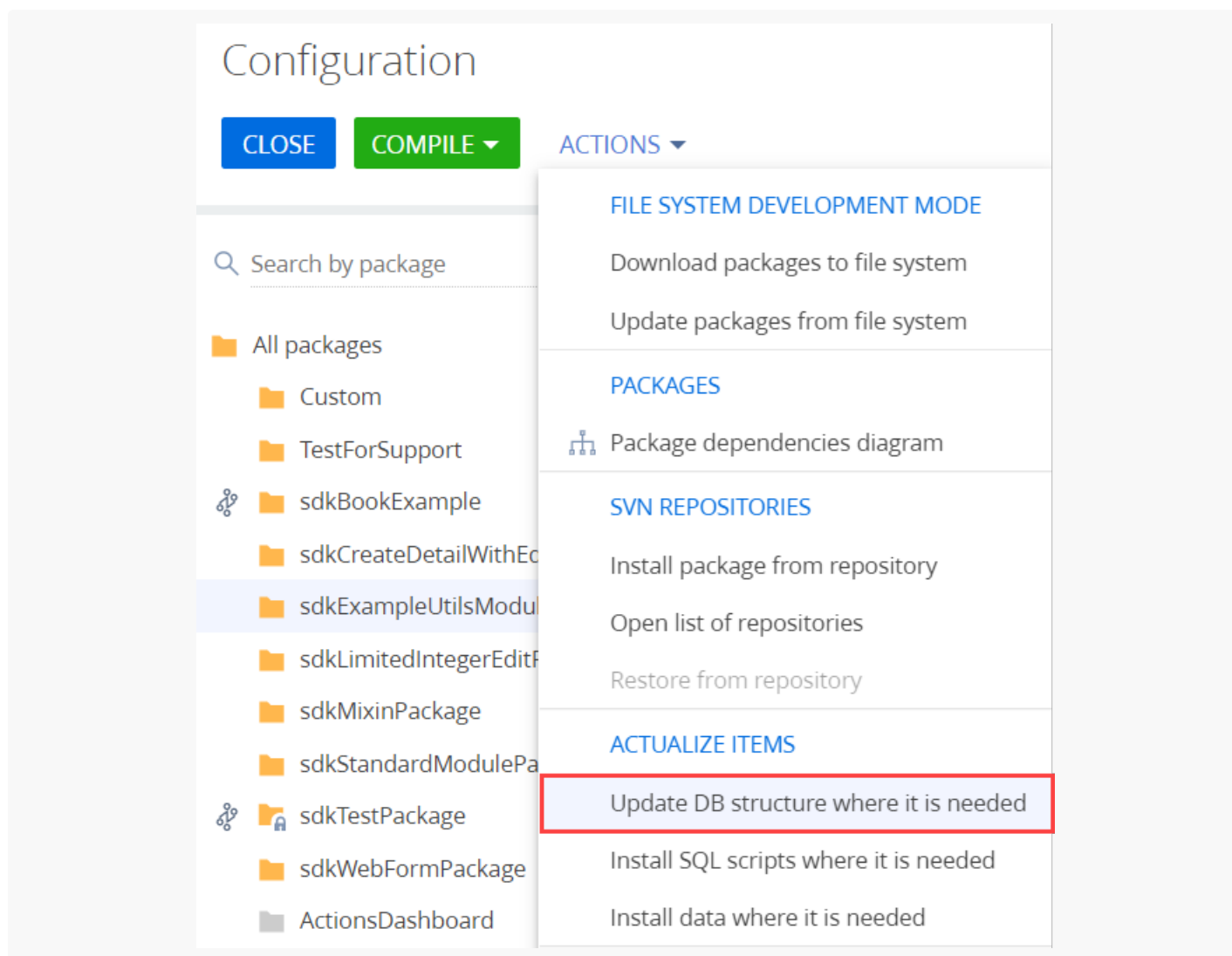
Чтобы **скомпилировать изменения**, на панели инструментов нажмите [ *Компилировать* ] ([ *Compile* ]).



Необходимость обновления структуры базы данных, установки SQL-скриптов и привязанных данных отображается в колонке [ Статус ] ([ Status ]) рабочей области раздела [ Конфигурация ] ([ Configuration ]).

## 5. Обновить структуру базы данных

Чтобы **обновить структуру базы данных**, на панели инструментов в группе действий [ Актуализировать элементы ] ([ Actualize items ]) выберите [ Обновить структуру БД для требующих обновления ] ([ Update DB structure where it is needed ]).



## 6. Установить SQL-сценарии и привязанные данные (опционально)

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

После установки в приложении станет доступна реализованная в пакете функциональность. В нашем примере это функциональность детали с редактируемым реестром.

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

## Автоматически установить пакет из SVN в режиме разработки в файловой системе

### 1. Включить автоматическое применение изменений

Чтобы **включить автоматическое применение изменений**, в файле `..\Terrasoft.WebApp\Web.config` установите значение `true` для ключей элемента `<appSettings>`:

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

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

```
<appSettings>
```


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

Затем выполните шаги 1—2 последовательность действий пункта [Вручную установить пакет из SVN в режиме разработки в файловой системе](#).

## Настроить взаимодействие с хранилищем SVN (опционально)

Creatio позволяет настроить взаимодействие с хранилищем SVN как из раздела [ *Конфигурация* ] ([ *Configuration* ]), так и из файловой системы.

Чтобы **настроить взаимодействие с хранилищем SVN**:

1. Перейдите в дизайнер системы по кнопке .
2. В блоке [ *Конфигурирование разработчиком* ] ([ *Admin area* ]) перейдите по ссылке [ *Управление конфигурацией* ] ([ *Advanced settings* ]).
3. Установите пакет из хранилища SVN. Подробная инструкция содержится в статье [Контроль версий в Creatio IDE](#).
4. Выгрузите пакет в файловую систему. Подробная инструкция содержится в статье [Настроить Creatio для работы в файловой системе](#).

Затем выполните шаги 3—6 последовательность действий пункта [Вручную установить пакет из SVN в режиме разработки в файловой системе](#).

## Привязать к SVN не связанный с хранилищем пакет



Может возникнуть задача, когда пользователю необходимо привязать к хранилищу пакет с файловой системы. Выполнить эту привязку можно только в on-site приложении Creatio. После привязки пакета к SVN в файловой системе его можно [установить](#), используя встроенные средства Creatio.

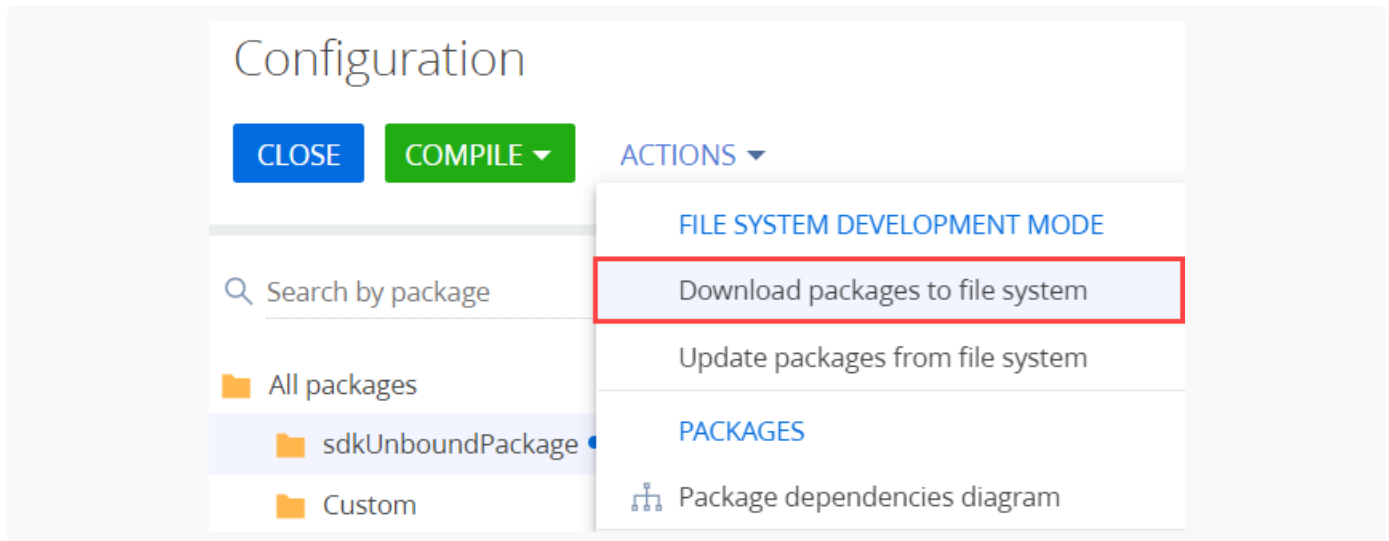
**Пример.** Привязать к хранилищу не связанный с хранилищем SVN существующий пользовательский пакет `sdkUnboundPackage`.

Адрес пакета в хранилище SVN — `.../SDKPackages/sdkUnboundPackage`.

# 1. Выгрузить пакет в файловую систему

Чтобы **выгрузить пакет в файловую систему**:

1. Настройте Creatio для работы в файловой системе. Настройка описана в статье [Внешние IDE](#).
2. На панели инструментов в группе действий [ *Разработка в файловой системе* ] ([ *File system development mode* ]) выберите [ *Выгрузить все пакеты в файловую систему* ] ([ *Download packages to file system* ]).



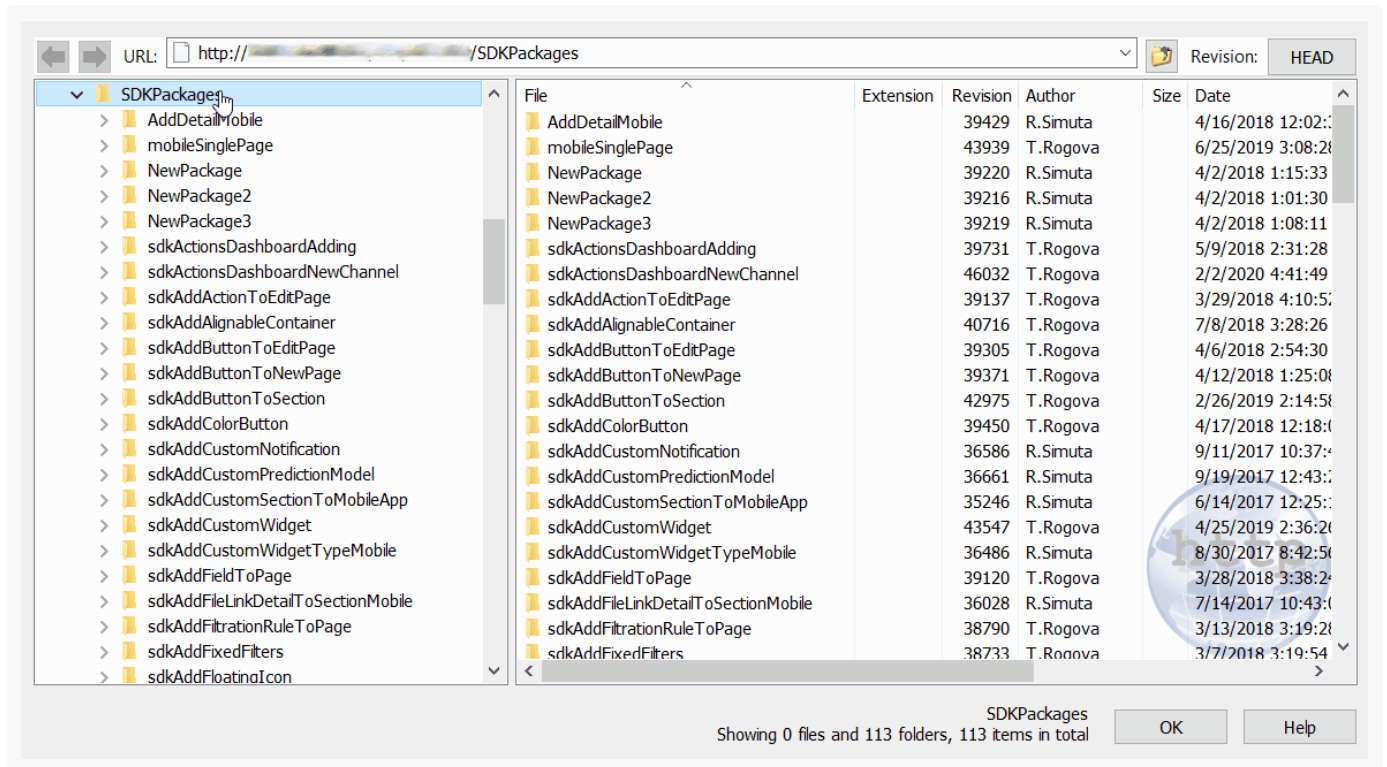
В результате все пакеты будут выгружены по пути `..\Terrasoft.WebApp\Terrasoft.Configuration\Pkg` в каталог с соответствующим названием пакета.

# 2. Создать каталоги для пакета в хранилище SVN

Чтобы создать каталоги для пакета в хранилище SVN необходимо использовать клиентское приложение для работы с SVN (например, [TortoiseSvn](#)).

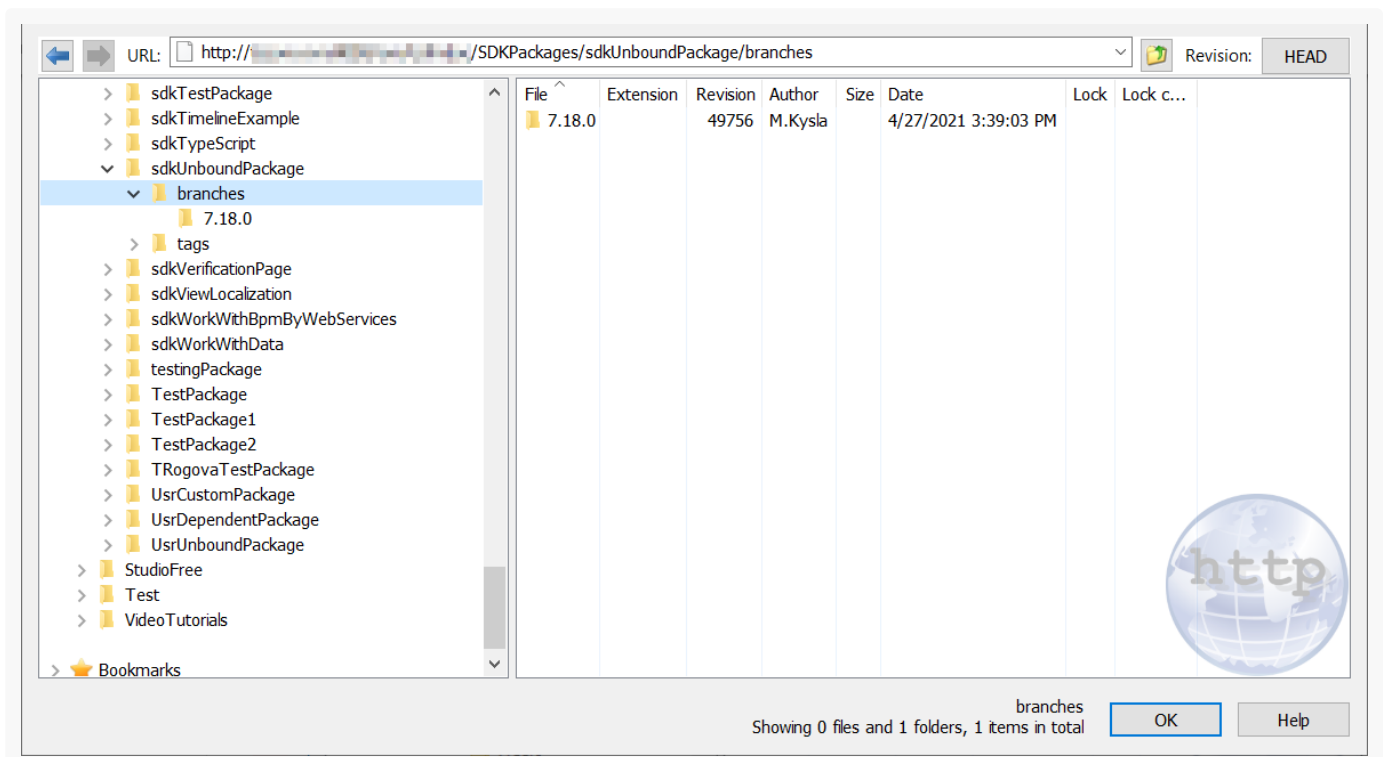
Чтобы **создать каталоги** для пакета в хранилище SVN:

1. Перейдите в репозиторий, указанный в свойствах пакета.
2. В репозитории создайте каталог, название которого совпадает с названием созданного в приложении пакета. В нашем примере это `sdkUnboundPackage`.



3. В созданном каталоге создайте подкаталоги `branches` и `tags`.

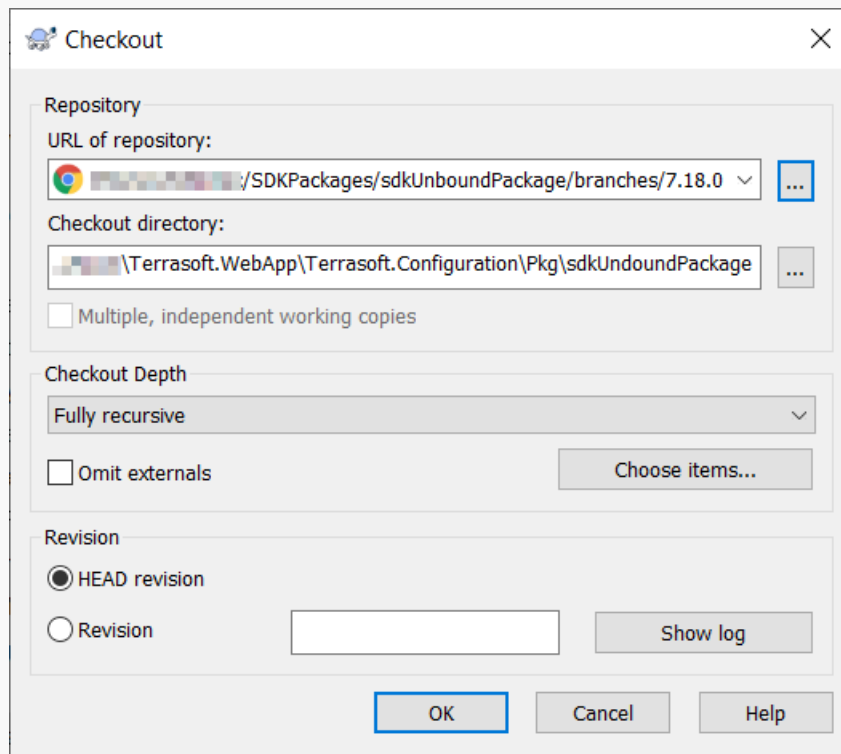
4. В каталоге `branches` создайте каталог, название которого совпадает с номером версии пакета — `7.18.0`.



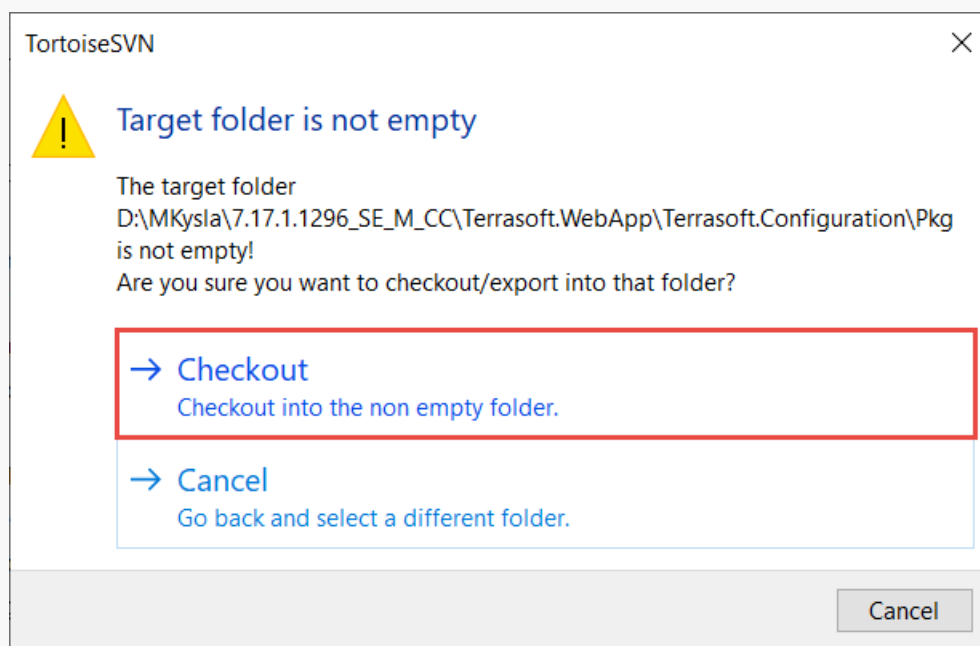
### 3. Создать рабочую копию версионной ветки пакета

Чтобы **создать рабочую копию версии ветки пакета**:

1. Выгрузите (команда `SVN Checkout...`) созданный на предыдущем шаге каталог из хранилища в каталог пакета в файловой системе. В нашем примере это каталог `7.18.0`.



2. Подтвердите выгрузку в существующий каталог.

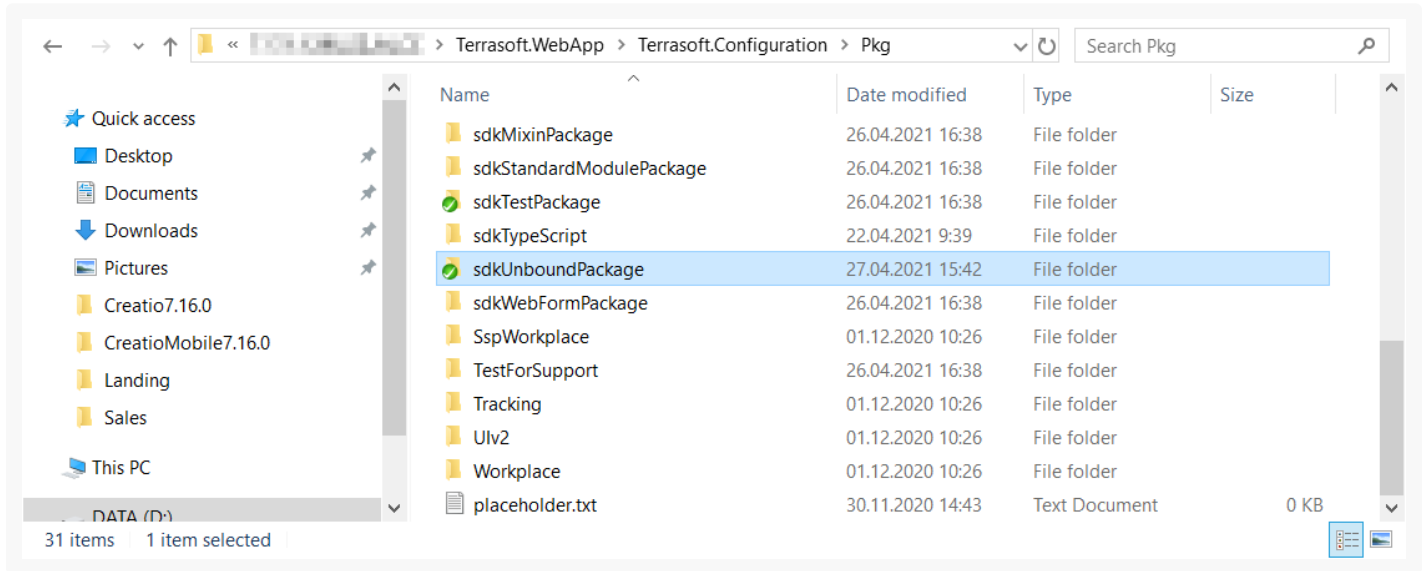


В результате каталог пакета в файловой системе

`...\\Terrasoft.WebApp\\Terrasoft.Configuration\\Pkg\\sdkUnboundPackage` будет связан с веткой версии `7.18.0`



пакета в хранилище.

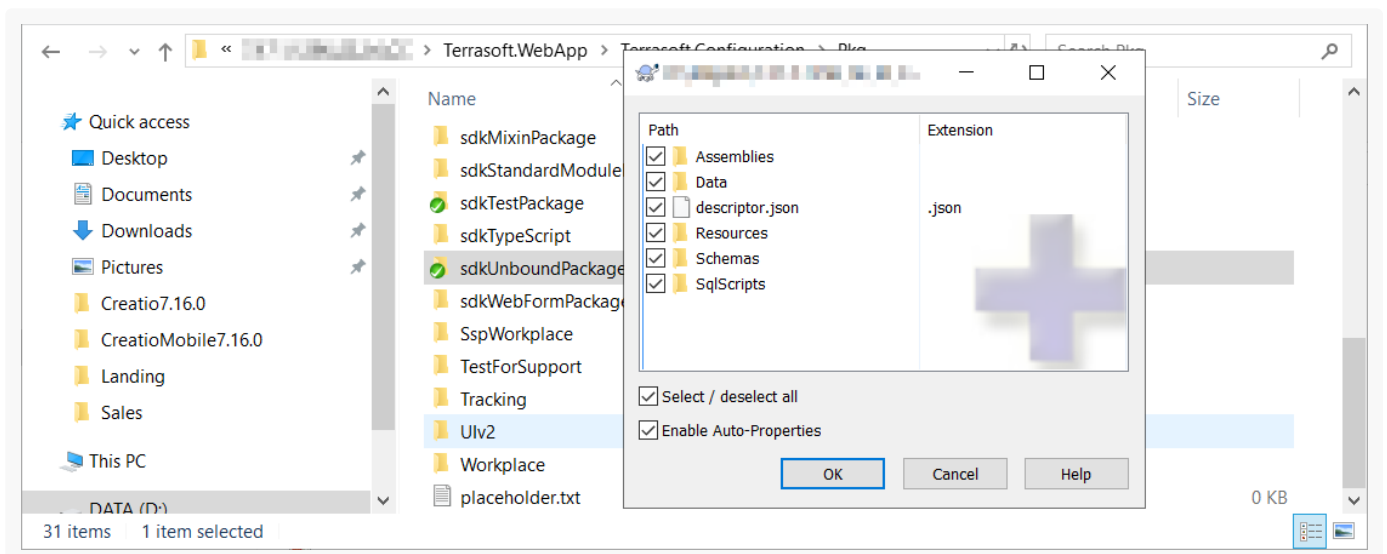


## 4. Зафиксировать в хранилище каталог пакета

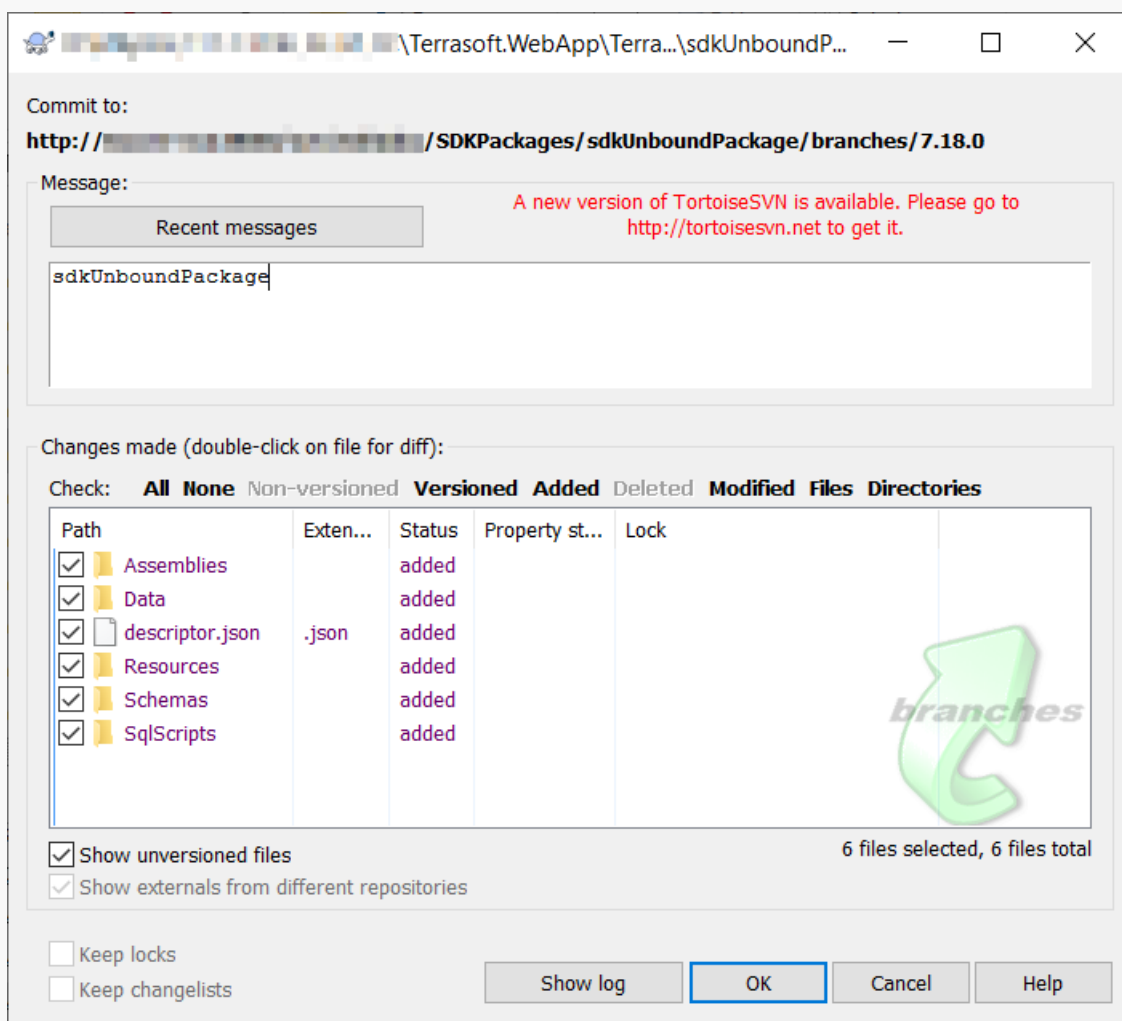
Чтобы **зафиксировать в хранилище каталог пакета**:

1. Добавьте в хранилище содержимое каталога

... \Terrasoft.WebApp\Terrasoft.Configuration\Pkg\sdkUnboundPackage .



2. Выполните фиксацию каталога в хранилище.



# Привязать к SVN не связанный с хранилищем пакет через запрос к базе данных

 Средний


**Пример.** Привязать к хранилищу не связанный с хранилищем SVN существующий пользовательский пакет `sdkUnboundPackage` через прямой запрос к базе данных.

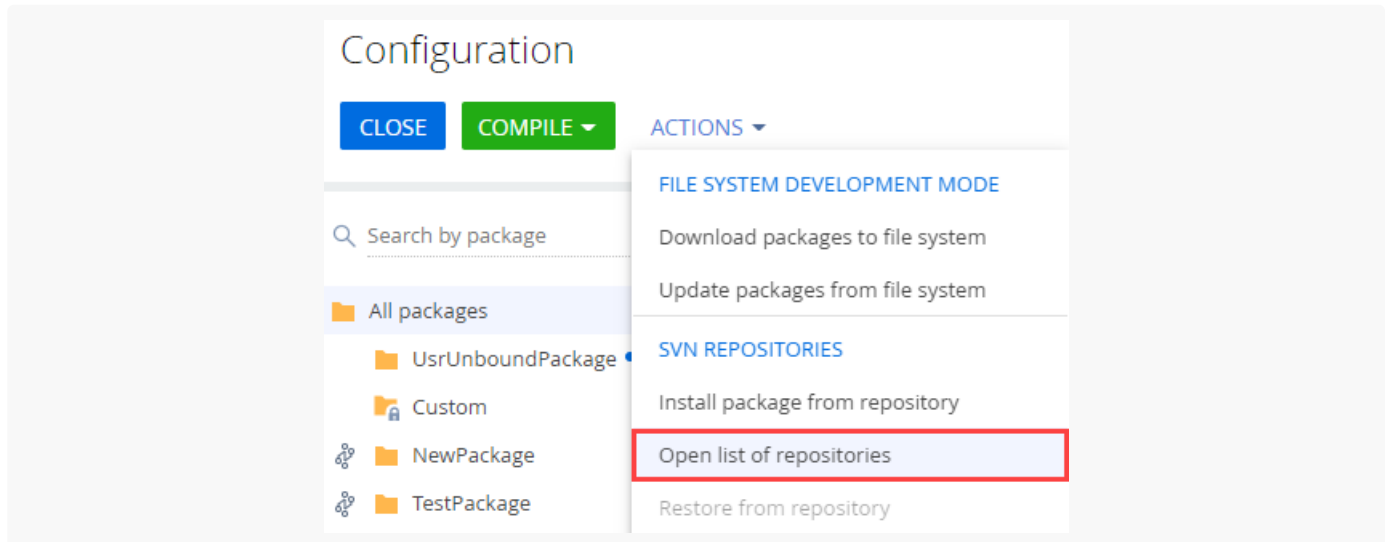
Адрес пакета в хранилище SVN — `.../SDKPackages/sdkUnboundPackage`.

## 1. Подключите к приложению хранилище SVN

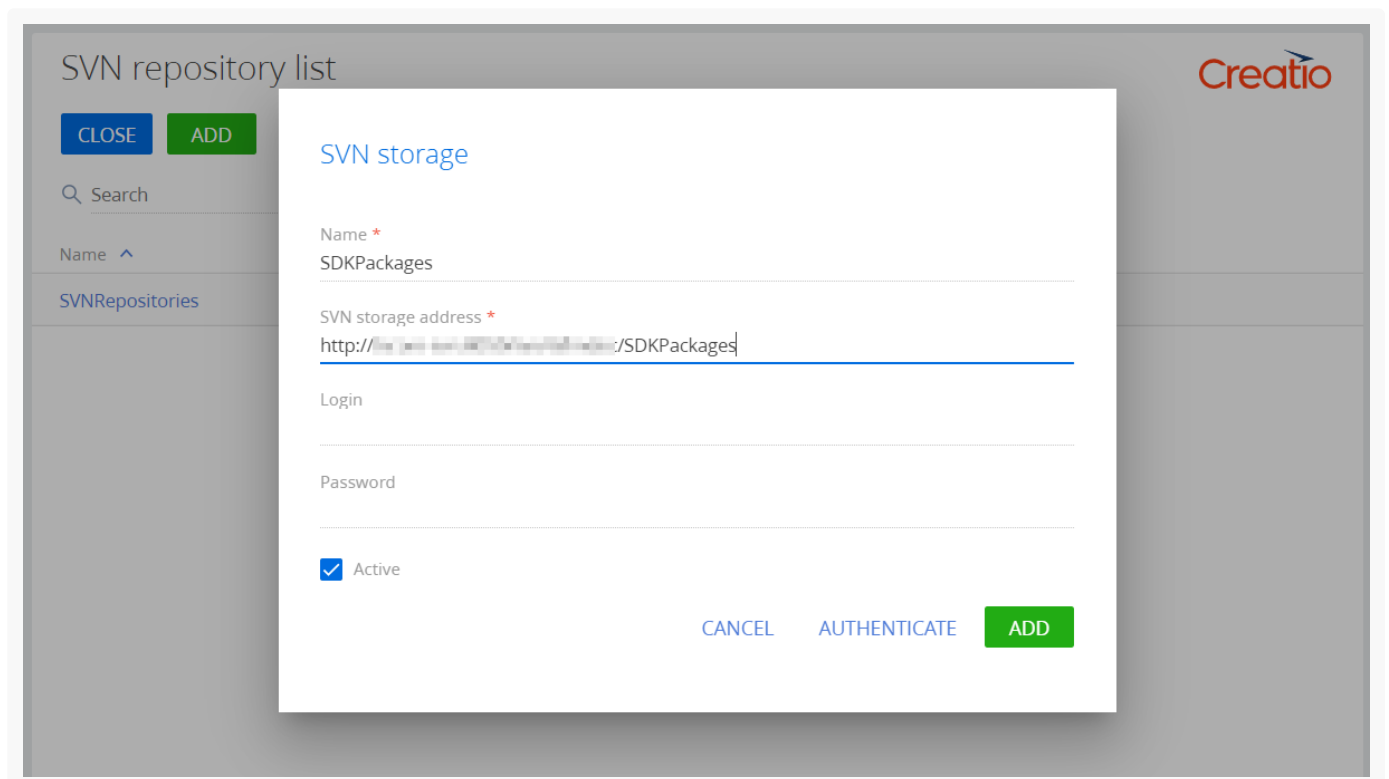
Если хранилище SVN подключено к приложению, то перейдите к следующему шагу.

Чтобы **подключить к приложению хранилище SVN**:

1. Перейдите в дизайнер системы по кнопке .
2. В блоке [ Конфигурирование разработчиком ] ([ Admin area ]) перейдите по ссылке [ Управление конфигурацией ] ([ Advanced settings ]).
3. На панели инструментов в группе действий [ Хранилища SVN ] ([ SVN repositories ]) выберите [ Открыть список хранилищ ] ([ Open list of repositories ]).



4. Нажмите [ Добавить ] ([ Add ]) и добавьте репозиторий. В нашем примере это `.../SDKPackages`.



5. Нажмите [ Аутентификация ] ([ Authenticate ]) и выполните аутентификацию.

## 2. Привязать хранилище SVN к пакету

Чтобы **привязать хранилище SVN к пакету**:

1. Перейдите в базу данных приложения.
2. В базе данных выполните SQL-запрос.

#### Пример SQL-запроса

```
UPDATE SysPackage
SET
    [SysRepositoryId] =
    (
        select top 1 Id from SysRepository
        where Name = 'SDKPackages' -- Название хранилища.
    )
where [Name]='sdkUnboundPackage' -- Название пользовательского пакета.
```

Этот запрос позволяет:

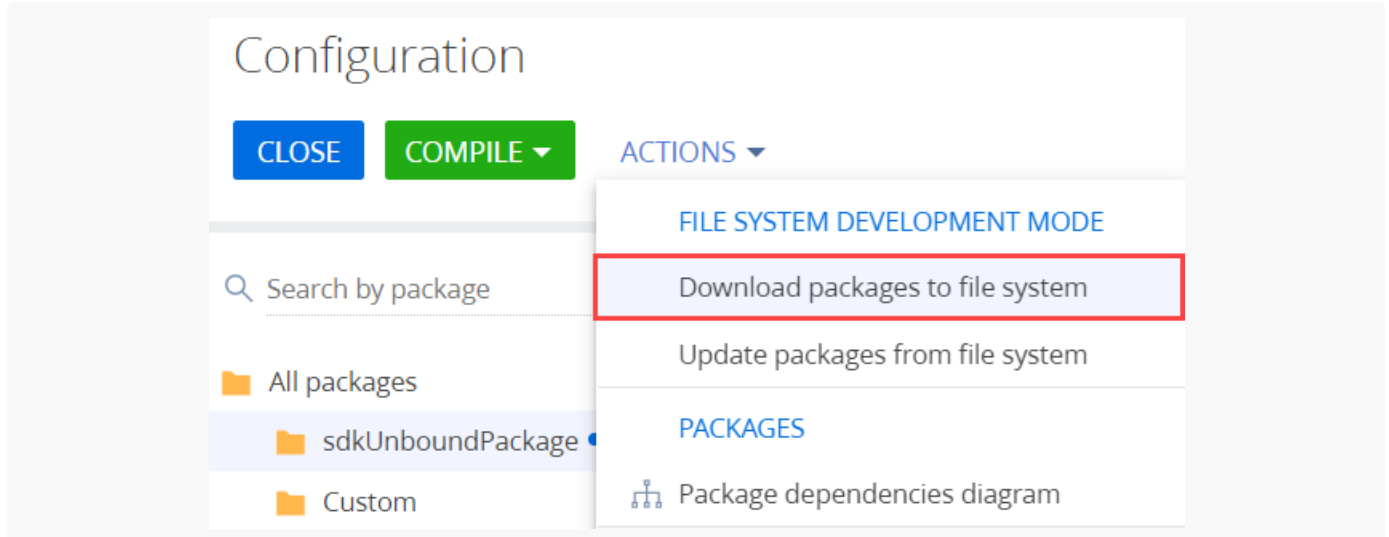
- Из таблицы [SysRepository] считать идентификатор записи, содержащей адрес хранилища SVN.
- Добавить полученный идентификатор в колонку [SysRepositoryId] записи, которая содержит имя не привязанного к хранилищу SVN пакета, таблицы [SysPackage] .

Чтобы изменения применились в приложении, необходимо выйти из приложения и зайти повторно.

## 3. Выгрузить пакет в файловую систему

Чтобы **выгрузить пакет** в файловую систему:

1. Настройте Creatio для работы в файловой системе. Настройка описана в статье [Внешние IDE](#).
2. На панели инструментов в группе действий [ *Разработка в файловой системе* ] ([ *File system development mode* ]) выберите [ *Выгрузить все пакеты в файловую систему* ] ([ *Download packages to file system* ]).



В результате все пакеты будут выгружены по пути `..\Terrasoft.WebApp\Terrasoft.Configuration\Pkg` в каталог с соответствующим названием пакета.

# 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`.

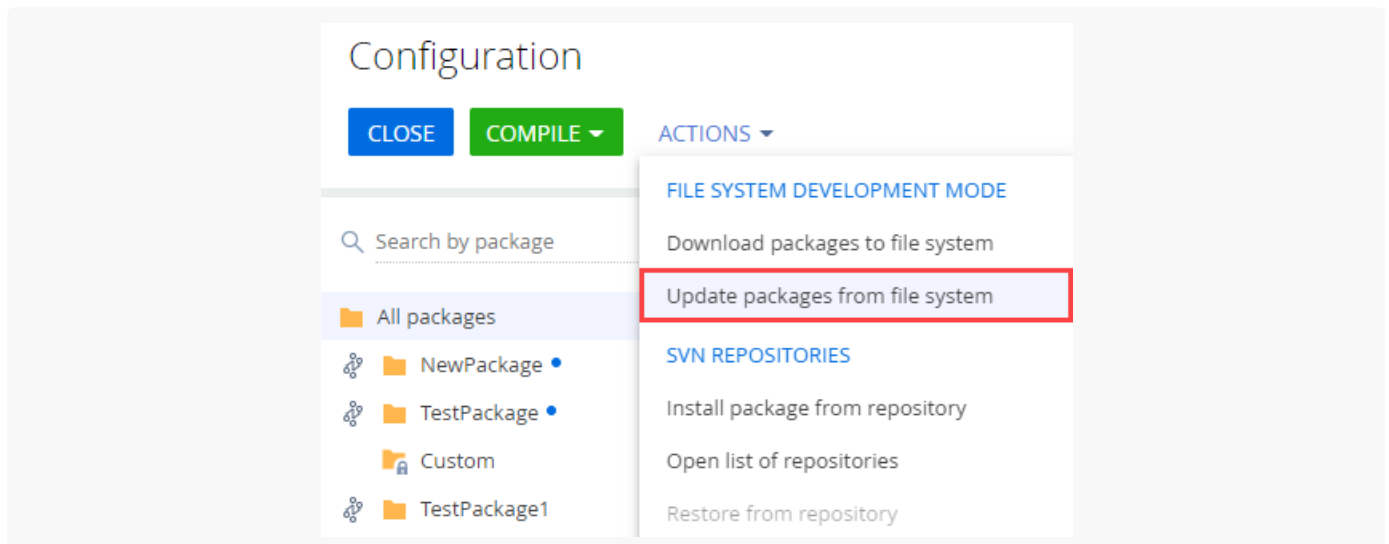
**Схема** `UsrGreetingService`

```

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

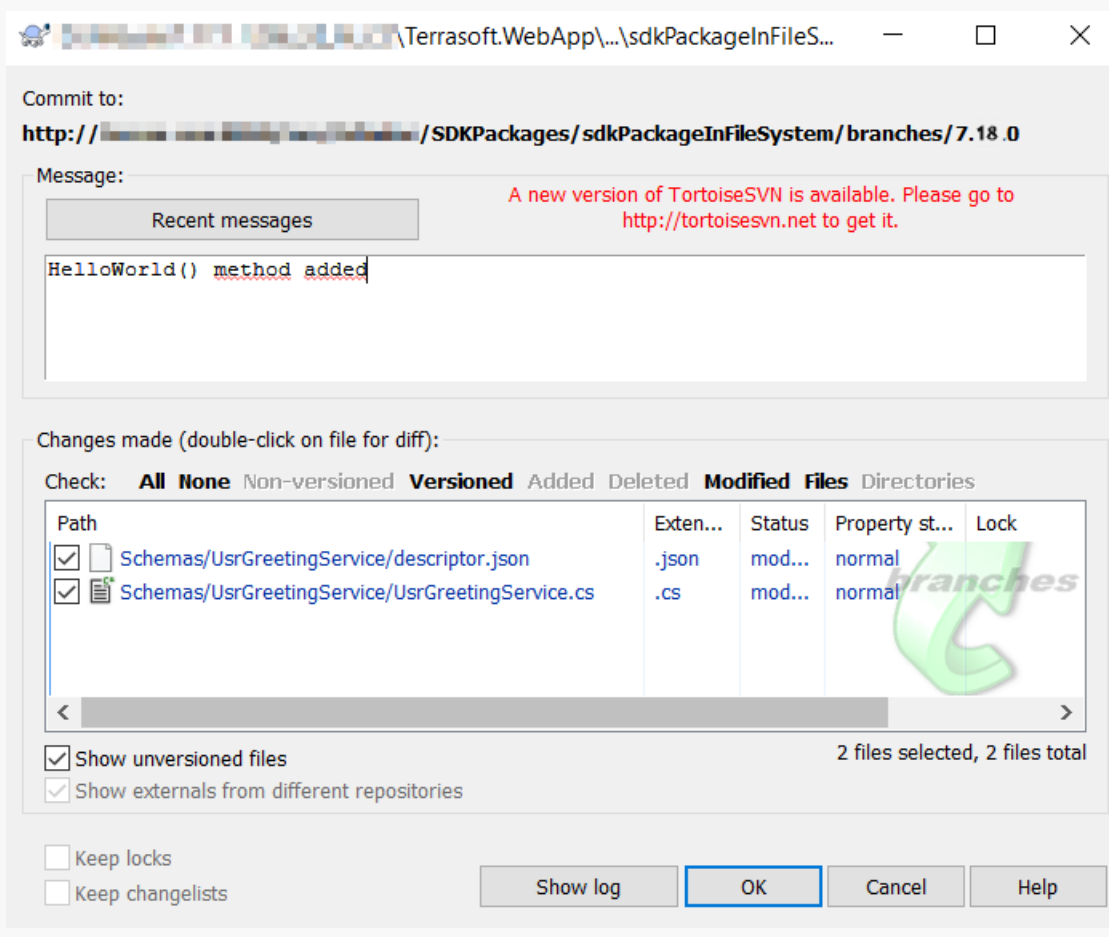
```
namespace Terrasoft.Configuration
{
    using System.ServiceModel;
    using System.ServiceModel.Activation;
    using System.ServiceModel.Web;
    [ServiceContract]
    [AspNetCompatibilityRequirements(RequirementsMode = AspNetCompatibilityRequirementsMode.RequirementsMode.None)]
    public class 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

**Пример.** Создать пользовательский пакет `sdkPackageForFileSystem`, 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.

Приведенный в этой статье пример требует четкого понимания **разницы между режимами разработки**.

Общие **рекомендации**:

- В режиме разработки в файловой системе работать с хранилищем SVN следует только из файловой системы.
- В режиме разработки с помощью встроенных средств работать с SVN нужно только встроенными средствами раздела [ *Конфигурация* ] ([ *Configuration* ]).



## 1. Задать путь к каталогу для рабочих копий пакетов

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 содержимое пользовательского пакета, выгруженное в файловую систему, не будет привязано к хранилищу SVN, даже если пакет был привязан к хранилищу в разделе [ *Конфигурация* ] ([ *Configuration* ]).

Если при создании пакета заполнить поле [ *Хранилище системы контроля версий* ] ([ *Version control system repository* ]) 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 в файле `ConnectionStrings.config` с помощью настройки `defPackagesWorkingCopyPath`. 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.

Чтобы **задать путь к каталогу для рабочих копий пакетов**, в настройке `defPackagesWorkingCopyPath` файла `ConnectionStrings.config` укажите полный путь к каталогу `..\Terrasoft.WebApp\Terrasoft.Configuration\Pkg`.



### Пример файла `ConnectionStrings.config`

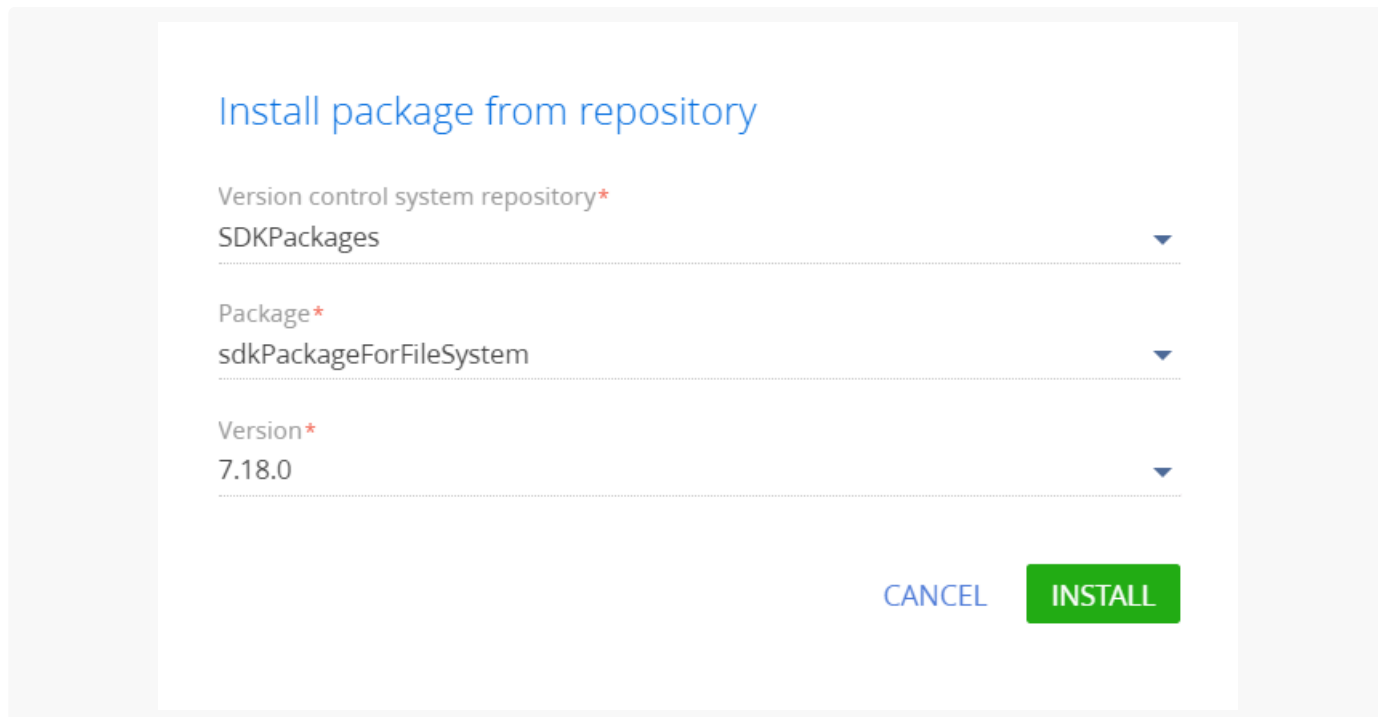
```
<?xml version="1.0" encoding="utf-8"?>
<connectionStrings>
  ...
  <add name="defPackagesWorkingCopyPath" connectionString="C:\creatio\Terrasoft.WebApp\Terrasoft
  ...
</connectionStrings>
```

Это изменение позволяет совместить каталог, в котором содержатся рабочие копии пользовательских пакетов, с каталогом, в который выгружаются пакеты в режиме разработки в файловой системе.

## 2. Создать пакет

Чтобы **создать пользовательский пакет**:

1. Перейдите в дизайнер системы по кнопке .
2. В блоке [ *Конфигурирование разработчиком* ] ([ *Admin area* ]) перейдите по ссылке [ *Управление конфигурацией* ] ([ *Advanced settings* ]).
3. В области работы с пакетами нажмите кнопку .
4. Заполните **свойства пакета**:



- [ *Название* ] ([ *Name* ]) — "sdkPackageForFileSystem".
- [ *Хранилище системы контроля версий* ] ([ *Version control system repository* ]) — "SDKPackages".
- [ *Версия* ] ([ *Version* ]) — "7.18.0".

5. Нажмите кнопку [ *Создать и добавить зависимости* ] ([ *Create and add dependencies* ]) и установите зависимости пакета.

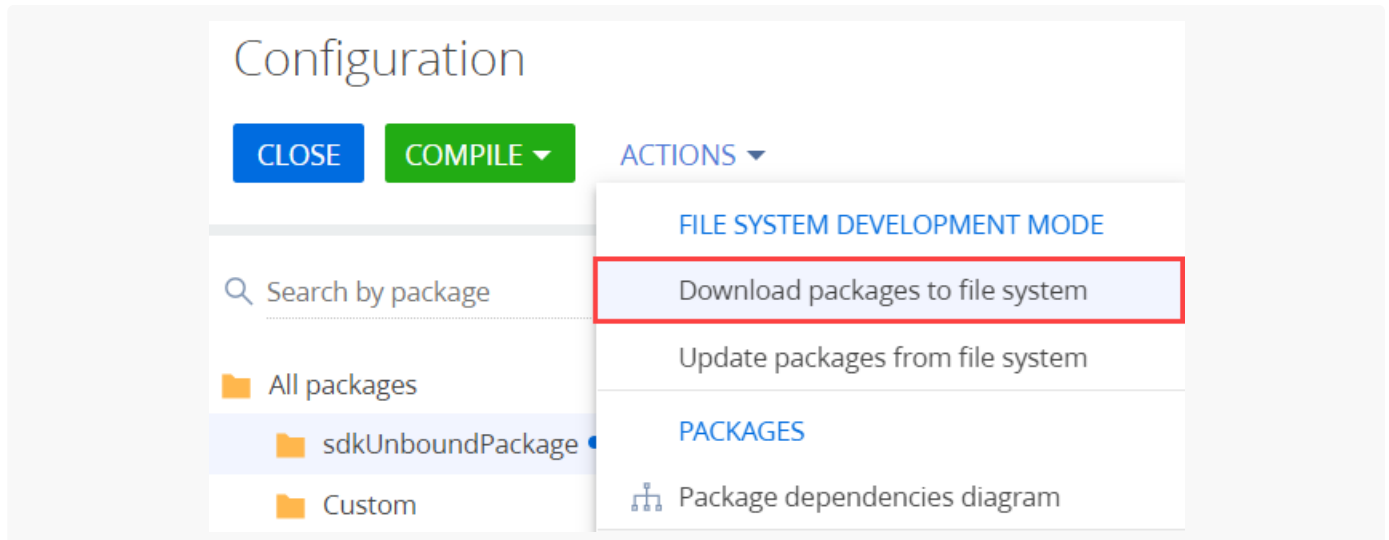
В результате пакет будет автоматически зафиксирован в SVN, а в каталоге

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

### 3. Выгрузить пакет в файловую систему

Чтобы **выгрузить пакет** в файловую систему:

1. Настройте Creatio для работы в файловой системе. Настройка описана в статье [Внешние IDE](#).
2. На панели инструментов в группе действий [ *Разработка в файловой системе* ] ([ *File system development mode* ]) выберите [ *Выгрузить все пакеты в файловую систему* ] ([ *Download packages to file system* ]).



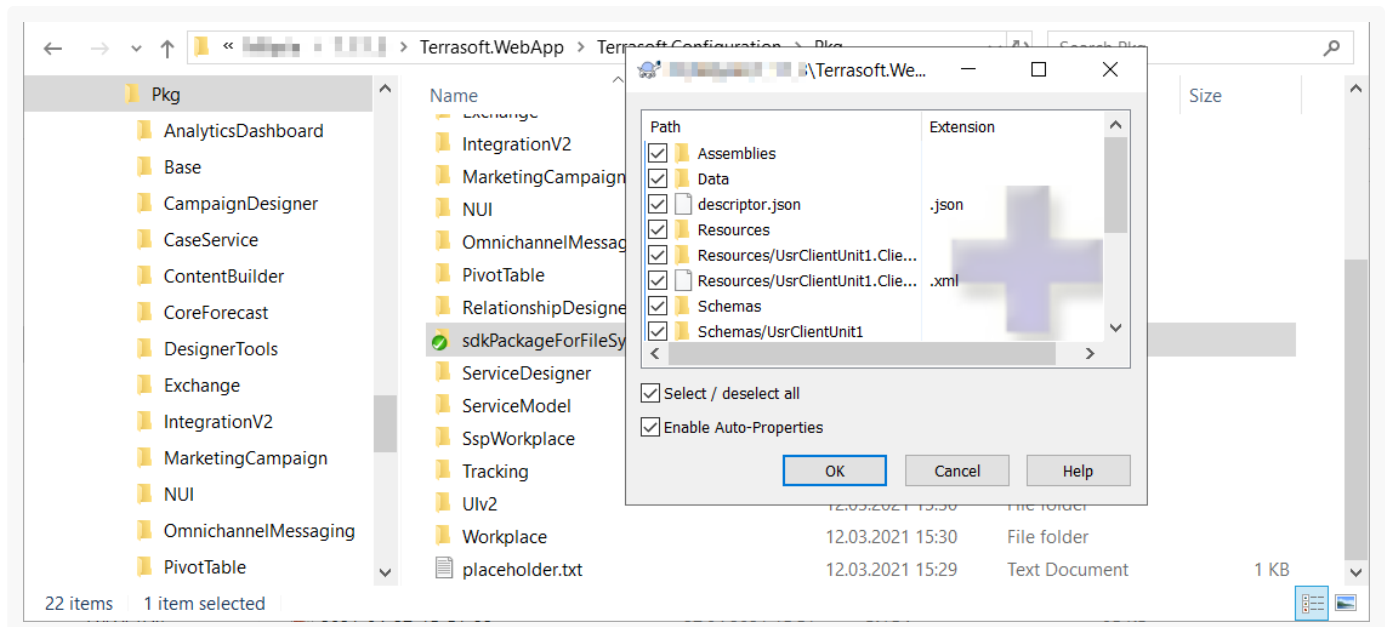
В результате все пакеты будут выгружены по пути `..\Terrasoft.WebApp\Terrasoft.Configuration\Pkg` в каталог с соответствующим названием пакета.

## 4. Зафиксировать пакет в хранилище

Чтобы **зафиксировать пакет в хранилище**:

1. Добавьте (команда `SVN Commit`) в хранилище содержимое каталога

`...\Terrasoft.WebApp\Terrasoft.Configuration\Pkg\sdkPackageForFileSystem`.



2. Выполните фиксацию каталога в хранилище.

Commit to:  
[http://\[redacted\]/SDKPackages/sdkPackageForFileSystem/branches/7.18.0](http://[redacted]/SDKPackages/sdkPackageForFileSystem/branches/7.18.0)

Message:  

A new version of TortoiseSVN is available. Please go to <http://tortoisesvn.net> to get it.

`sdKPackageForFileSystem`

Changes made (double-click on file for diff):

Check: **All** **None** Non-versioned **Versioned** **Added** Deleted **Modified** **Files** **Directories**

Path	Exten...	Status	Property st...	Lock
<input checked="" type="checkbox"/> Assemblies		added		
<input checked="" type="checkbox"/> Data		added		
<input checked="" type="checkbox"/> descriptor.json	.json	added		
<input checked="" type="checkbox"/> Resources		added		
<input checked="" type="checkbox"/> Resources/UsrClientUnit1.ClientUnit		added		
<input checked="" type="checkbox"/> Resources/UsrClientUnit1.ClientUnit/resource.en-US.xml	.xml	added		
<input checked="" type="checkbox"/> Schemas		added		

☒ Show unversioned files 13 files selected, 13 files total

☒ Show externals from different repositories

☐ Keep locks

☐ Keep changelists