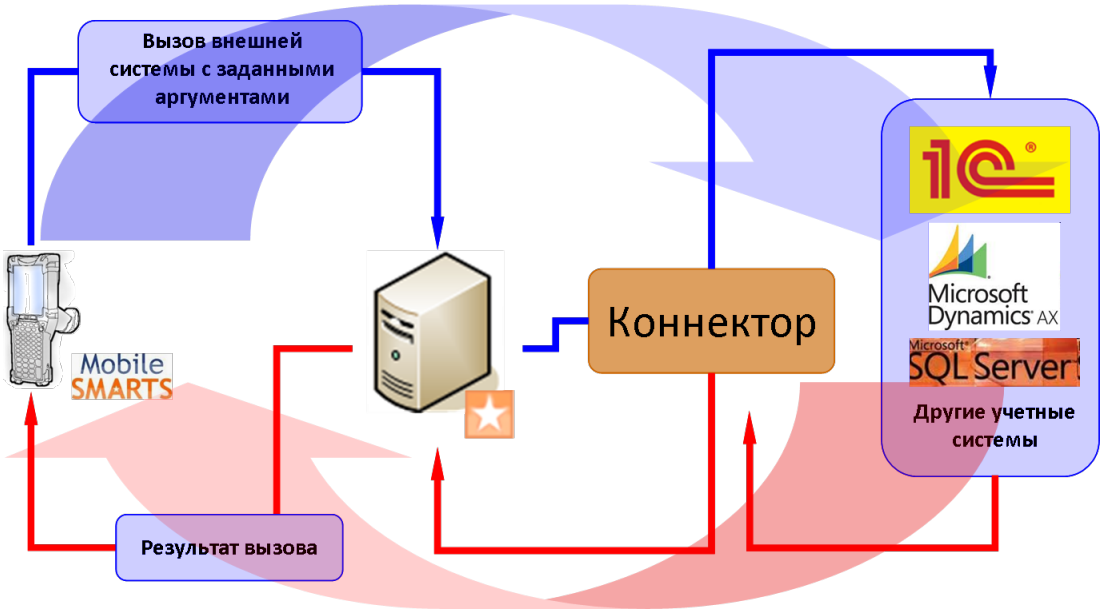
Основной принцип работы коннекторов

Последние изменения: 2024-03-26

Обращение к внешней системе с передачей и получением каких-то полезных данных происходит следующим образом:



- В клиенте Mobile SMARTS на ТСД действие «Вызов метода внешней системы» формирует аргументы для вызова внешней системы, упаковывает их в объект Cleverence.Warehouse.InvokeArgs и отправляет на сервер Mobile SMARTS вместе с именем коннектора, именем класса и именем метода;
- Сервер Mobile SMARTS ищет коннектор с требуемым именем в списке зарегистрированных и передает ему аргументы вместе с именем класса и метода для вызова;
- Коннектор осуществляет вызов указанного метода внешней системы и получает результат;
- Результат возвращается серверу Mobile SMARTS, упаковывается в объект Cleverence.Warehouse.InvokeResult и отправляется на ТСД;
- В клиенте Mobile SMARTS на ТСД действие «Вызов метода внешней системы» кладет полученный результат в текущую сессию под именем, которое задано у него в свойстве «Переменная сессии для результата»;
- Теперь любое другое действие в программе на ТСД может получить доступ к результату вызова для вывода его на экран или выполнения расчетов.

коннекторы

Не нашли что искали?

Задать вопрос в техническую поддержку

Список существующих стандартных коннекторов

Последние изменения: 2024-03-26

В состав дистрибутива Mobile SMARTS входят следующие стандартные коннекторы для сервера Mobile SMARTS:

Название
Необходимые файлы
Описание
1С Предприятие версия 7.7
Для панели управления: не требуется Для сервера: Cleverence.Connectivity.OneC.dll
Позволяет производить вызов функции из глобального модуля конфигурации 1С Предприятие 7.7 по имени и возвращать результат её выполнения.
1С Предприятие версия 8
Для панели управления: не требуется Для сервера: Cleverence.Connectivity.OneC.dll
Позволяет производить вызов функции из модуля указанной обработки 1С (из состава конфигурации 1С или внешней) по имени и возвращать результат её выполнения. Обеспечивает работу как через внешнее COM-соединение 1С, так и через Web-сервис 1С. Поддерживаются платформы 1С 8.1, 8.2, 8.3.

АРМ ЕГАИС - Серверный коннектор
Для панели управления: Cleverence.Connectivity.ArmEgais.Panel.dll Для сервера: Cleverence.Connectivity.ArmEgais.dll
Коннектор для обмена документами с системой АРМ ЕГАИС. Предназначен для обработки событий сервера Mobile SMARTS (получение документов он-лайн, отправка документов).

Axapta5Connector (Axapta 2009)
Для панели управления: не требуется Для сервера: Cleverence.Connectivity.Axapta5.dll
Позволяет вызывать статические методы классов Axapta и возвращать результат выполнения. Работа выполняется через .NET Axapta Business Connector

Microsoft SQL Server
Для панели управления: не требуется Для сервера: Cleverence.Connectivity.SqlServer.dll
Позволяет выполнять SQL-запросы к базе данных, а также вызывать хранимые процедуры, возвращает результаты запросов, хранимых процедур.

коннекторы

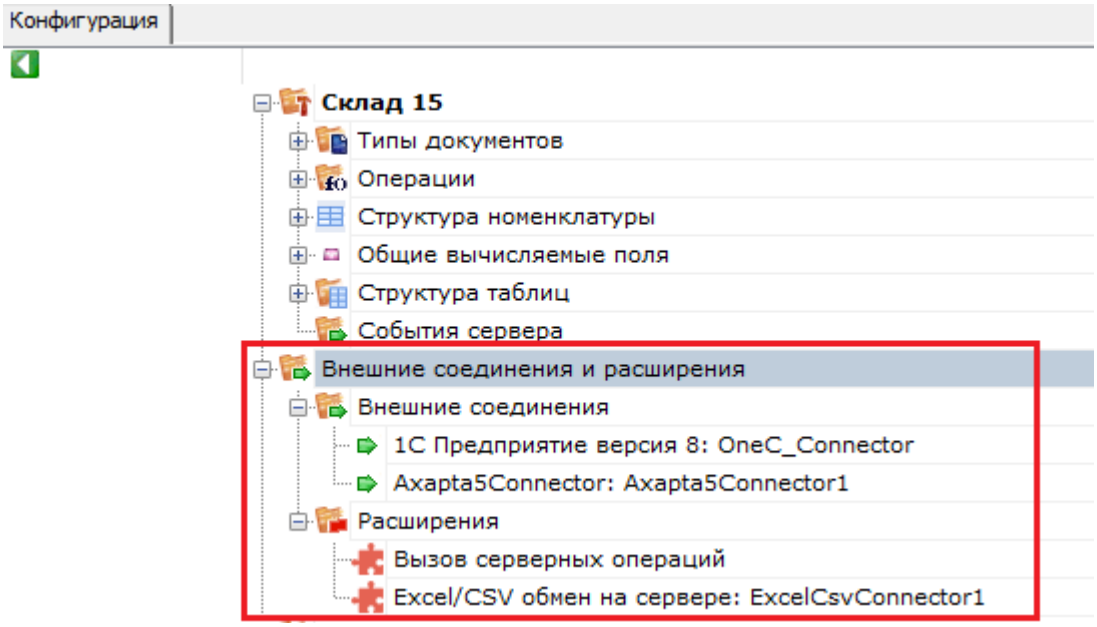
Не нашли что искали?

Задать вопрос в техническую поддержку

Регистрация и запуск коннекторов

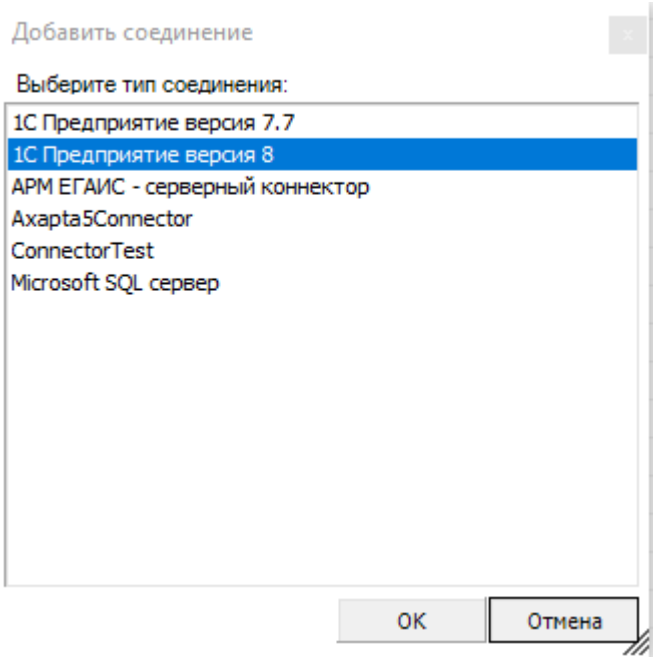
Последние изменения: 2024-03-26

Для того, чтобы начать использовать коннектор, необходимо сначала его зарегистрировать (добавить), а затем запустить. Эти операции выполняются в панели управления Mobile SMARTS и находятся внутри узла «Внешние соединения и расширения».

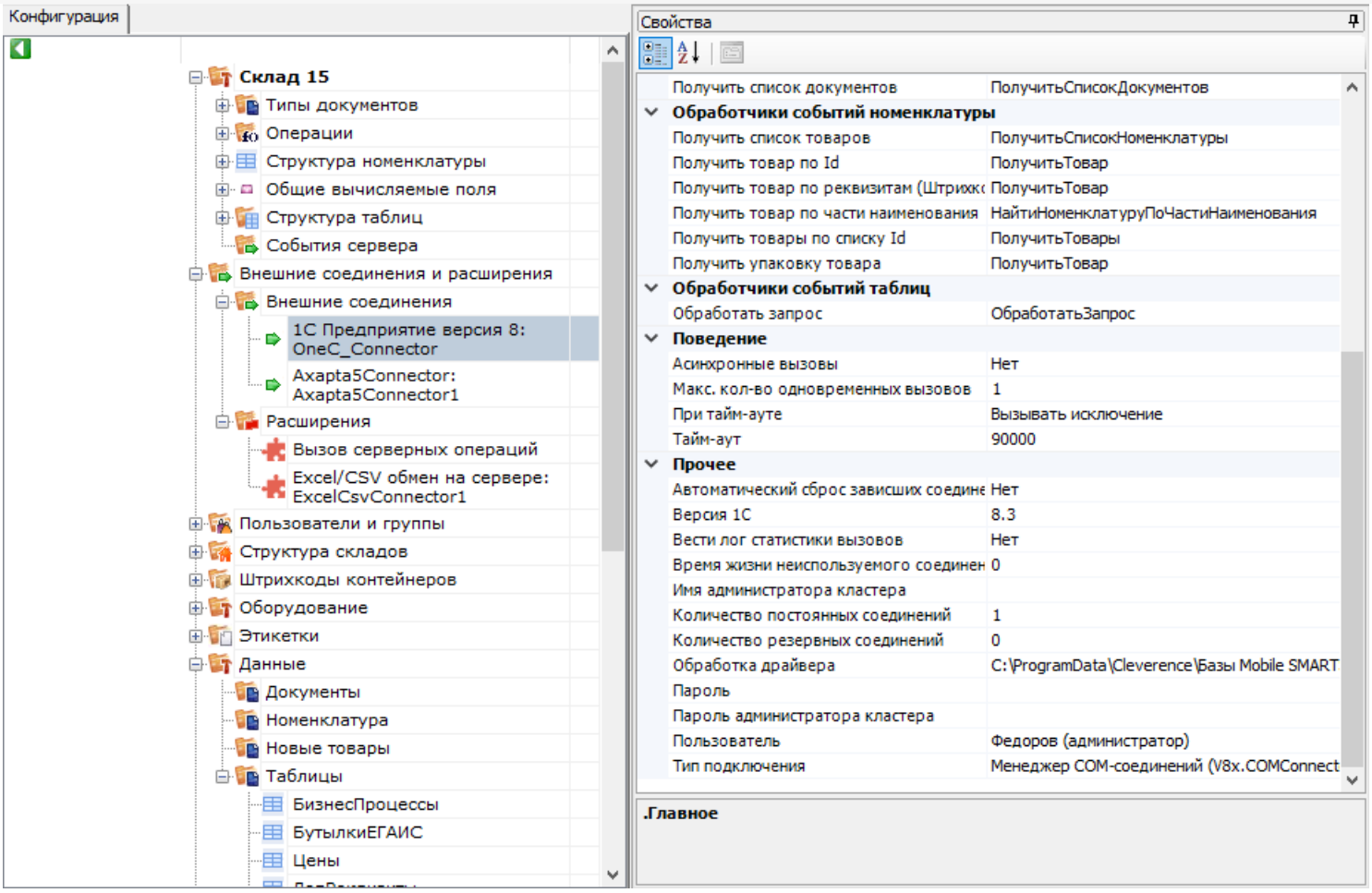


Регистрация коннектора

Добавление в конфигурацию нового коннектора выполняется с помощью щелчка правой кнопкой мыши на узле «Внешние соединения», в контекстном меню нужно выбрать «Добавить внешнее соединение...», в списке выбрать нужный модуль и нажать ОК:

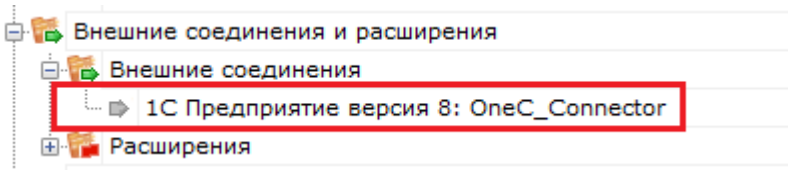


Настройка параметров коннектора выполняется через боковую панель свойств:

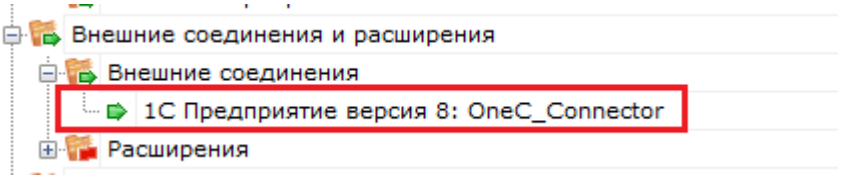


После добавления коннектора и настройки параметров нужно сохранить конфигурацию.

Если узел коннектора в дереве конфигурации отображается в виде серой стрелки, то это означает, что вызовы через коннектор запрещены (отключены):

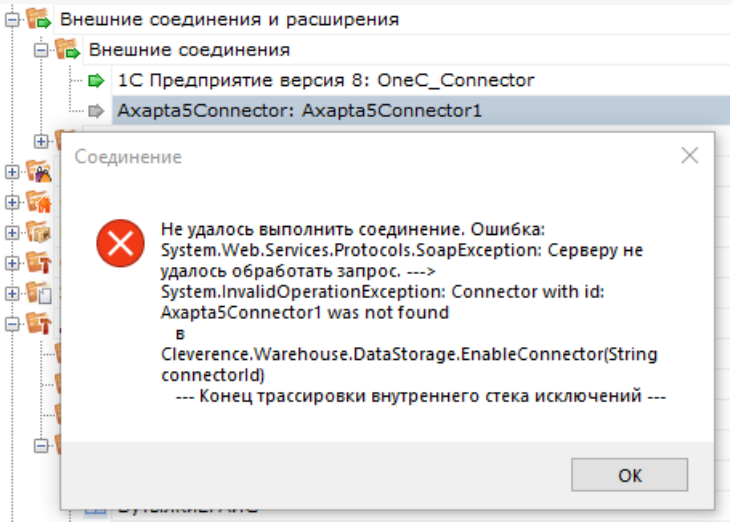


Для того, чтобы разрешить вызовы, нажмите правой кнопкой мыши на узле дерева, выберите в контекстном меню «Разрешить». Модуль должен перейти в разрешенное состояние (зеленая стрелка):

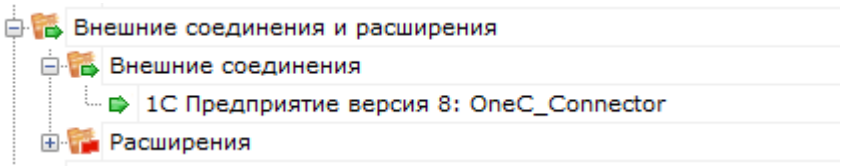


Для работы модуля под управлением сервера Mobile SMARTS и настройки параметров модуля через панель управления используются две версии библиотеки (файлы dll). Первая dll, предназначенная для сервера, размещается в <Папка базы Mobile SMARTS>\Server\DataService\bin\. Вторая dll, для панели управления - в <Папка базы Mobile SMARTS>\Control panel\Addins. Некоторые стандартные модули, входящие в дистрибутив платформы Mobile SMARTS, не требуют дополнительных файлов dll (см. Список существующих стандартных модулей).

Если библиотека модуля для панели управления загружена, а для сервера библиотека отсутствует (или не была загружена в процесс сервера Mobile SMARTS), то при сохранении конфигурации Mobile SMARTS после добавления модуля в панели управления данные о добавленном модуле не запишутся в файлы конфигурации. При попытке разрешить вызовы через коннектор будет происходить ошибка.



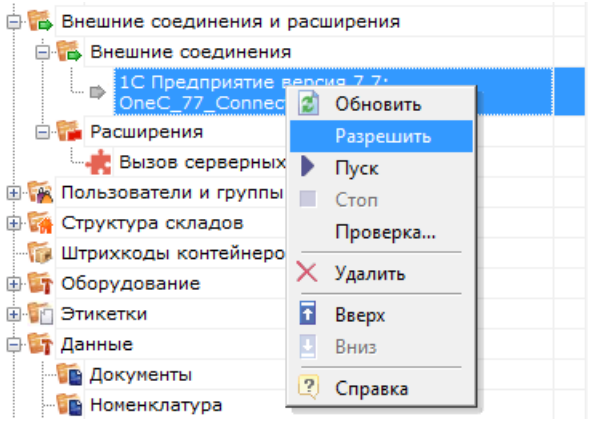
Если загрузить конфигурацию с сервера (используя кнопку «Обновить» в панели управления), отображаются данные о том, что добавленного модуля нет:

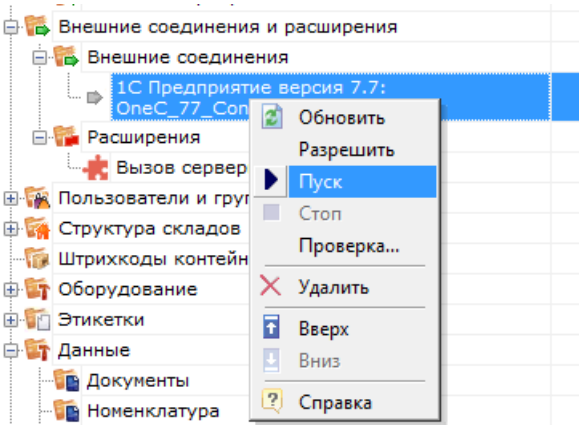


Для устранения ошибки необходимо переместить серверную версию dll модуля в <Папка базы Mobile SMARTS>\Server\DataService\bin\, перезапустить сервер базы данных Mobile SMARTS и заново добавить модуль через панель управления.

Запуск коннектора

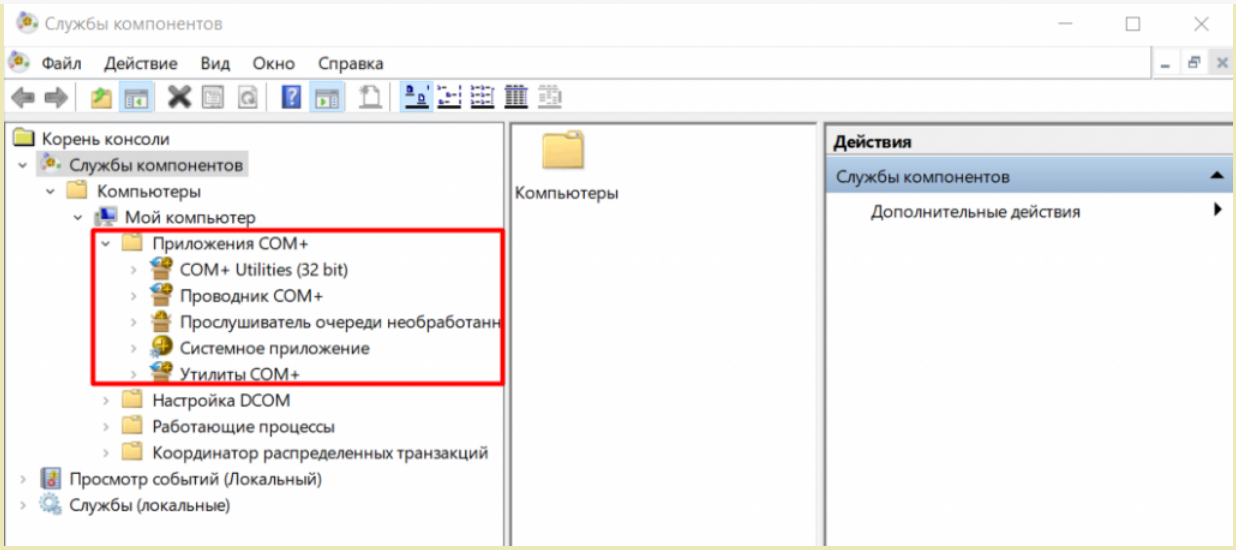
После регистрации коннектора следует проверить, выполняется ли подключение. Для этого следует разрешить доступ к коннектору («Разрешить»), и затем запустить его («Пуск»).





В ряде случаев случаев попытка запуска завершится неудачей, т.к. у сервера Mobile SMARTS не будет хватать пользовательских прав на подключение к внешней системе. Основные причины таких проблем и способы их решения описаны в статье «Диагностика и исправление проблем» на сайте «Клеверенс».

Если в «Службах компонентов» есть установленные вручную COM+ приложения, это может помешать запуску COM-коннектора. В таком случае потребуется удалить COM+ приложения, после чего заново зарегистрировать и запустить COM-коннектор.



коннекторы

Не нашли что искали?

Задать вопрос в техническую поддержку

Описание свойств коннекторов Mobile SMARTS

Последние изменения: 2024-03-26

Коннектор «1С: Предприятие версии 8»

Свойство
Заполняется автоматически
Принимаемые значения
Расшифровка
Максимальное количество одновременных вызовов
5
Произвольное число
Максимальное количество одновременных асинхронных вызовов 0 — нет ограничений
При тайм-ауте
Вызывать исключение
Вызывать исключение/ Переподключаться
Поведение при истечении тайм-аута Вызывать исключение — выполнение вызова останавливается и возвращается ошибка Переподключаться — отключить от системы и повторить вызов
Тайм-аут (в мс)
90000
Произвольное число
При истечении указанного времени вызов будет прерван и выполнится действие, указанное в пункте «При тайм-ауте» 0 — тайм-аут не используется, вызов будет выполняться до тех пор, пока вызываемая система не вернет результат или ошибку
Формат обмена данными
Auto
Auto/ Json/ Xml
В каком формате сериализованные данные передаются/возвращаются из «1С:Предприятие» в коннектор. Xml содержит сериализованные объекты Mobile SMARTS При работе через Web-коннектор используется Json, при работе через COM-соединение может использоваться как Xml, так и Json. Auto — формат определяется автоматически
Автоматический сброс зависших соединений
Нет
Да/ Нет
Только для COM-соединения. Если включено, происходит попытка завершения сеансов COM-соединения, которые долгое время неактивны (используется подключения к агенту сервера «1С:Предприятия»)

Версия 1С
8.3
8.1/ 8.2/ 8.3
Только для COM-соединения. Версия COM-объекта для подключения к базе «1С:Предприятие»
Вести лог статистики вызовов
Нет
Да/ Нет
Если включено, данные о вызовах будут записываться в базу sqlite по пути <папка базы Mobile SMARTS>\Logs\StatCon<Дата>.sqlite
Время жизни неиспользуемого соединения
30
Произвольное число
Только для COM-соединения. Время в секундах, в течение которого неиспользуемое COM-соединение будет оставаться в пуле соединений. Если указать <= 0, по умолчанию используется 60.
Домен
-
Произвольное имя
Используется только для Web-подключения при Windows-авторизации. Имя домена.
Имя администратора кластера
-
Произвольное имя
Только для COM-соединения. Необходимо для подключения к агенту сервера «1С:Предприятия», если включен автоматический сброс зависших соединений.

Количество постоянных соединений
Произвольное число
Только для COM-соединения. Максимальное количество постоянных соединений в пуле.
Количество резервных соединений
0
Произвольное число
Только для COM-соединения. Количество резервных соединений в пуле.
Максимальное число клиентов, ожидающее вызов 1С
48
Произвольное число
Только для COM-соединения. Максимальное число клиентов, ожидающее вызов 1С. Если значение превышено, вызов завершается с ошибкой.

Обработка драйвера
+
Произвольная строка
Только для COM-соединения. Имя обработки в конфигурации 1С или путь к внешней обработке. Функции вызываются из модуля обработки, и должны быть объявлены как экспортные.
Пароль
-
Произвольная строка
Пароль пользователя 1С
Пароль администратора кластера
-
Произвольная строка
Только для COM-соединения. Необходимо для подключения к агенту сервера «1С:Предприятия», если включен автоматический сброс зависших соединений.
Пользователь 1С
-
Произвольная строка
Имя пользователя 1С
Тип авторизации
Default
Default/ Win
Только для Web-подключения Default — базовая http-авторизация Win — используется Windows-авторизация
Тип подключения
Менеджер COM-соединений
Менеджер COM-соединений/ Automation сервер/ Менеджер на каждое COM-соединение/ Web-connector
Менеджер на каждое COM-соединение — для подключения используется COM-объект v8x.ComConnector Automation сервер — для подключения используется COM-объект v8x.Application, Web-connector — используется http-подключение к базе 1С, опубликованной на web-сервере

Коннектор «1С: Предприятие версии 7.7»

Свойство
Заполняется автоматически
Принимаемые значения
Расшифровка

Максимальное количество одновременных вызовов
0
Произвольное число
Максимальное количество одновременных асинхронных вызовов 0 — нет ограничений

При тайм-ауте
Вызывать исключение
Вызывать исключение/ Переподключаться
Поведение при истечении тайм-аута Вызывать исключение — выполнение вызова останавливается и возвращается ошибка Переподключаться — отключить от системы и повторить вызов
Тайм-аут
0
Произвольное число
При истечении указанного времени вызов будет прерван и выполнится действие, указанное в пункте «При тайм-ауте» 0 — тайм-аут не используется, вызов будет выполняться до тех пор, пока вызываемая система не вернет результат или ошибку
Объект соединения
V77.Application
V77.Application/ V1CEnterprise.Application/ V77S_Application/V77M_Application/ V77L_Application
COM-объект 1С, используемый для подключения к базе 1С: V1CEnterprise.Application — версия независимый ключ V77.Application — версия зависимый ключ V77S.Application — версия зависимый ключ, SQL версия V77L.Application — версия зависимый ключ, локальная версия V77M.Application — версия зависимый ключ, сетевая версия
Пароль
-
Произвольная строка
Пароль пользователя 1С
Пользователь
-
Произвольная строка
Имя пользователя 1С
Путь
-
Произвольная строка
Путь к папке с базой данных 1С

Axapta5Connector

Свойство
Заполняется автоматически
Принимаемые значения
Расшифровка
Company
-
Произвольная строка
Наименование компании (параметр подключения через Axapta Business Connector)
ConfigurationName
-
Произвольная строка
Наименование конфигурации (параметр подключения через Axapta Business Connector)
Domain
-
Произвольная строка
Домен, в котором зарегистрирован пользователь (параметр подключения через Axapta Business Connector)
Language
-
Произвольная строка
Язык (параметр подключения через Axapta Business Connector)
ObjectServer
-
Произвольная строка
Сервер Axapta (параметр подключения через Axapta Business Connector)
UserName
-
Произвольная строка
Имя пользователя (параметр подключения через Axapta Business Connector)
Пароль
-
Произвольная строка
Строка (параметр подключения через Axapta Business Connector)

При тайм-ауте
Вызывать исключение
Вызывать исключение/ Переподключаться
Поведение при истечении тайм-аута Вызывать исключение — выполнение вызова останавливается и возвращается ошибка Переподключаться — отключить от системы и повторить вызов
Тайм-аут
0
Произвольное число (в мс)
При истечении указанного времени вызов будет прерван и выполнится действие, указанное в пункте «При тайм-ауте» 0 — тайм-аут не используется.

Microsoft SQL Server

Свойство
Заполняется автоматически
Принимаемые значения
Расшифровка
DataSource
-
Произвольная строка
Имя или сетевой адрес SQL-сервера
InitialCatalog
-
Произвольная строка
Имя базы данных на SQL-сервере
IntegratedSecurity
True
True/ False
True — для подключения используется текущая учетная запись Windows False — используется заданные имя пользователя базы SQL и пароль пользователя
UserID
-
Произвольная строка
Идентификатор пользователя базы SQL

Пароль
-
Произвольная строка
Пароль пользователя базы SQL

При тайм-ауте
-
Вызывать исключение/ Переподключаться
Поведение при истечении тайм-аута. Вызывать исключение — выполнение вызова останавливается и возвращается ошибка Переподключаться — отключить от системы и повторить вызов
Тайм-аут
-
Произвольное число (в мс)
При истечении указанного времени вызов будет прерван и выполнится действие, указанное в пункте «При тайм-ауте»

Не нашли что искали?

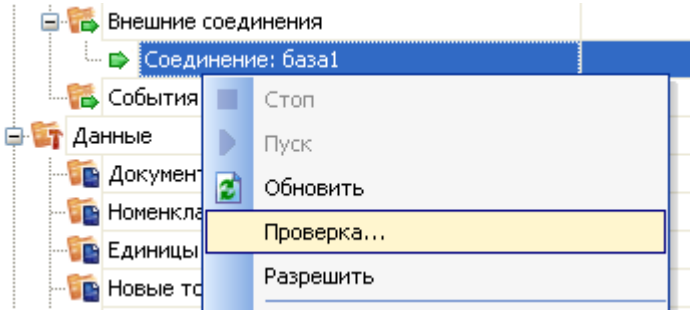
Задать вопрос в техническую поддержку

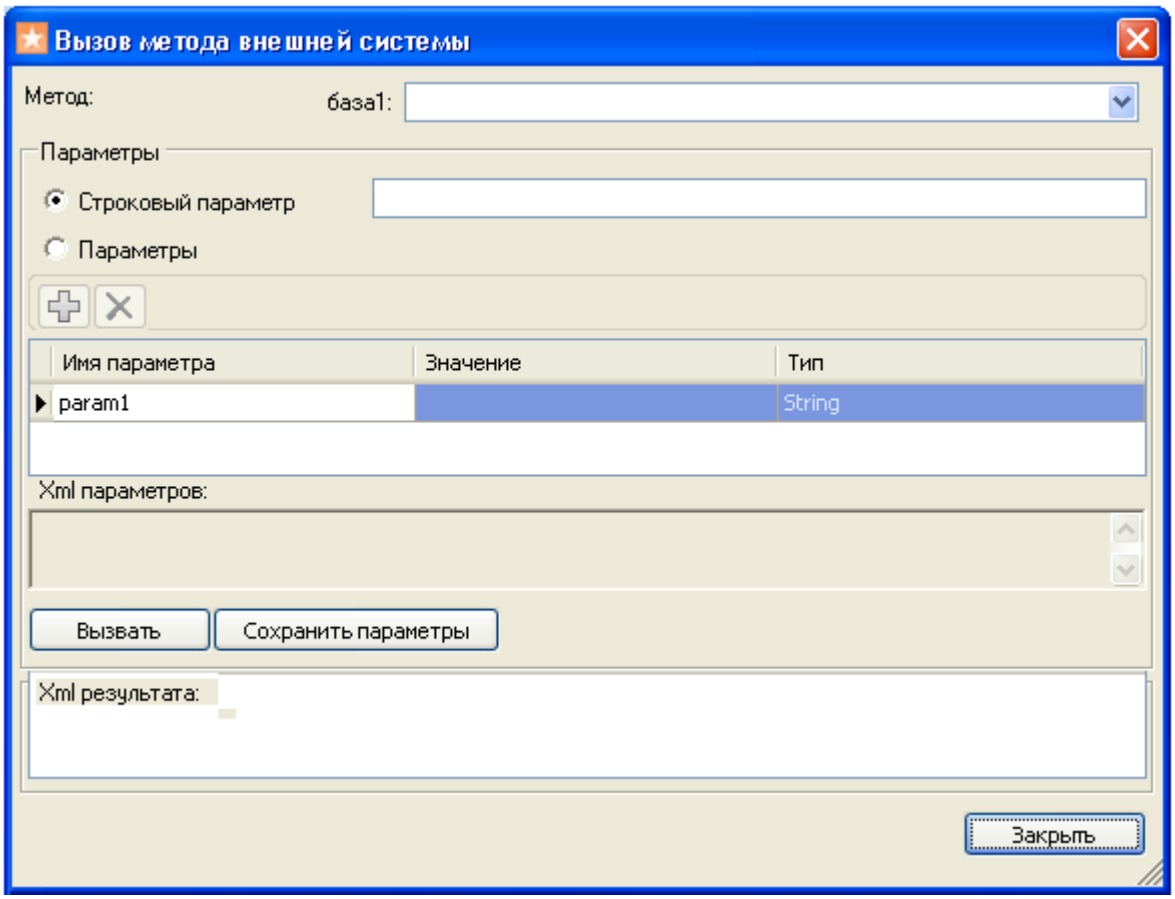
Отладка вызовов к внешней системе через коннекторы

Последние изменения: 2024-03-26

Когда оператор терминала сбора данных сканирует штрихкод для запроса остатков товара, со стороны внешней системы (учетной системы, товарной базы) кто-то должен обработать этот запрос и вернуть результат. В случае 1С это будет метод глобального контекста или контекста внешнего соединения, для Axapta это будет публичный статический метод в каком-то классе, а в случае SQL базы данных это может быть хранимая процедура (хотя можно выполнить и простой запрос).

Для отладки работы кода внешней системы, который должен будет вызываться с ТСД, в панели управления Mobile SMARTS предусмотрено специальное окно, вызываемое из контекстного меню коннектора:





В этом окне можно задать имя вызываемого метода внешней системы, указать передаваемые параметры, вызвать внешнюю систему при помощи коннектора и посмотреть на результат.

Как видно, коннектору передаются именованные параметры. Метод внешней системы, который будет принимать эти параметры, может получить их в двух вариантах, в зависимости от реализации коннектора. Первый вариант – будет учитываться только порядок передачи параметров, а имена не имеют значения и отбрасываются. Второй вариант – метод принимает единственный строковой аргумент, в котором передается

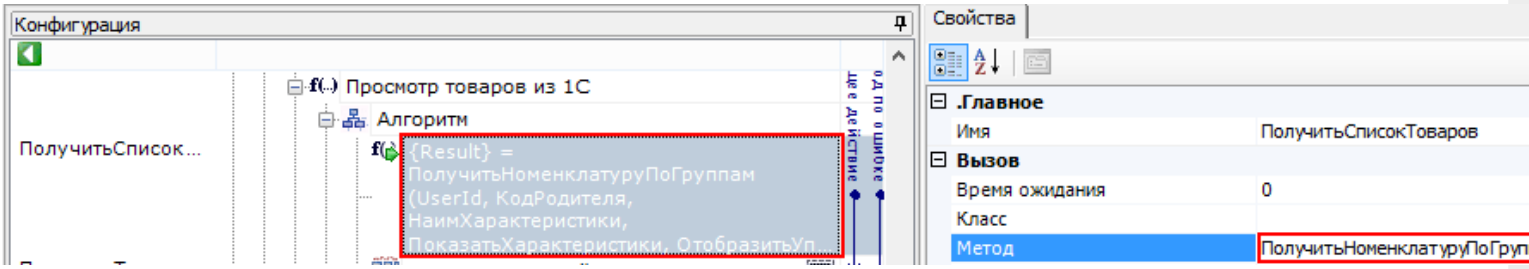
XML с сериализованным объектом Cleverence.Warehosue.InvokeArgs; это выглядит примерно как «<?xml version="1.0" encoding="utf-8" ?><InvokeArgs xmlns:clr="http://schemas...».

Отладка процедур и функций базы 1С драйвера

Разберем, как подключить отладчик 1С к конфигурации, с которой взаимодействует Mobile SMARTS, чтобы отладить выполнение какой-нибудь функции, которая вызывается с терминала.

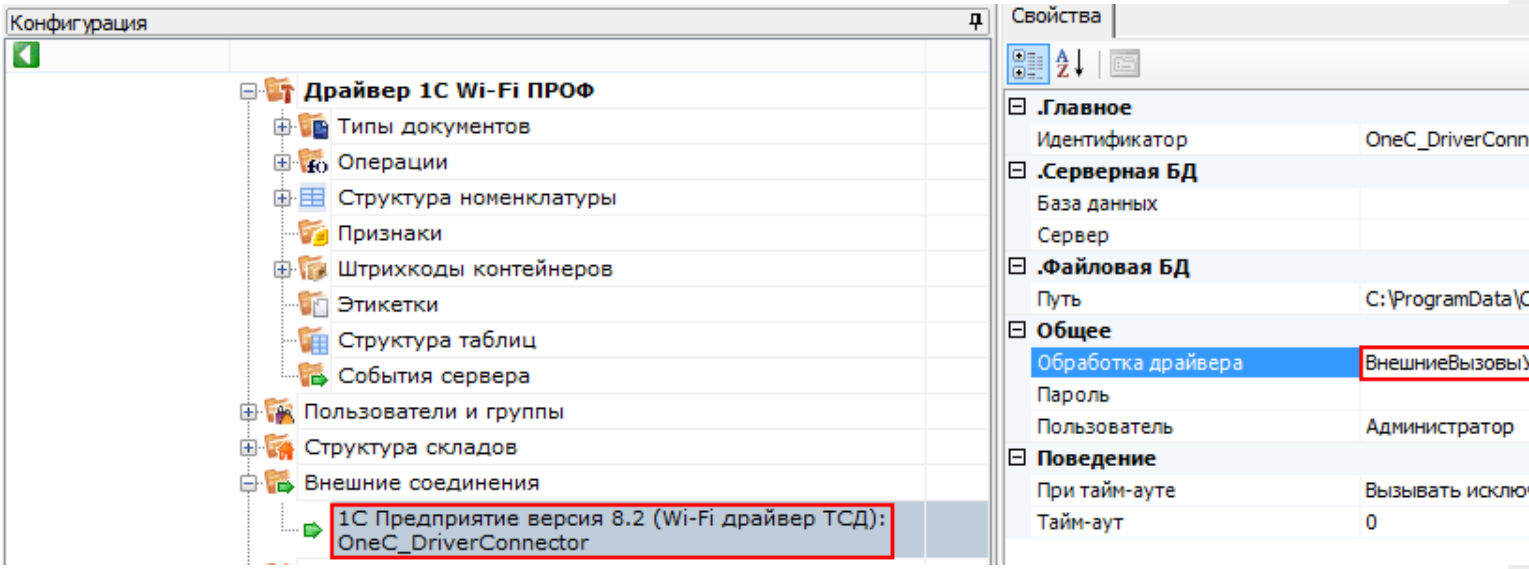
Рассмотрим подключение на основе операции «Просмотр товаров из 1С» из стандартной поставки Wi-Fi ПРОФ драйвера. Разберем, что вызывается и отладим процедуру, которая вызывается.

Для начала найдем соответствующий алгоритм в панели управления и посмотрим, как называется метод, вызываемый из 1С.



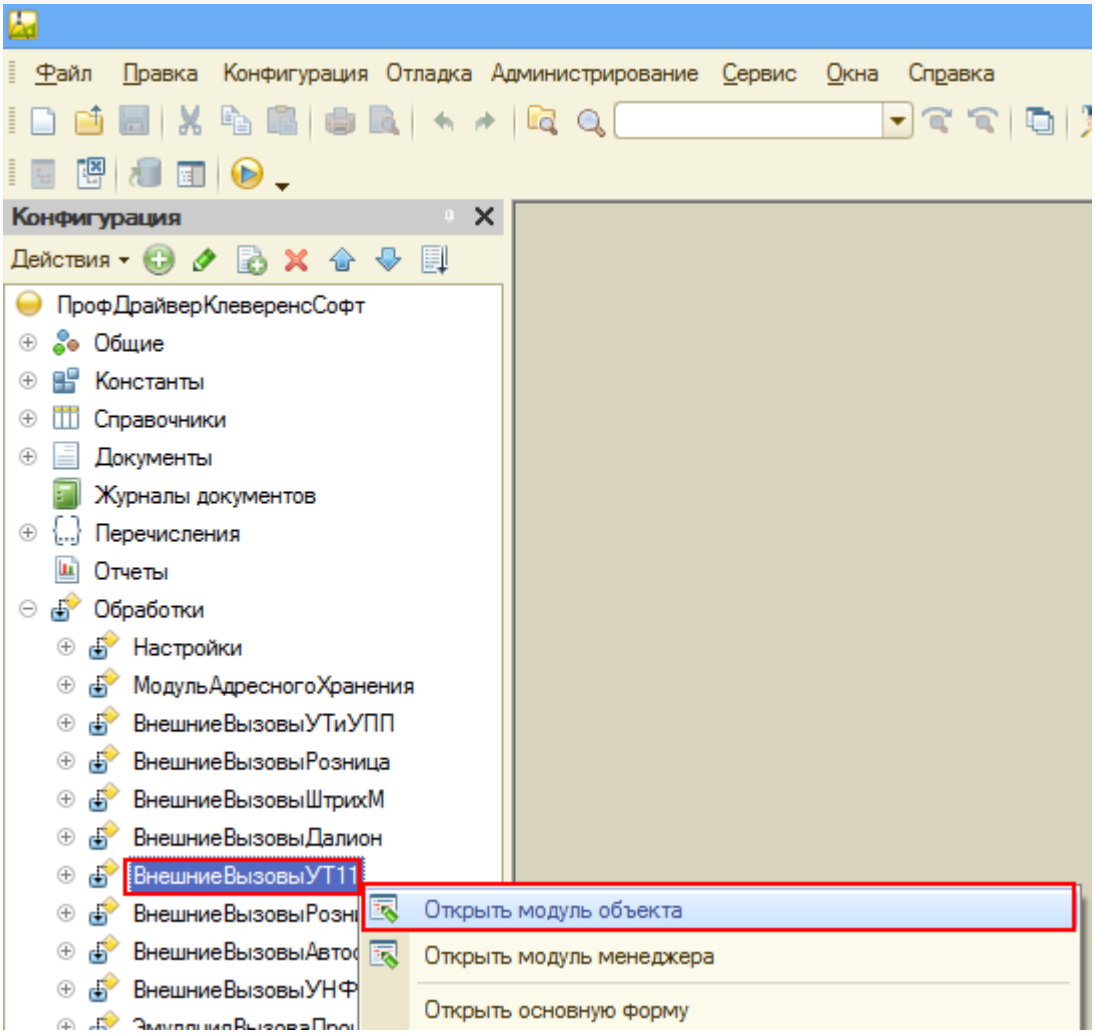
Для данной операции метод, который вызывается из 1С называется «ПолучитьНоменклатуруПоГруппам».

Затем посмотрим какая обработка подключена. Для этого нужно посмотреть Внешние соединения.

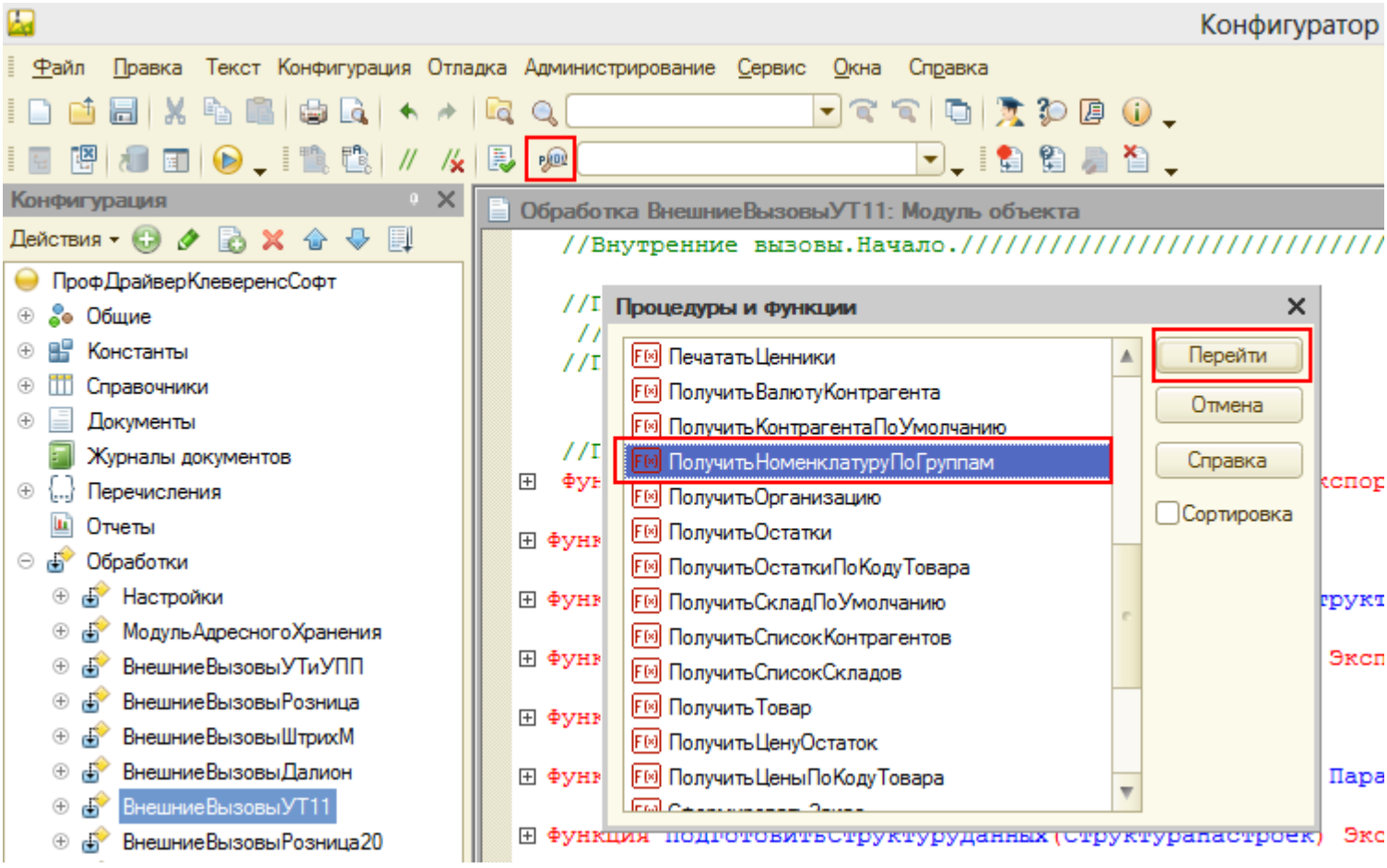


В данном примере подключена обработка «ВнешниеВызовыУТ11».

Теперь найдем данную процедуру в модуле обработки.



Открываем модуль объекта, данной обработки, ищем нашу функцию и нажимаем кнопку «Перейти».

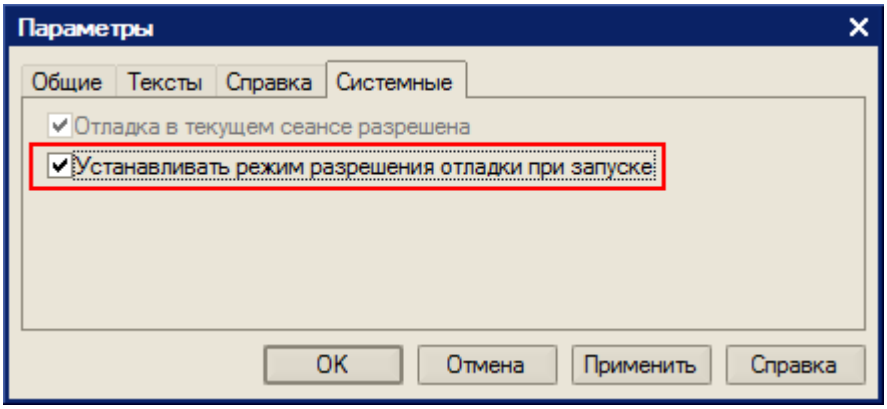


Обработка ВнешниеВызовыУТ11: Модуль объекта

```
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////

//Вызовы внешней системы.Начало.**********************************************
Функция ДобавитьШтрихкод(Штрихкод, КодТовара, КодЕдиницыИзмерения, Характеристика=Неопределено) Эк

Функция ПечататьЗаказ(НомерДокумента) Экспорт...

Функция ПечататьЦенники(UserId, СтрокиДокументаXml) Экспорт...

Функция ПолучитьВалютуКонтрагента(UserId, КонтрагентИмя) Экспорт...

Функция ПолучитьКонтрагентаПоУмолчанию(UserId) Экспорт...

Функция ПолучитьНоменклатуруПоГруппам(UserId, КодГруппы, НаимХарактеристики, ПоказатьХарактеристик
    Возврат УниверсальныеМеханизмы.ПолучитьНоменклатуруПоГруппам(UserId, КодГруппы, НаимХарактерис
КонецФункции

Функция ПолучитьОрганизацию(UserId) Экспорт...

Функция ПолучитьОстатки(Номенклатура, СерияНоменклатуры, ХарактеристикаНоменклатуры, Склад = Неопр

Функция ПолучитьОстаткиПоКодуТовара(UserId, Склад, Штрихкод, КодТовара, КодЕдиницыИзмерения) Экспо
```

Клиент-файловый вариант работы

В клиент-файловом варианте необходимо проверить подключена ли отладка. Для этого запускаем режим «Предприятие», заходим в Сервис -> Параметры -> вкладка Системные. Смотрим, проставлен ли флаг «Устанавливать режим разрешения отладки при запуске». Если не проставлено необходимо проставить и сохранить.



В случае, если флаг «Устанавливать режим разрешения отладки при запуске» не сохраняется, необходимо скопировать файл comcntrcfg.xml (предварительно скачать и извлечь из архива comcntrcfg.zip) в папки «c:\Program Files\1cv82\8.x.xx.xxx\bin\conf\», «c:\Program Files\1cv82\conf\».

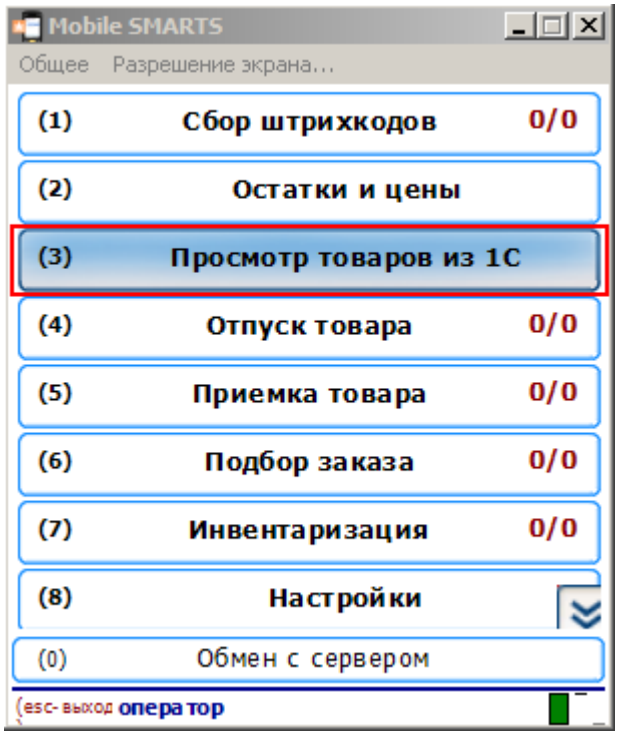
В файле прописано, что отладка разрешена и по умолчанию включается на локальной машине.

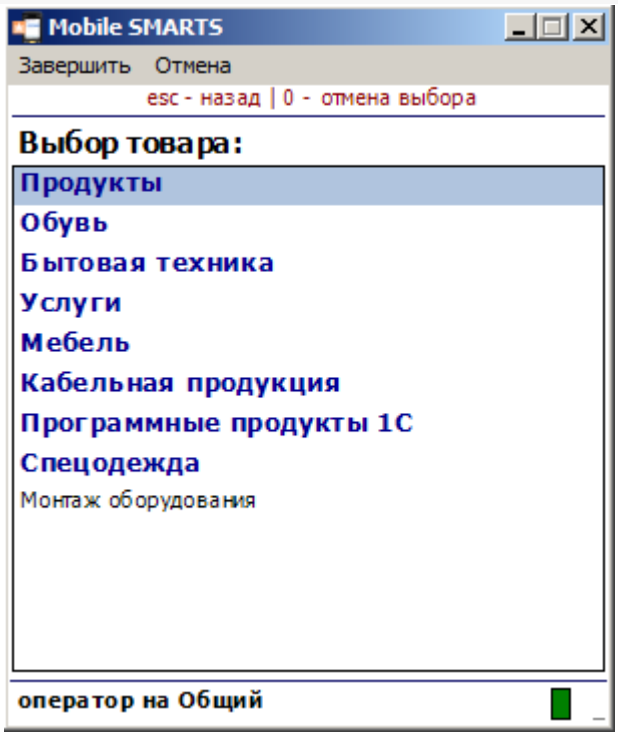
```
Файл  Правка  Вид  Кодировка  Справка
<config xmlns="http://v8.2c.ru/v8/comcntrcfg">
    <debugconfig debug="true"
    debuggerURL="tcp://localhost:1560"/>
</config>
```

Дальше поставим «точку останова».

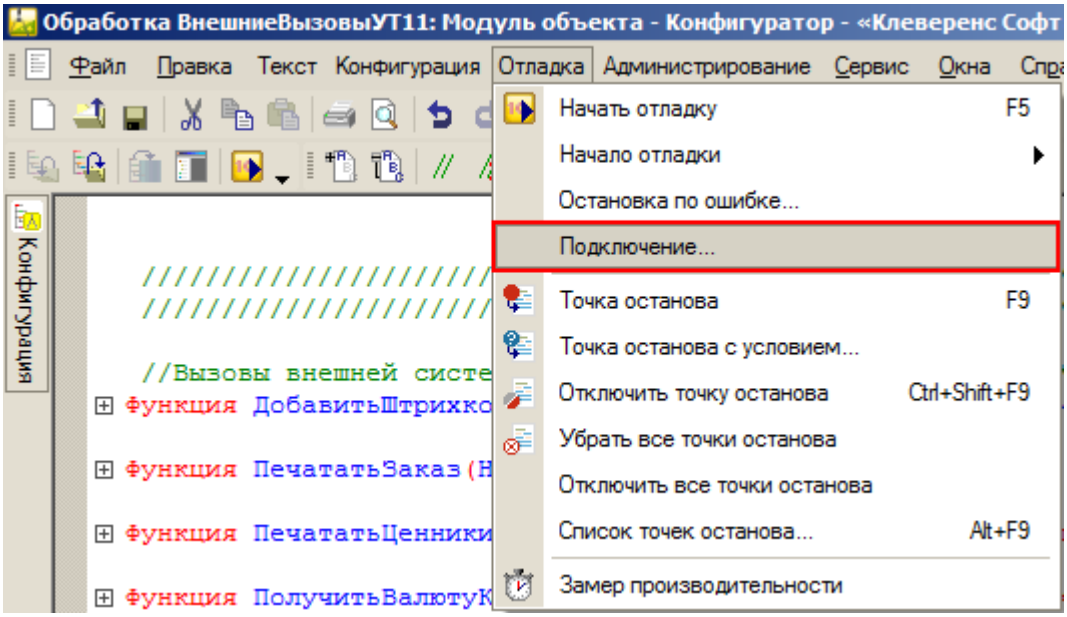
```
Функция ПолучитьНоменклатуруПоГруппам(UserId, КодГруппы, НаимХарактеристики, ПоказатьХарактеристики,Ото
    Возврат УниверсальныеМеханизмы.ПолучитьНоменклатуруПоГруппам(UserId, КодГруппы, НаимХарактеристики,
КонецФункции
```

Запустим вызов процедуры, выбрав операцию.

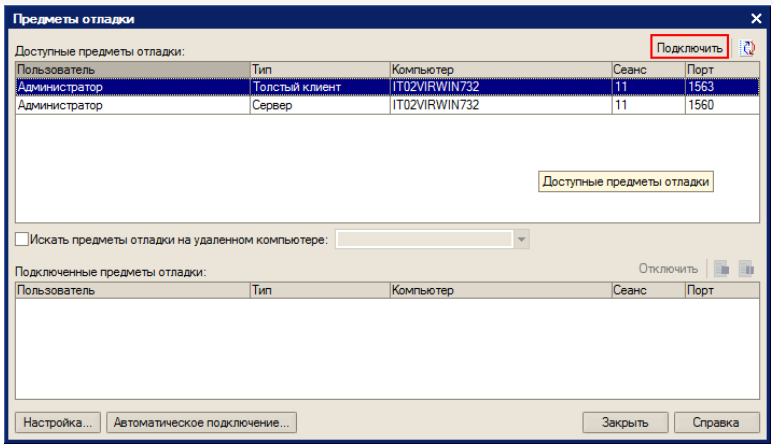


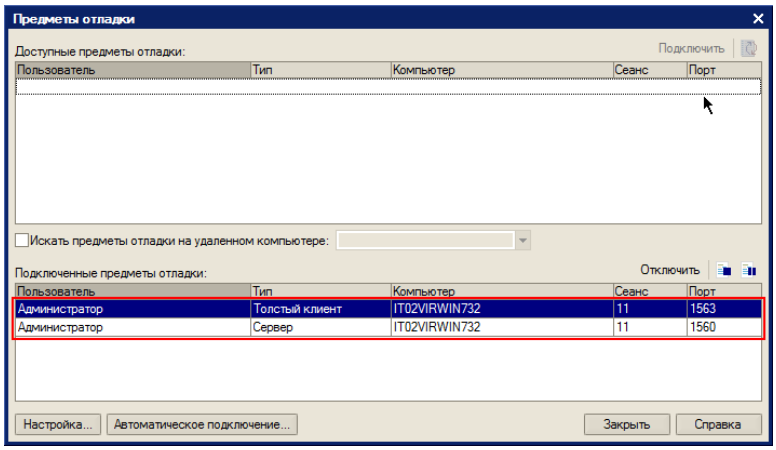


Создалось подключение сервера Mobile SMARTS к базе драйвера и теперь необходимо подключиться к той сессии, которая создалась. Выбираем Отладка -> Подключение.

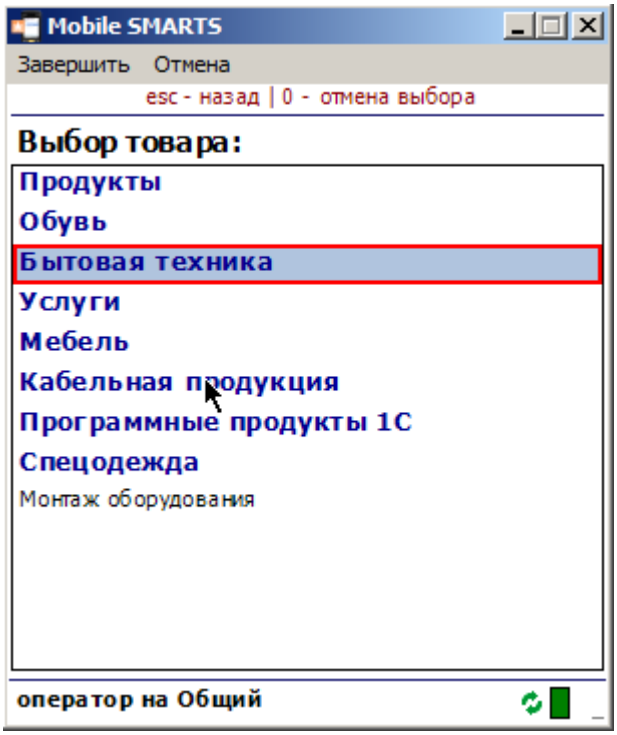


Производим подключение.





Вызовем нашу процедуру еще раз.



Система остановилась, на нашей «точке останова».

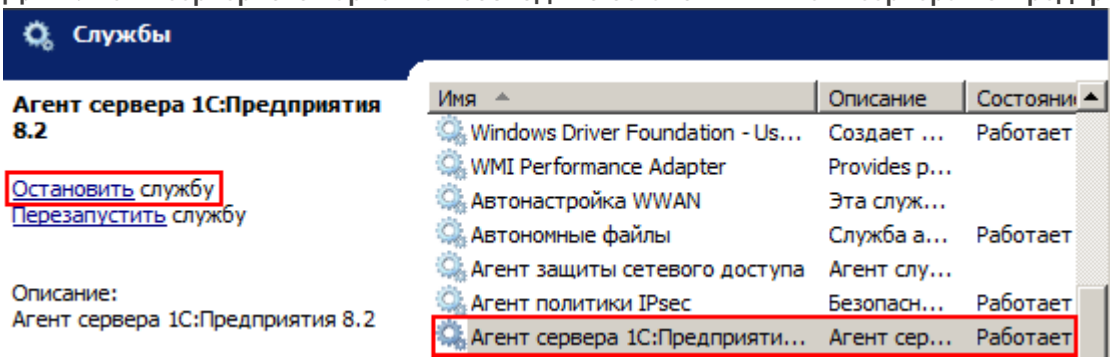
```
Функция ПолучитьНоменклатуруПоГруппам(UserId, КодГруппы, НаимХарактеристики, ПоказатьХарактеристики,Ото
    Возврат УниверсальныеМеханизмы.ПолучитьНоменклатуруПоГруппам(UserId, КодГруппы, НаимХарактеристики,
КонецФункции
```

Далее стандартными методами 1С (зайти в процедуру, шагнуть) можно отладить соответствующую процедуру 1С.

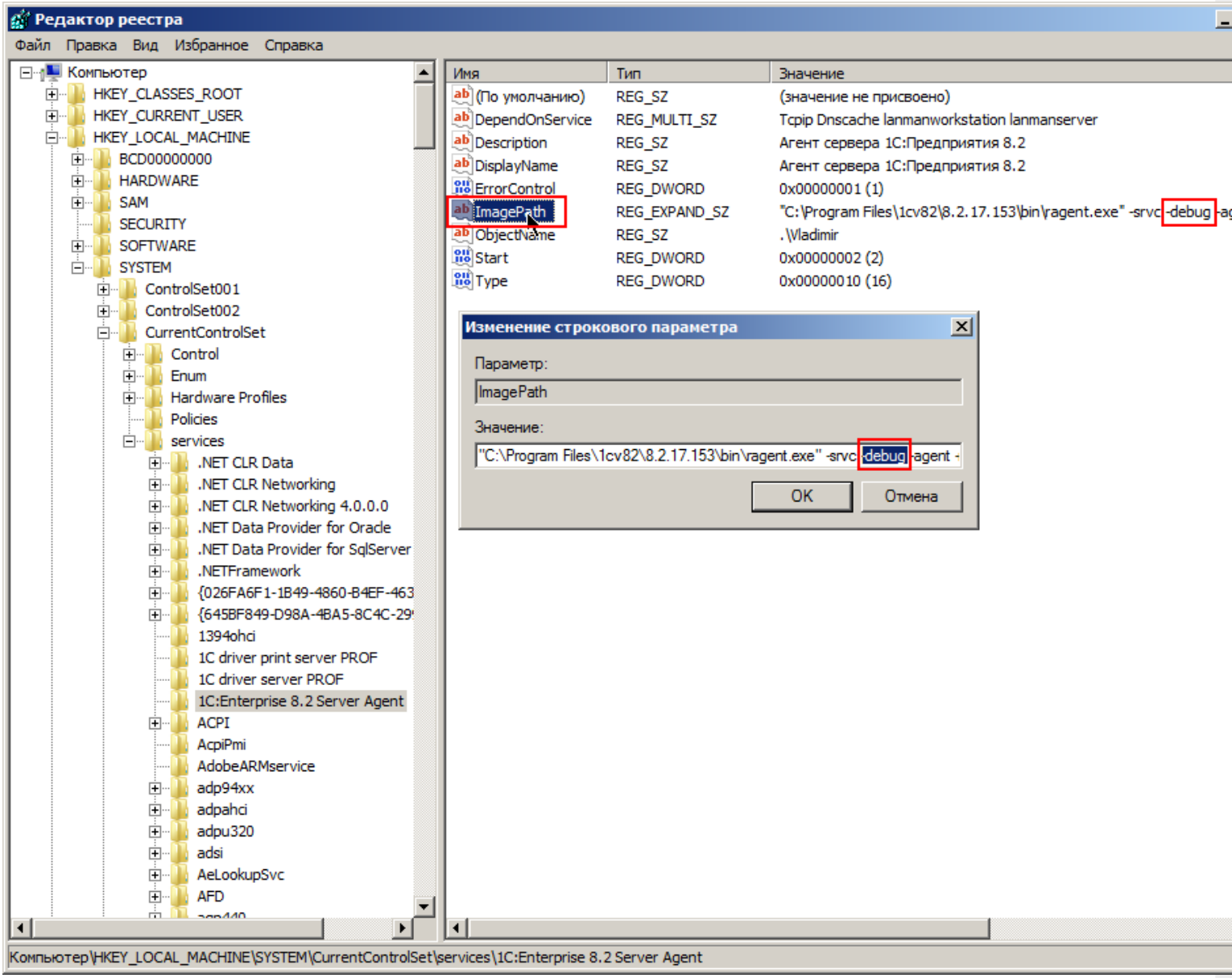
Клиент-серверный вариант работы

> Для клиент-серверного варианта работы отладку нужно производить на том же компьютере, где запущен «Агент сервера 1С:Предприятия».

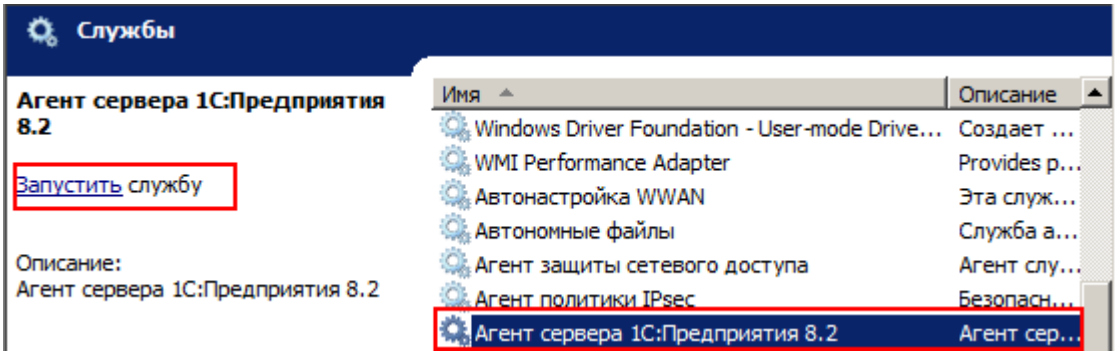
Для клиент-серверного варианта необходимо остановить «Агент сервера 1С:Предприятия».



В реестре (Компьютер\HKEY_LOCAL_MACHINE\SISTEM\CurrentControlSet\services\1C:Enterprise 8.2 Server Agent) найти параметр ImagePath и написать ключик -debug.



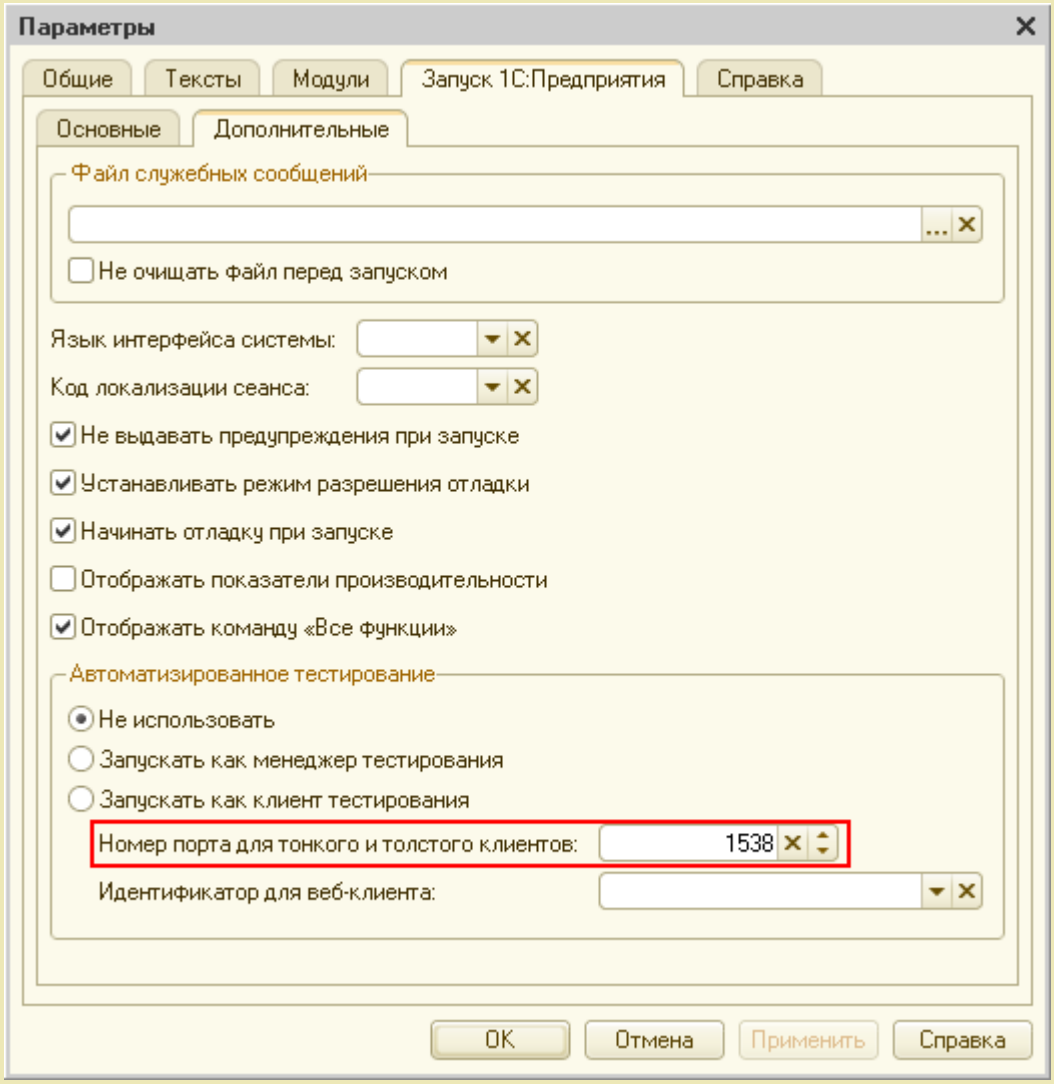
Запускаем службу.



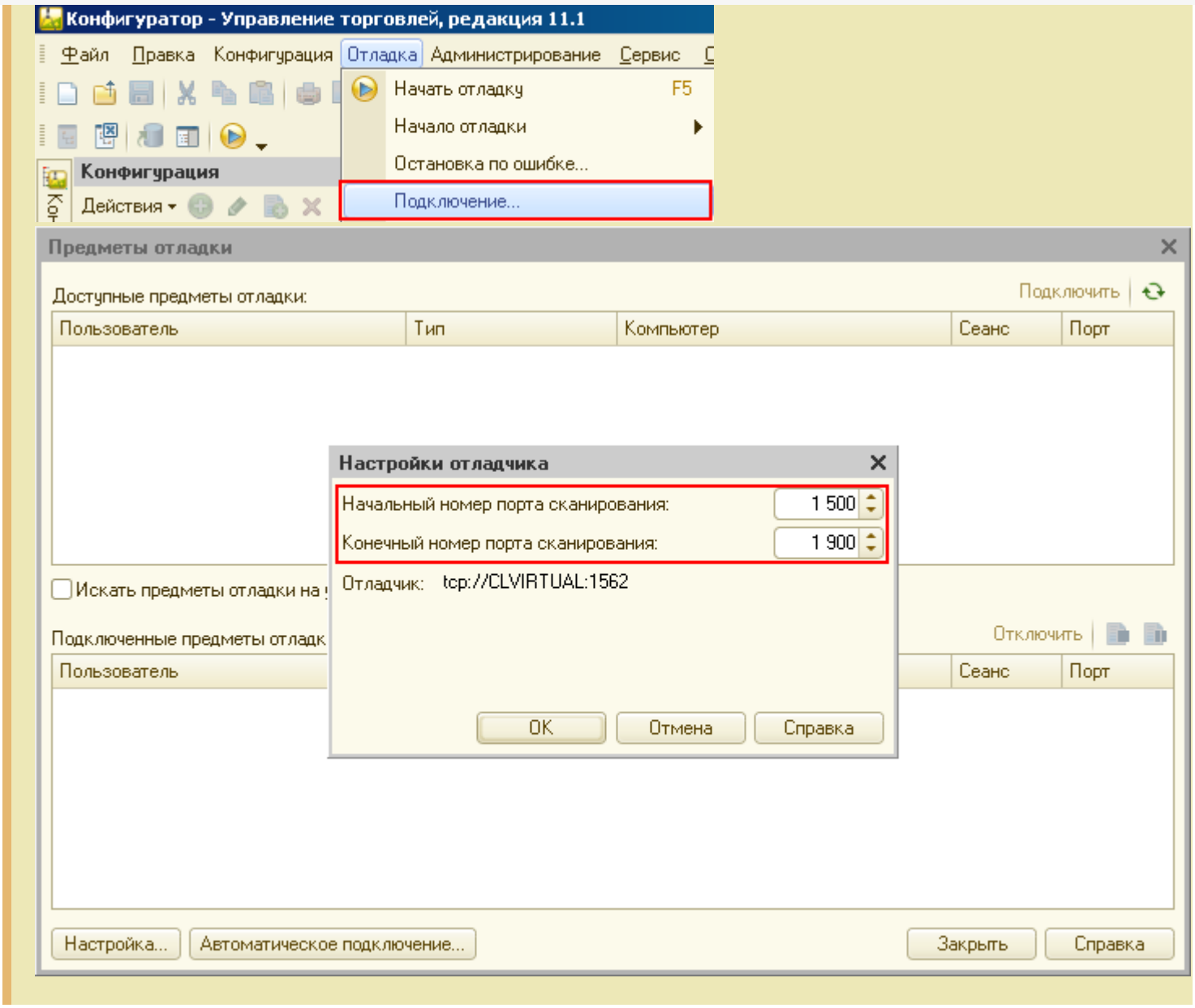
Запускаем конфигуратор базы драйвера.

И аналогично, как в клиент-файловом варианте, ставим «точку останова», вызываем метод 1С из Mobile SMARTS, отлаживаем код в 1С.

В 1С 8.3 появился параметр «Номер порта для тонкого и толстого клиента».



Для отладки это значение должно находится в пределах портов установленных в параметрах:



отладка, драйвер ПРОФ

Не нашли что искали?

Задать вопрос в техническую поддержку

Событие сервера Mobile SMARTS «ДокументДобавляется»

Последние изменения: 2024-03-26

Событие, возникающее в процессе добавления документа на сервере. Вызывается при выгрузке документа в базу Mobile SMARTS из внешней системы.

Параметры

Имя параметра
Тип данных
Описание
ИдДокумента
string (строка)
Код документа, который выгружается на сервер.
ИдПользователя
string (строка)
Передается null
XmlСессии
string (строка)
Объект сессии сервера Mobile SMARTS, сериализованный в xml. Параметр передается, только если включена настройка «Добавлять объект сессии в вызов событий».

Событие может быть обработано модулем «Вызов серверных операций», в этом случае в серверную операцию передается сам документ (переменная сессии Document). Если выполнение серверной операции прерывается (переход [прервать операцию]), добавление документа на сервер отменяется. Если отмена добавления не требуется, операция должна завершиться по переходу [завершить операцию].

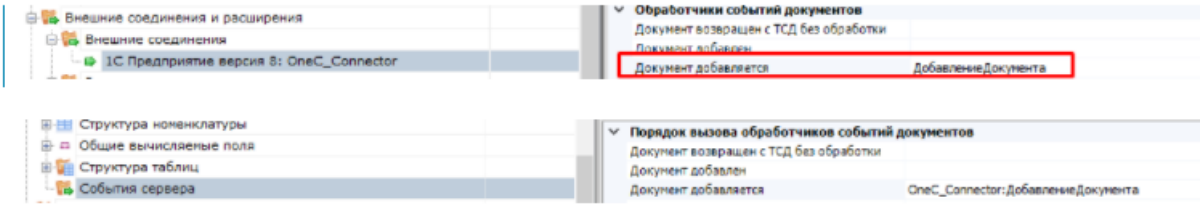
В случае обычных коннекторов, если обработчик события вернул null, добавление документа на сервер отменяется.

Описание в панели управления:

```
C#

<ид. коннектора>:ДобавлениеДокумента
```

Ид. коннектора — задается в панели управления.

Например: OneC_DriverConnector:ДобавлениеДокумента



Не нашли что искали?

Задать вопрос в техническую поддержку

Событие сервера Mobile SMARTS «ДокументДобавлен»

Последние изменения: 2024-03-26

Событие, возникающее после добавления документа на сервере. Вызывается при выгрузке документа в базу Mobile SMARTS из внешней системы.

Параметры

Имя параметра
Тип данных
Описание
ИдДокумента
string (строка)
Код документа, который выгружен на сервер.
ИдПользователя
string (строка)
Передается null.
XmlСессии
string (строка)
Объект сессии сервера Mobile SMARTS, сериализованный в xml. Параметр передается, только если включена настройка «Добавлять объект сессии в вызов событий».

Событие может быть обработано модулем «Вызов серверных операций», в этом случае в серверную операцию передается сам документ (переменная сессии Document).

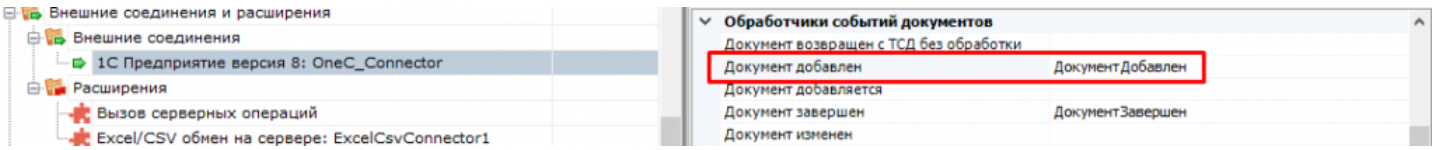
Обработчик может внести изменения в документ. После вызова обработчика документ сохраняется на сервере.

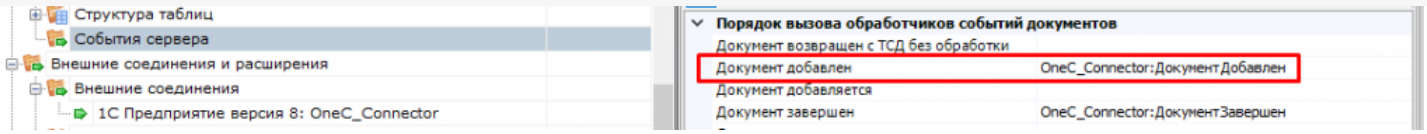
Описание в панели управления**:**

```
C#

<ид. коннектора>:ДокументДобавлен
```

Описание в панели управления:
Ид. коннектора - задается в панели управления.

Например: OneC_Connector:ДокументДобавлен





события сервера, коннекторы

Не нашли что искали?

Задать вопрос в техническую поддержку

Событие сервера Mobile SMARTS «ДокументНазначаетсяПользователю»

Последние изменения: 2024-03-26

Событие о том, что документ готов передаваться на мобильное устройство. Вызывается при запросе с ТСД получения документа для работы.

Параметры

Имя параметра
Тип данных
Описание
ИдДокумента
string (строка)
Код документа, который получает пользователь для работы на ТСД.
ИдПользователя
string (строка)
Ид. пользователя ТСД, который выполнил запрос.
XmlСессии
string (строка)
Объект сессии сервера Mobile SMARTS, сериализованный в xml. Параметр передается, только если включена настройка«Добавлять объект сессии в вызов событий».

Событие может быть обработано модулем «Вызов серверных операций», в этом случае в серверную операцию передается сам документ (переменная сессии Document). Если выполнение серверной операции прерывается (переход [прервать операцию]), назначение документа пользователю ТСД отменяется. Если отмена назначения не требуется, операция должна завершиться по переходу [завершить операцию].

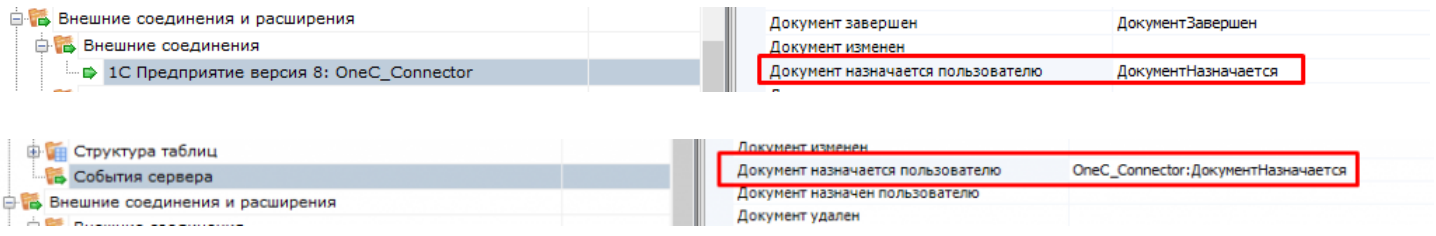
В случае обычных коннекторов, если обработчик события вернул null, назначение документа отменяется.

Описание в панели управления:

```
C#

<ид. коннектора>:ДокументНазначается
```

Ид. коннектора - задается в панели управления.

Например: OneC_Connector:ДокументНазначается



Не нашли что искали?

Задать вопрос в техническую поддержку

Событие сервера Mobile SMARTS «ДокументНазначенПользователю»

Последние изменения: 2024-03-26

Событие о том, что документ захвачен на обработку. Вызывается в момент, когда документ был передан на ТСД для работы с ним.

Параметры

Имя параметра
Тип данных
Описание
ИдДокумента
string (строка)
Код документа, который отправлен для работы на ТСД.
ИдПользователя
string (строка)
Ид. пользователя ТСД, который выполнил запрос.
XmlСессии
string (строка)
Объект сессии сервера Mobile SMARTS, сериализованный в xml. Параметр передается, только если включена настройка «Добавлять объект сессии в вызов событий».

Событие может быть обработано модулем «Вызов серверных операций», в этом случае в серверную операцию передается сам документ (переменная сессии Document).

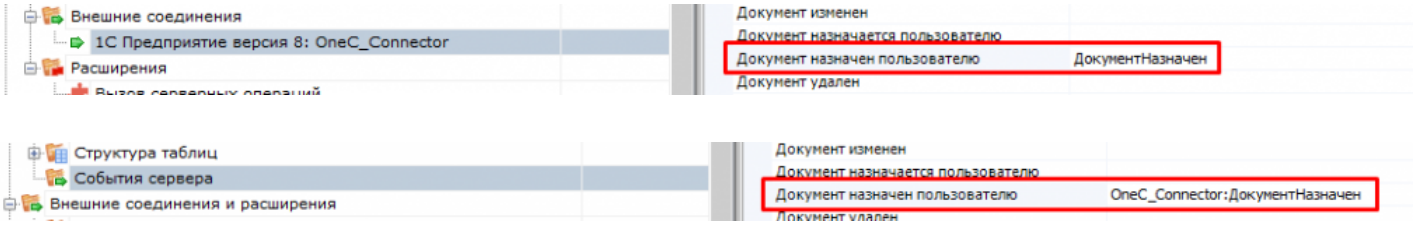
Обработчик может внести изменения в документ. После вызова обработчика документ сохраняется на сервере.

Описание в панели управления:

```
C#

<ид. коннектора>:ДокументНазначен
```

Ид. коннектора - задается в панели управления.

Например: OneC_Connector:ДокументНазначен



Не нашли что искали?

Задать вопрос в техническую поддержку

Событие сервера Mobile SMARTS «ДокументИзменен»

Последние изменения: 2024-03-26

Событие об изменении документа. Вызывается при сохранении документа на сервер в процессе работы на ТСД при использовании в конфигурации Mobile SMARTS действия «Сохранение документа на сервер».

Параметры

Имя параметра	Тип данных	Описание
ИдДокумента	string (строка)	Код документа, который был изменен.
ИдПользователя	string (строка)	Идентификатор пользователя Mobile SMARTS, при работе которого на ТСД, произошел вызов обработчика.
XmlСессии	string (строка)	Объект сессии сервера Mobile SMARTS, сериализованный в xml. Параметр передается, только если включена настройка «Добавлять объект сессии в вызов событий».

Событие может быть обработано модулем «Вызов серверных операций», в этом случае в серверную операцию передается сам документ (переменная сессии Document).

Описание в панели управления:

```
C#

<ид. коннектора>:ДокументИзменен
```

Ид. коннектора - задается в панели управления.

Например: OneC_DriverConnector:ДокументИзменен

события сервера, коннекторы

Не нашли что искали?

Задать вопрос в техническую поддержку

Событие сервера Mobile SMARTS «ДокументЗавершен»

Последние изменения: 2024-03-26

Событие о завершении обработки документа. Вызывается при получении сервером завершенного документа с ТСД. Может использоваться для автоматической загрузки завершенных документов в учетную систему. Учетная система может выполнить загрузку документа по ID с помощью функции GetDocument компоненты Cleverence.Warehouse.StorageConnector, после загрузки документ может быть удален из базы с помощью вызова RemoveDocument. Подробнее описано здесь.

Параметры

Имя параметра
Тип данных
Описание
ИдДокумента
string (строка)
Код документа, который был завершен.
XmlСессии
string (строка)
Объект сессии сервера Mobile SMARTS, сериализованный в xml. Параметр передается, только если включена настройка «Добавлять объект сессии в вызов событий».

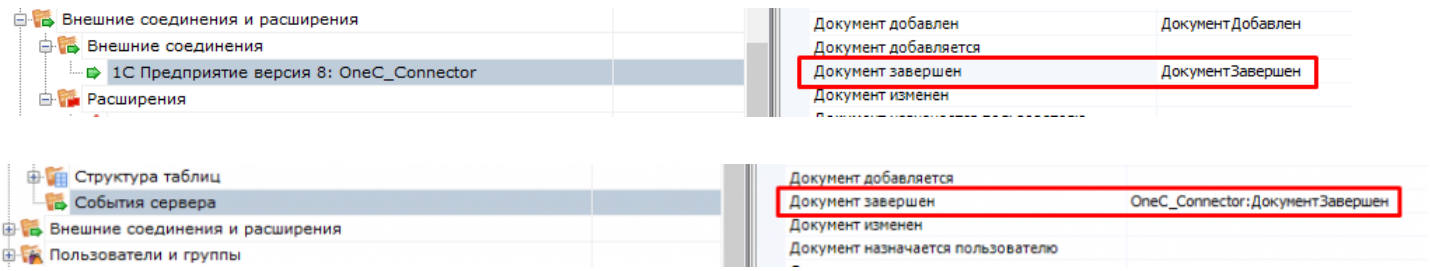
Событие может быть обработано модулем «Вызов серверных операций», в этом случае в серверную операцию передается сам документ (переменная сессии Document).

Описание в панели управления:

```
C#

<ид. коннектора>:ДокументЗавершен
```

Ид. коннектора - задается в панели управления.

Например: OneC_Connector:ДокументЗавершен



события сервера, коннекторы

Не нашли что искали?

Задать вопрос в техническую поддержку

Событие сервера Mobile SMARTS «ДокументВозвращенСТсдБезОбработки»

Последние изменения: 2024-03-26

Вызывается, когда документ с терминала был возвращен пользователем вызовом release (вернуть документ без изменений).

Параметры

Имя параметра
Тип данных
Описание
ИдДокумента
string (строка)
Код документа, который получает пользователь для работы на ТСД.
ИдПользователя
string (строка)
Ид. пользователя ТСД, который выполнил запрос.
XmlСессии
string (строка)
Объект сессии сервера Mobile SMARTS, сериализованный в xml. Параметр передается, только если включена настройка «Добавлять объект сессии в вызов событий».

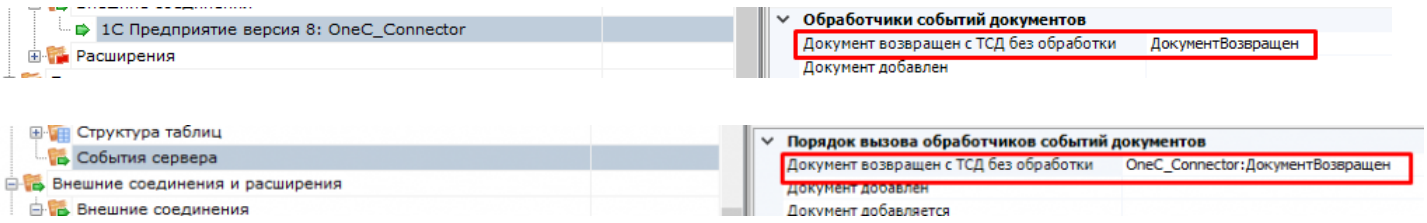
Событие может быть обработано модулем «Вызов серверных операций», в этом случае в серверную операцию передается сам документ (переменная сессии Document).

Описание в панели управления:

```
C#

<ид. коннектора>:ДокументВозвращен
```

Ид. коннектора - задается в панели управления.

Например: OneC_Connector:ДокументВозвращен



Не нашли что искали?

Задать вопрос в техническую поддержку

Событие сервера Mobile SMARTS «ДокументУдален»

Последние изменения: 2024-03-26

Событие об удалении документа с сервера. Удаление выполняется внешней системой.

Параметры

Имя параметра
Тип данных
Описание
ИдДокумента
string (строка)
Код документа, который получает пользователь для работы на ТСД.
ИдПользователя
string (строка)
Идентификатор пользователя Mobile SMARTS, при работе которого на ТСД, произошел вызов обработчика.
XmlСессии
string (строка)
Объект сессии сервера Mobile SMARTS, сериализованный в xml. Параметр передается, только если включена настройка «Добавлять объект сессии в вызов событий».

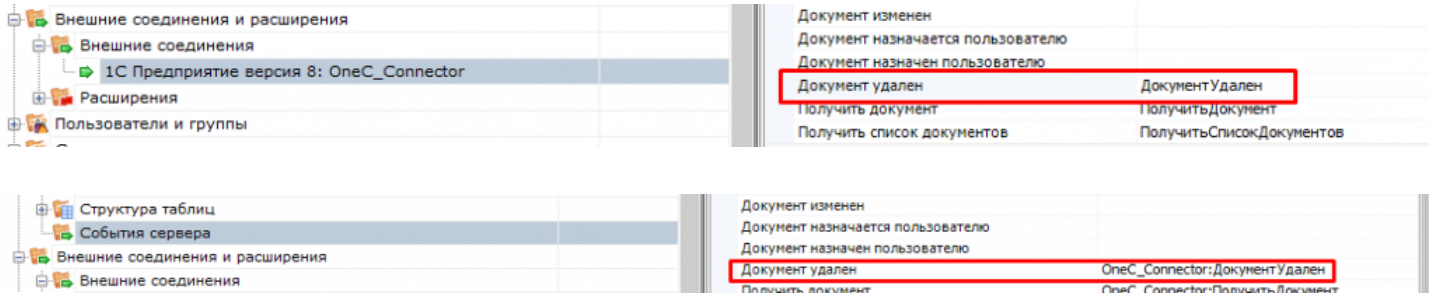
Событие может быть обработано модулем «Вызов серверных операций», в этом случае в серверную операцию передается сам документ (переменная сессии Document).

Описание в панели управления:

```
C#

<ид. коннектора>:ДокументУдален
```

Ид. коннектора - задается в панели управления.

Например: OneC_DriverConnector:ДокументУдален



события сервера, коннекторы

Не нашли что искали?

Задать вопрос в техническую поддержку

Событие сервера Mobile SMARTS «ПолучитьДокумент»

Последние изменения: 2024-03-26

Событие о запросе получения документа по идентификатору или штрихкоду с сервера. Событие может использоваться совместно с событием «Получить список документов». Обработчик события «Получить список документов» возвращает список описаний документов (Cleverence.Warehouse.DocumentDescriptionCollection). Список отображается в окне выбора документа на ТСД, пользователь ТСД выбирает позицию из списка, идентификатор выбранного документа передается сервером в обработчик события «Получить документ». Обработчик «Получить документ» возвращает документ Mobile SMARTS (объект Cleverence.Warehouse.Document, сериализованный в xml).Также получение документа может выполняться по штрихкоду, отсканированному в окне выбора документов. В этом случае в параметрах типа документа должна быть включена настройка «Выбирать по штрихкоду с сервера — Да».

Параметры

Имя параметра
Тип данных
Описание
ИдПользователя
string (строка)
Идентификатор пользователя Mobile SMARTS, при работе которого на ТСД, произошел вызов обработчика.
КодДокумента
string (строка)
Идентификатор или штрихкод запрашиваемого документа.
ТипДокумента
string (строка)
Тип документов, которые запрашиваются с терминала.
Режим
int (целое число)
=0 — получить документ по коду; =1 — получить документ по штрихкоду.
XmlСессии
string (строка)
Объект сессии сервера Mobile SMARTS, сериализованный в xml. Параметр передается, только если включена настройка «Добавлять объект сессии в вызов событий».

Возвращаемое значение

Функция должна вернуть объект типа Cleverence.Warehouse.Document в виде XML или пустую строку, если документ не найден.

Событие может быть обработано модулем «Вызов серверных операций». В этом случае в серверную операцию в переменную сессии Document передается результат вызова предыдущего обработчика в цепочке обработчиков данного события (если задано несколько обработчиков). Если предыдущих обработчиков в цепочке нет или результат вызова предыдущего обработчика = null, то создается новый документ с указанным типом. Также в сессию заносятся следующие значения: DocumentTypeName (тип запрошенного документа), DocumentId (ид. или штрихкод запрошенного документа), GetDocumentMode (режим получения документа: 0 — по ид., 1 — по штрихкоду). Возвращаемое значение (документ) операция должна занести в переменную Result.

Описание в панели управления:

```
C#

<ид. коннектора>:ПолучитьДокумент
```

Ид. коннектора — задается в панели управления.

Например: OneC_Connector:ПолучитьДокумент

Пример функции

Для «1С:Предприятия 8»:

```
1C

Функция ПолучитьДокумент(UserId, КодДокумента, ОперацияТСД, Режим, mXmlDoc =
Неопределено) Экспорт
//код обработки поиска документа
...
ДокументТСД = Новый COMОбъект ("Cleverence.Warehouse.Document");
//заполнение документа ТСД данными из документа 1С
...
Результат = StorageConnector.ToXml (ДокументТСД);
Возврат Результат;
КонецФункции
```

события сервера, коннекторы

Не нашли что искали?

Задать вопрос в техническую поддержку

Событие сервера Mobile SMARTS «ПолучитьСписокДокументов»

Последние изменения: 2024-03-26

Событие о запросе получения списка документов с сервера. Вызывается при открытии списка документов на терминале и периодически при нахождении внутри списка. Позволяет реализовать поиск и отбор документов по параметрам непосредственно в базе учетной системы, без предварительной выгрузки.

Вызов события может происходить только в том случае, если включен режим отображения серверных документов в списке документов терминала.

Если в панели управления в типах документов «Показывать в списке документы на сервере» проставлено «Нет», то никакого события на сервере не происходит. В случае, когда проставлено «Да», то терминал по необходимости будет запрашивать документы с сервера, заодно вызывая событие получения списка.

Параметры

Имя параметра
Тип данных
Описание
ИдПользователя
string (строка)
Идентификатор пользователя Mobile SMARTS, при работе которого на ТСД, произошел вызов обработчика.
ТипДокумента
string (строка)
Тип документов, которые запрашиваются с терминала. Если тип документа null (неопределено) или пустая строка — запрашиваются все документы, вне зависимости от их типа.
XmlСессии
string (строка)
Объект сессии сервера Mobile SMARTS, сериализованный в xml. Параметр передается, только если включена настройка «Добавлять объект сессии в вызов событий»

Возвращаемое значение

В качестве возвращаемого значения ожидается коллекция описаний документов Cleverence.Warehouse.DocumentDescriptionCollection, сохраненная в виде XML.

Событие может быть обработано модулем «Вызов серверных операций», в этом случае в серверную операцию в параметр Result передается результат вызова предыдущего обработчика в цепочке обработчиков данного события (если задано несколько обработчиков). Результат также возвращается в переменной Result. Серверная операция может изменить полученный из внешней системы список документов или вернуть свой собственный.

Описание в панели управления:

```
C#

<ид. коннектора>:ПолучитьСписокДокументов
```

Ид. коннектора — задается в панели управления.

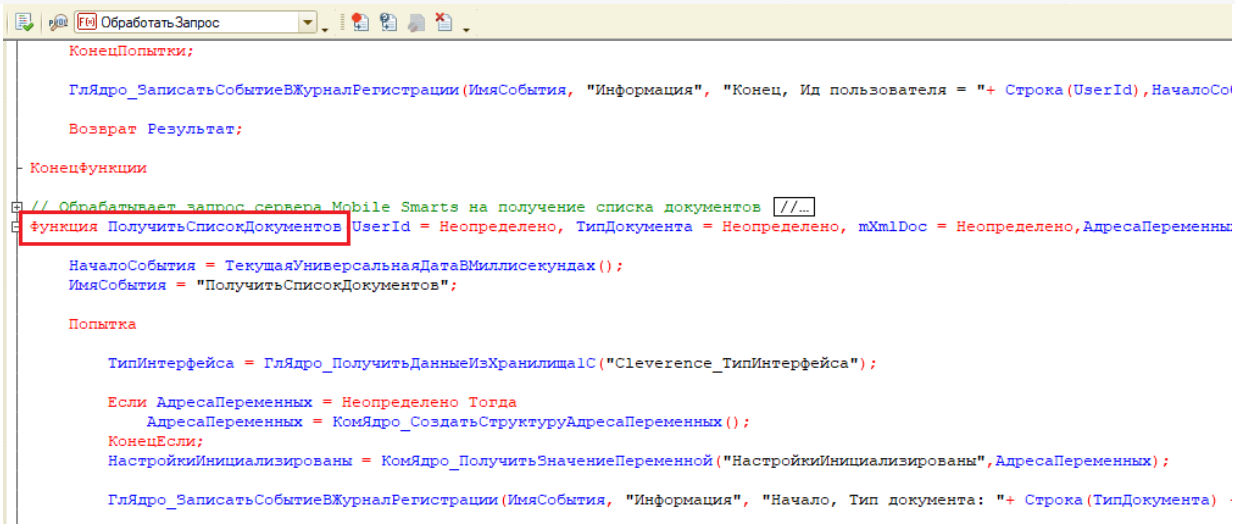
Например: OneC_Connector:ПолучитьСписокДокументов

Пример функции для «1С:Предприятия 8»

Ниже приведен пример метода «ПолучитьСписокДокументов» для использования в «1С:Предприятии» с целью получения списка документов в нужном формате. Процесс формирования списка документов на стороне 1С подробно не рассматривается (в данном примере он происходит в рамках функции «ОтобратьДокументы (...)»).

```
1C

Функция ПолучитьСписокДокументов(UserId, ТипДокумента, mXmlDoc = Неопределено)
Экспорт
StorageConnector = Новый COMObject("Cleverence.Warehouse.StorageConnector");
ДокументыТСД = Новый COMObject("Cleverence.Warehouse.DocumentDescriptionCollection");
ОтобранныеДокументы = ОтобратьДокументы(UserId, ТипДокумента);
Для Каждого СтрокаДок из ОтобранныеДокументы Цикл
DocDescr = Новый COMОбъект("Cleverence.Warehouse.DocumentDescription");
DocDescr.Id = XMLСтрока(СтрокаДок.Ссылка);
cDescr.Name = Строка(СтрокаДок.Ссылка);
DocDescr.DocumentTypeName = ТипДокумента;
DocDescrs.Add(DocDescr);
КонецЦикла;
Результат = StorageConnector.ToXml(DocDescrs);
Возврат Результат;
КонецФункции
```

В готовой интеграции «1С:Предприятия» и Mobile SMARTS рассматриваемая функция уже добавлена в модуль основной обработки «КлеверенсТСД_ОсновнаяОбработка.epf». Если вы делаете самостоятельную интеграцию Mobile SMARTS с «1С:Предприятием» с помощью основной обработки, то вы можете использовать готовые методы.



Итогом выполнения функции «ПолучитьСписокДокументов» будет отображение документов из обработки 1С на экране мобильного устройства, подключенного к учетной системе (при работе в онлайн-режиме).

события сервера, коннекторы

Не нашли что искали?

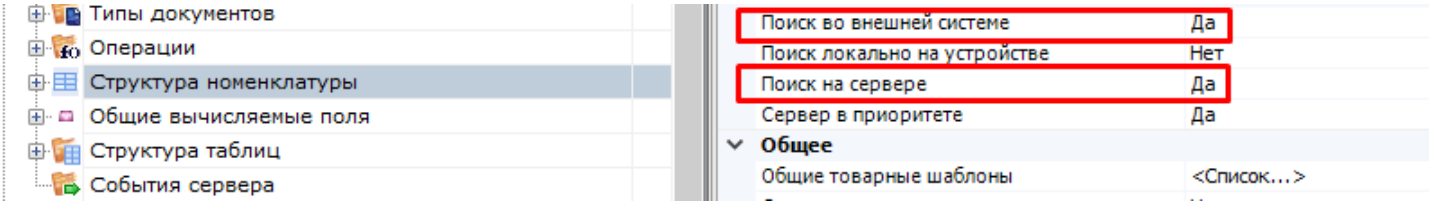
Задать вопрос в техническую поддержку

Событие сервера Mobile SMARTS «ПолучитьТовар»

Последние изменения: 2024-03-26

Событие о том, что товар был запрошен терминалом по идентификатору или каким-либо реквизитам (штрихкод, артикул, код) с сервера и не был найден в серверном справочнике. Позволяет реализовать поиск товаров непосредственно в базе учетной системы, без предварительной выгрузки.

Для того, чтобы выполнялся вызов обработчика события, в настройках номенклатуры должен быть включен «Поиск на сервере» и «Поиск во внешней системе».



Параметры

Имя параметра
Тип данных
Описание
ИдТовара
string (строка)
Идентификатор или атрибут для поиска (штрихкод, артикул, код) для поиска.
ИдУпаковки
string (строка)
Идентификатор упаковки товара. Заполняется только для режимов 2 и 3.
ИдПользователя
string (строка)
Идентификатор пользователя Mobile SMARTS, при работе которого на ТСД, произошел вызов обработчика.
Режим
int (целое число)
Режим поиска **=0 — поиск по коду товара, без указания конкретной упаковки.** **=1 — поиск по штрихкоду, артикулу или любой другой характеристике товара.** **=2 — поиск по коду товара с заданной упаковкой.** **=3 — поиск упаковки для товара.**
XmlСессии
string (строка)
Объект сессии сервера Mobile SMARTS, сериализованный в xml. Параметр передается, только если включена настройка «Добавлять объект сессии в вызов событий»

Возвращаемое значение

В качестве возвращаемого значения ожидаются разные типы данных, в зависимости от режима:

Cleverence.Warehouse.Product, Cleverence.Warehouse.Packing, Cleverence.Warehouse.PackedProduct или Cleverence.Warehouse.PackedProductCollection в виде XML.

Режим 0:

Требуется найти товар по его идентификатору. Возвращаемое значение Cleverence.Warehouse.Product в виде XML. Если товар не найден, необходимо вернуть null (неопределено).

> **Следует возвращать товар с базовым типом упаковки. Если другие упаковки потребуются в процессе работы, программа запросит их с помощью режима 2 или 3.**

В данном режиме событие вызывается, когда на ТСД было вызвано получение товара по идентификатору. Например, при отображении строк документа в списке выполняется получение товаров по их идентификаторам из строк документа.

Режим 1:

Требуется найти товар по штрихкоду, артикулу или иному реквизиту, по которому предусмотрен поиск. Возвращаемое значение Cleverence.Warehouse.PackedProduct или Cleverence.Warehouse.PackedProductCollection в виде XML. Если товар не найден, необходимо вернуть null (неопределено).

В данном режиме событие вызывается при сканировании на ТСД штрихкода товара в действии Выбор номенклатуры или при вводе пользователем значения, по которому требуется найти товар.

Режим 2:

Требуется найти товар по коду товара с заданной упаковкой. Возвращаемое значение Cleverence.Warehouse.Product в виде XML. Если товар не найден, необходимо вернуть null (неопределено). Отличие от режима 0 состоит в том, что в режиме 0 должен возвращаться товар с базовой упаковкой, а в данном режиме запрашивается товар с конкретной упаковкой.

Режим 3:

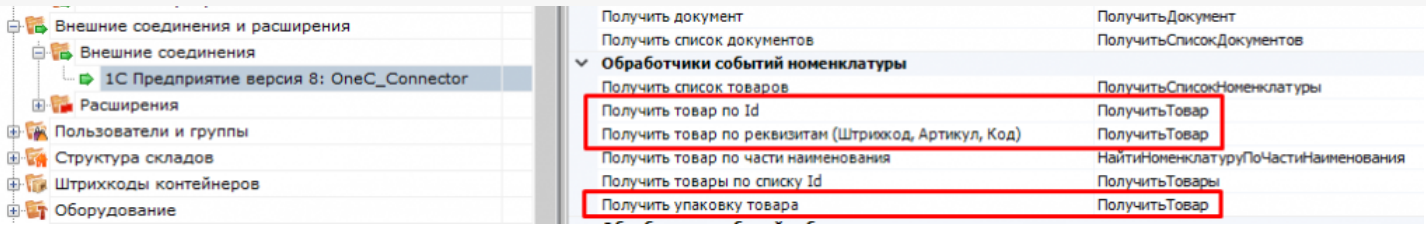
Требуется найти упаковку товара. Возвращаемое значение Cleverence.Warehouse.Packing в виде XML. Если товар не найден, необходимо вернуть null (неопределено).

Описание в панели управления:

```
C#

<ид. коннектора>:ПолучитьТовар
```

Ид. коннектора — задается в панели управления.

Например: OneC_Connector:ПолучитьТовар



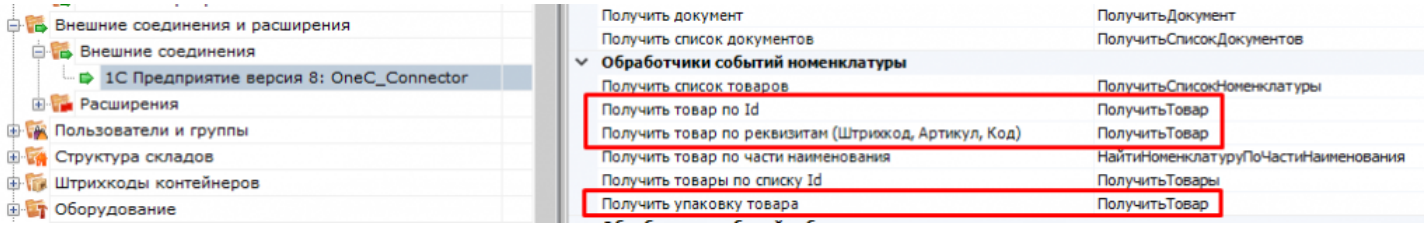
При использовании коннекторов, у которых имена обработчиков указываются в свойствах самого коннектора (производные от ConnectorTypical), есть возможность указать разные обработчики для разных режимов получения товара:

Режим 0 и Режим 2 — «Получить товар по Id»,

Режим 1 — «Получить товар по реквизитам (Штрихкод, Код, Артикул)»,

Режим 3 — «Получить упаковку товара».

Параметры обработчиков те же, что описаны выше.



Пример функции

Для «1С:Предприятия 8»:

```
1C

Функция ПолучитьТовар(ПараметрНоменклатуры, ПараметрУпаковки, UserId, Режим = 0,
mXmlDoc = Неопределено) Экспорт
Результат = Неопределено;
//код обработки поиска товара
...
StorageConnector = Новый COMОбъект("Cleverence.Warehouse.StorageConnector");
//создание объекта драйвера для передачи данных в компоненту
PackedProductCollection = Новый COMОбъект("Cleverence.Warehouse.PackedProductCollection");
...
//преобразование объекта драйвера в формат XML
Результат = StorageConnector.ToXml (PackedProductCollection);
//передача данных на сервер Mobile SMARTS
Возврат Результат ;
КонецФункции
```

Кроме изложенного варианта возврата результата из обработчика в виде xml-представления объектов Cleverence.Warehouse.Product, ProductCollection, PackedProduct и т. п., есть возможность возвращать из 1С таблицу значений. Наименования колонок таблицы значений должны начинаться на «Product_», если поле относится к товару и на «Packing_», если это поле упаковки. Например, «Product_Id», «Packing_Barcode», «Packing_Характеристика». Поле объекта PackedProduct (возвращается в режиме 2), не относящиеся

ни к товару, ни к упаковке, указывается без префиксов «Product_» и «Packing_» (например, «Quantity»).

Пример:

Product_Id	Packing_Marking	Product_Marking	Product_Barcode	Packing_Barcode	Product_Name	Product_BasePackingId	Packing_Id	Packing_Name
"dee6e1d9-5...	"K-120001"	"K-120001"	"00000000018"	"2000019017158"	"Кроссовки мужские, кожа"	"пара"	"пара"	"пара"

Не нашли что искали?

Задать вопрос в техническую поддержку

Событие сервера Mobile SMARTS «ПолучитьТоварыПоСпискуId»

Последние изменения: 2024-03-26

Обработчик события вызывается при запросе с терминала группы товаров с конкретными упаковками, если товары не найдены в выгруженном справочнике.

Для того, чтобы выполнялся вызов обработчика события, в настройках номенклатуры должен быть включен «Поиск на сервере» и «Поиск во внешней системе» .

Параметры

Имя параметра
Тип данных
Описание
ИдТоваров
Массив строк (COMSafeArray)
Массив идентификаторов запрошенных товаров.
ИдУпаковок
Массив строк (COMSafeArray)
Массив идентификаторов упаковок.
ИдПользователя
string (строка)
Идентификатор пользователя Mobile SMARTS, при работе которого на ТСД, произошел вызов обработчика.
XmlСессии
string (строка)
Объект сессии сервера Mobile SMARTS, сериализованный в xml. Параметр передается, только если включена настройка «Добавлять объект сессии в вызов событий».

Массивы «ИдТоваров» и «ИдУпаковок» имеют одинаковую длину и содержат пары «ИдТовара-ИдУпаковки» данного товара.

Событие аналогично событию «Получить товар» в режиме 2 (поиск по коду товара с заданной упаковкой). Отличие в том, что «Получить товар принимает одну пару ИдТовара-ИдУпаковки» , а «Получить товары по списку Id» набор таких пар.

Возвращаемое значение

Результат, возвращаемый функцией, должен быть в виде Xml-представления объекта Cleverence.Warehouse.ProductCollection (коллекция товаров, см. справочник). Xml-представление объектов Mobile SMARTS следует получать с помощью функции ToXml компоненты StorageConnector.

В случае 1С функция может возвращать таблицу значений, которая на сервере Mobile SMARTS преобразуется в объект коллекции. Наименования колонок таблицы значений должны начинаться на “Product_”, если поле

относится к товару и на “Packing_”, если это поле упаковки. Например, “Product_Id”, “Packing_Barcode”, “Packing_Характеристика”.

Обработчик события может найти не все запрошенные товары, тогда в таблице должны быть только строки, соответствующие найденным товарам. Если не найдено ни чего, возвращается пустая таблица.

Описание в панели управления:

```
C#

<ид. коннектора>:ПолучитьТовары
```

Ид. коннектора - задается в панели управления.

Например: OneC_Connector:ПолучитьТовары

Пример функции

Для «1С:Предприятия 8»:

C#

```
Функция ПолучитьТовары(ИдТоваров, ИдУпаковок, UserId, mXmlDoc=Неопределено) Экспорт
ЗапросНоменклатуры = Новый Запрос;
...
СписокНоменклатуры = Новый СписокЗначений;
Для Каждого Ид из ИдТоваров Цикл
СписокНоменклатуры.Добавить(ПолучитьСсылку("Номенклатура", Ид));
КонецЦикла;
ЗапросНоменклатуры.УстановитьПараметр("П", СписокНоменклатуры);
ТаблицаТоваров = ЗапросНоменклатуры.Выполнить().Выгрузить();
Результат = Новый ТаблицаЗначений;
Результат.Колонки.Добавить("Product_Id");
Результат.Колонки.Добавить("Product_Marking" );
Результат.Колонки.Добавить("Product_Barcode" ); Результат.Колонки.Добавить("Product_Name"
);
Результат.Колонки.Добавить("Product_BasePackingId" );
...
Результат.Колонки.Добавить("Packing_Id" );
Результат.Колонки.Добавить("Packing_Name" );
Результат.Колонки.Добавить("Packing_Marking" );
Результат.Колонки.Добавить("Packing_Barcode" );
...
Для Каждого СтрокаТовара Из ТаблицаТоваров Цикл
СтрокаРезультата = Результат.Добавить();
СтрокаРезультата["Product_Id"] = СтрокаТовара["ИдНоменклатуры"];
...
КонецЦикла;
Возврат Результат;
КонецФункции
```

Событие сервера Mobile SMARTS «ПолучитьТоварПоЧастиНаименования»

Последние изменения: 2024-03-26

Обработчик события вызывается, когда пользователь на терминале вводит в окне выбора номенклатуры из списка текст для поиска позиций номенклатуры.

Для того, чтобы выполнялся вызов обработчика события, в настройках номенклатуры должен быть включен «Поиск на сервере» и «Поиск во внешней системе».

Параметры

Имя параметра
Тип данных
Описание
ТекстДляПоиска
string (строка)
Текст, введенный пользователем на терминале.
ИдПользователя
string (строка)
Идентификатор пользователя Mobile SMARTS, при работе которого на ТСД, произошел вызов обработчика.
XmlСессии
string (строка)
Объект сессии сервера Mobile SMARTS (Cleverence.Warehouse.ServerSession см. справочник), сериализованный в xml. Параметр передается, только если включена настройка «Добавлять объект сессии в вызов событий»

Возвращаемое значение

Результат, возвращаемый функцией, должен быть в виде Xml-представления объекта Cleverence.Warehouse.PackedProductCollection (коллекция товаров с упаковками, см. справочник). Xml-представление объектов Mobile SMARTS следует получать с помощью функции ToXml компоненты StorageConnector.

В случае 1С функция может возвращать таблицу значений, которая на сервере Mobile SMARTS преобразуется в объект коллекции. Наименования колонок таблицы значений должны начинаться на «Product_», если поле относится к товару и на «Packing_», если это поле упаковки. Например, «Product_Id», «Packing_Barcode», «Packing_Характеристика».

Описание в панели управления:

```
C#

<ид. коннектора>:НайтиНоменклатуруПоЧастиНаименования
```

Ид. коннектора — задается в панели управления.

Например: OneC_DriverConnector:НайтиНоменклатуруПоЧастиНаименования

Пример функции

Для «1С:Предприятия 8»:

```
1C

Функция НайтиНоменклатуруПоЧастиНаименования(ТекстДляПоиска, userId,
mXmlDoc=Неопределено) Экспорт
//код обработки поиска товара
...
StorageConnector = Новый COMОбъект("Cleverence.Warehouse.StorageConnector");
//создание объекта драйвера для передачи данных в компоненту
PackedProductCollection = Новый COMОбъект("Cleverence.Warehouse.PackedProductCollection")
...
//преобразование объекта драйвера в формат XML
Результат = StorageConnector.ToXml (PackedProductCollection);
//передача данных на сервер Mobile SMARTS
Возврат Результат;
КонецФункции
```

события сервера, коннекторы

Не нашли что искали?

Задать вопрос в техническую поддержку

Событие сервера Mobile SMARTS «ПолучитьСписокТоваров»

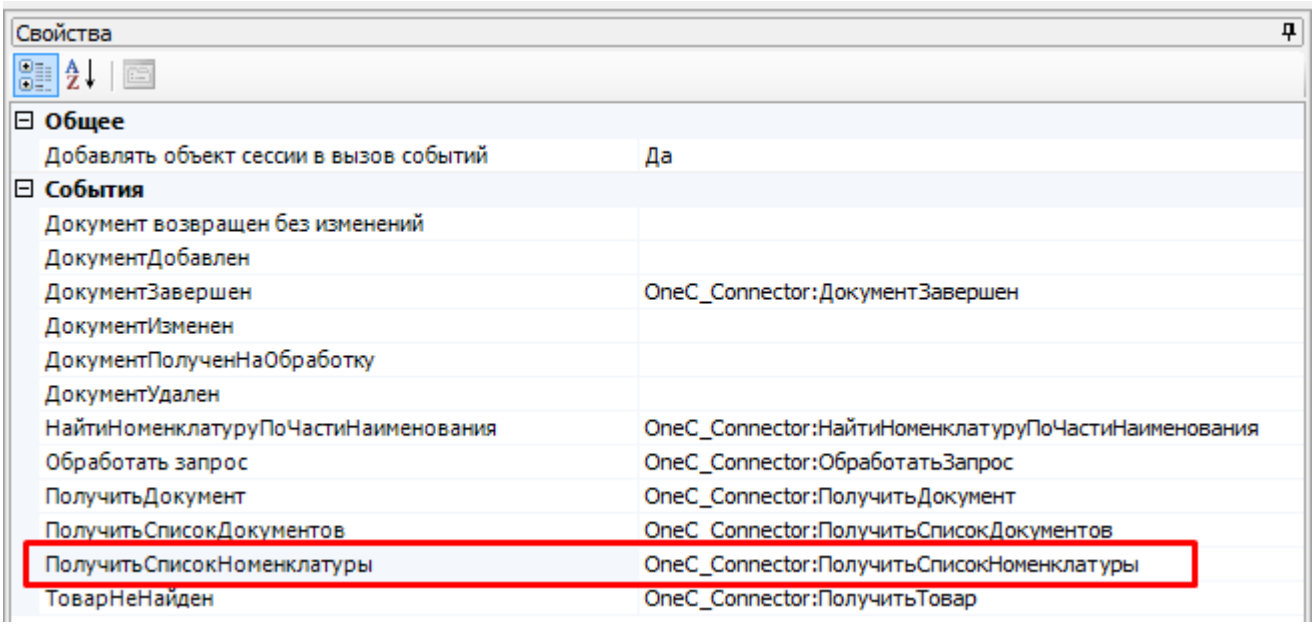
Последние изменения: 2024-03-26

Обработчик события вызывается при отображении на терминале справочника номенклатуры с сервера, позволяет организовать вывод иерархического справочника номенклатуры из учетной системы.

Аргументы:

Имя параметра	Тип данных	Описание
ИдРодителя		
string (строка)		
Идентификатор позиции (папки), вложенные элементы которой следует вернуть. Для верхнего уровня - пустая строка или Null.		
ИдПользователя		
string (строка)		
Идентификатор пользователя Mobile SMARTS, при работе которого на ТСД, произошел вызов обработчика.		
XmlСессии		
string (строка)		
Xml представление объекта Cleverence.Warehouse.ServerSession (см. справочник) или Null (если в настройках событий сервера в конфигурации Mobile SMARTS отключено добавление сессии в обработчики событий). Объект может быть загружен с помощью функции FromXml объекта Cleverence.Warehouse.StorageConnector.		

В панели управления:



<id коннектора>:<имя функции-обработчика>

При запросе позиций верхнего уровня ИдРодителя пустой. Когда пользователь на терминале при просмотре

списка номенклатуры выбирает элемент, являющийся группой, в параметр ИдРодителя передается идентификатор выбранной группы. При этом функция должна вернуть элементы, вложенные в группу (элементы могут сами являться группами или позициями номенклатуры).

Результат, возвращаемый функцией, должен быть в виде Xml-представления объекта Cleverence.Warehouse.PackedProductCollection (коллекция товаров с упаковками, см. справочник). Xml-представление объектов Mobile SMARTS следует получать с помощью функции ToXml компоненты StorageConnector.
В случае 1С функция может возвращать таблицу значений, которая на сервере Mobile SMARTS преобразуется в объект коллекции. Наименования колонок таблицы значений должны начинаться на “Product_”, если поле относится к товару и на “Packing_”, если это поле упаковки. Например, “Product_Id”, “Packing_Barcode”, “Packing_Характеристика”.

Если возвращаемый элемент является группой, у объекта PackedProduct, добавляемого в коллекцию, не должно быть заполнено свойство Packing (упаковка), этот признак используется клиентом Mobile SMARTS для определения того, что внутрь элемента при выборе его из списка можно зайти. В случае таблицы значений в строке, представляющей группу, должны быть заполнены только колонки товара (Product) и не заполнены колонки упаковки (Packing).

Пример

1C

```
Функция ПолучитьСписокНоменклатуры(ИдРодителя, userId, mXmlDoc=Неопределено) Экспорт

Родитель = Неопределено;

Если Не ЗначениеЗаполнено(ИдРодителя) Тогда

Родитель = Справочники.Номенклатура.ПустаяСсылка();

Иначе

Попытка

Гуид = Новый УникальныйИдентификатор(ИдРодителя);

Родитель = Справочники.Номенклатура.ПолучитьСсылку(Гуид);

Исключение

Родитель = Справочники.Номенклатура.ПустаяСсылка();

КонецПопытки;

КонецЕсли;

ЗапросНоменклатуры = Новый Запрос;

...

ЗапросНоменклатуры.УстановитьПараметр("Родитель", Родитель);
```

```
ТаблицаТоваров = ЗапросНоменклатуры.Выполнить().Выгрузить();

Результат = Новый ТаблицаЗначений;

Результат.Колонки.Добавить("Product_Id");

Результат.Колонки.Добавить("Product_Marking" );

Результат.Колонки.Добавить("Product_Barcode" ); Результат.Колонки.Добавить("Product_Name"
);

Результат.Колонки.Добавить("Product_BasePackingId" );

...

Результат.Колонки.Добавить("Packing_Id" );

Результат.Колонки.Добавить("Packing_Name" );

Результат.Колонки.Добавить("Packing_Marking" );

Результат.Колонки.Добавить("Packing_Barcode" );

...

Для Каждого СтрокаТовара Из ТаблицаТоваров Цикл

СтрокаРезультата = Результат.Добавить();

СтрокаРезультата["Product_Id"] = СтрокаТовара["ИдНоменклатуры"];

СтрокаРезультата["Product_Name"] = Строка(СтрокаТовара["Номенклатура"]);

Если СтрокаТовара["ЭтоГруппа"] Тогда

СтрокаРезультата["Product_ЭтоГруппа"] = Истина;

Иначе

...

СтрокаРезультата["Product_Marking"] = Артикул;

СтрокаРезультата["Product_Barcode"] = Код;

СтрокаРезультата["Packing_Barcode"] = ШК;

СтрокаРезультата["Product_BasePackingId"] = ФлагБазовойЕдиницы;

СтрокаРезультата["Packing_Id"] = ЕдиницаИзмерения;

СтрокаРезультата["Packing_Name"] = ЕдиницаИзмерения;
```

```
...

КонецЕсли;

КонецЦикла;

Возврат Результат;

КонецФункции
```

события сервера, коннекторы

Не нашли что искали?

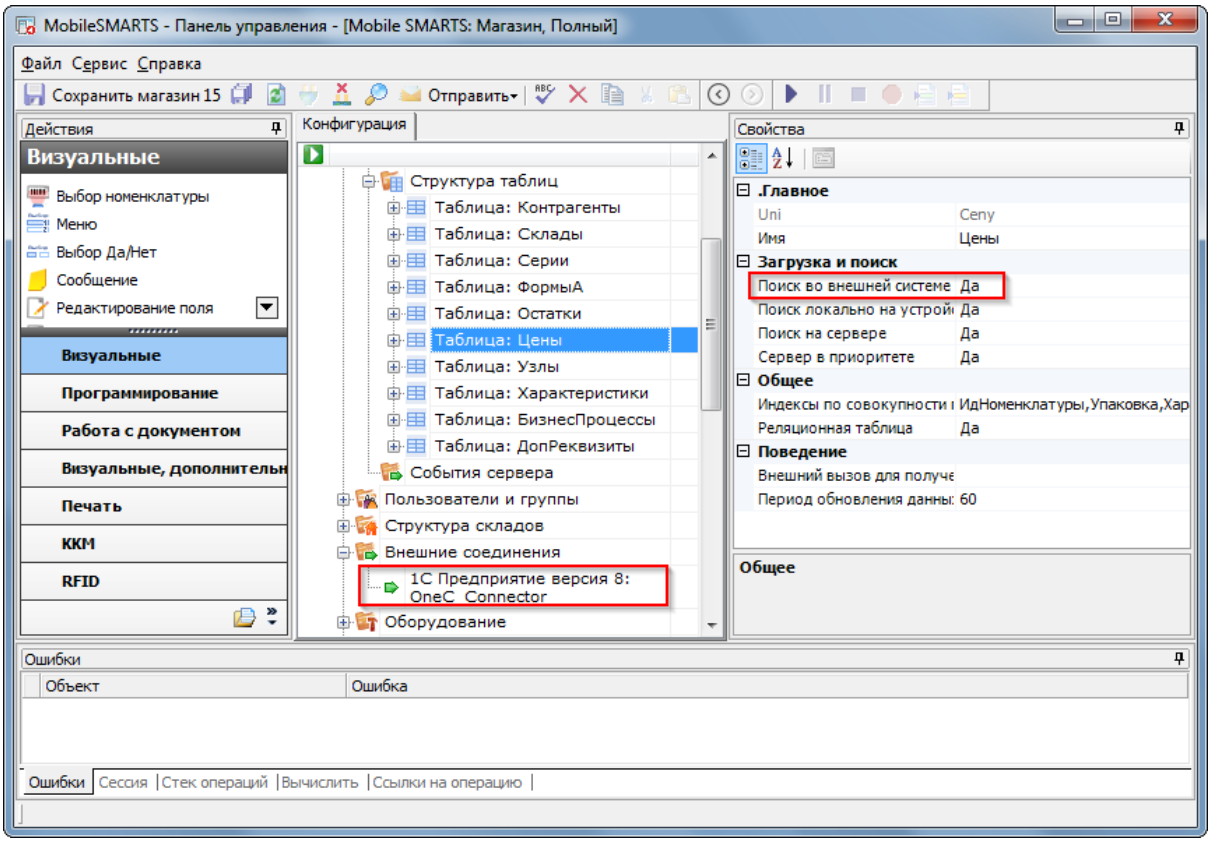
Задать вопрос в техническую поддержку

Событие сервера Mobile SMARTS «ОбработатьЗапрос»

Последние изменения: 2024-03-26

Обработчик события вызывается при выполнении запроса к серверной таблице, имеющейся в конфигурации Mobile SMARTS. Данный механизм позволяет организовать получение по запросу с терминала различных данных, которые при работе без онлайн вызовов, хранятся в выгружаемых таблицах (например, цены, остатки, серии).

Чтобы обработчик вызывался при запросе данных из таблицы, в настройках таблицы должно быть включено «Поиск на сервере — Да», «Поиск во внешней системе — Да».



Параметры

Имя параметра
Тип данных
Описание
ЗапросXML
string (строка)
Массив идентификаторов запрошенных товаров.
ИдПользователя
string (строка)
Идентификатор пользователя Mobile SMARTS, при работе которого на ТСД, произошел вызов обработчика.

XmlСессии
string (строка)
Объект сессии сервера Mobile SMARTS, сериализованный в xml. Параметр передается, только если включена настройка «Добавлять объект сессии в вызов событий»

Для обработки запроса необходимо загрузить из xml объект Cleverence.Warehouse.DocumentQuery с помощью функции FormXML объекта Cleverence.Warehouse.StorageConnector.

DocumentQuery содержит следующие свойства, используемые при обработке запроса:

Свойство	Тип данных	Описание
From	Строка	Наименование таблицы, из которой выполняется выборка данных (например, Остатки). Равно наименованию одной из таблиц, имеющихся в конфигурации Mobile SMARTS.
WhereRootElement	Ссылка на объект IQueryElement	Корневой элемент синтаксического дерева выражения условия в запросе.

По значению, заданному в свойстве From, обработчик должен определить, откуда следует выбрать данные. Далее нужно обойти элементы дерева выражения условия и наложить соответствующее условие на запрос учетной системы. IQueryElement содержит:

Свойство	Тип данных	Описание
IsGroup	Boolean	Признак, является ли элемент группой (И, ИЛИ, НЕ)
Группа: IsGroup = Истина например, (ИдНоменклатуры == «cbcf493e-55bc-11d9-848a-00112f43529a» && Характеристика == «светло-серый» && ВидЦеныВнутр == 1)		
GroupTypeStr	Строка	Тип группы (какой логической операцией объединены условия в группе). Одно из следующих значений: «Or» (Или), «And» (И), «Not» (Не).
Elements	QueryElementCollection	Коллекция элементов группы, может содержать как отдельные элементы условия (IsGroup=Ложь), так и другие группы (IsGroup=Истина).

Элемент: IsGroup = Ложь например, ИдНоменклатуры == «cbcf493e-55bc-11d9-848a-00112f43529a»

ComparisonTypeStr
Строка
Вид сравнения. Одно из следующих значений: «==» (равно), «!=» (не равно), «<» (меньше), «>» (больше), «<=» (меньше или равно), «>=» (больше или равно), «Contains» (содержит), «StartsWith» (начинается с).
LeftValue
Строка
Наименование сравниваемого поля
RightValue
Object
Значение сравниваемого поля (число, строка).

Возвращаемое значение

Результат, возвращаемый функцией, должен быть в виде Xml-представления объекта Cleverence.Warehouse.RowCollection (коллекция строк таблицы, см. справочник). Xml-представление объектов Mobile SMARTS следует получать с помощью функции ToXml компоненты StorageConnector.

В случае 1С функция может возвращать таблицу значений. Наименования полей (как в случае строк Cleverence.Warehouse.Row, так и в случае колонок таблицы значений 1С) должны соответствовать наименованиям полей таблицы из конфигурации Mobile SMARTS.

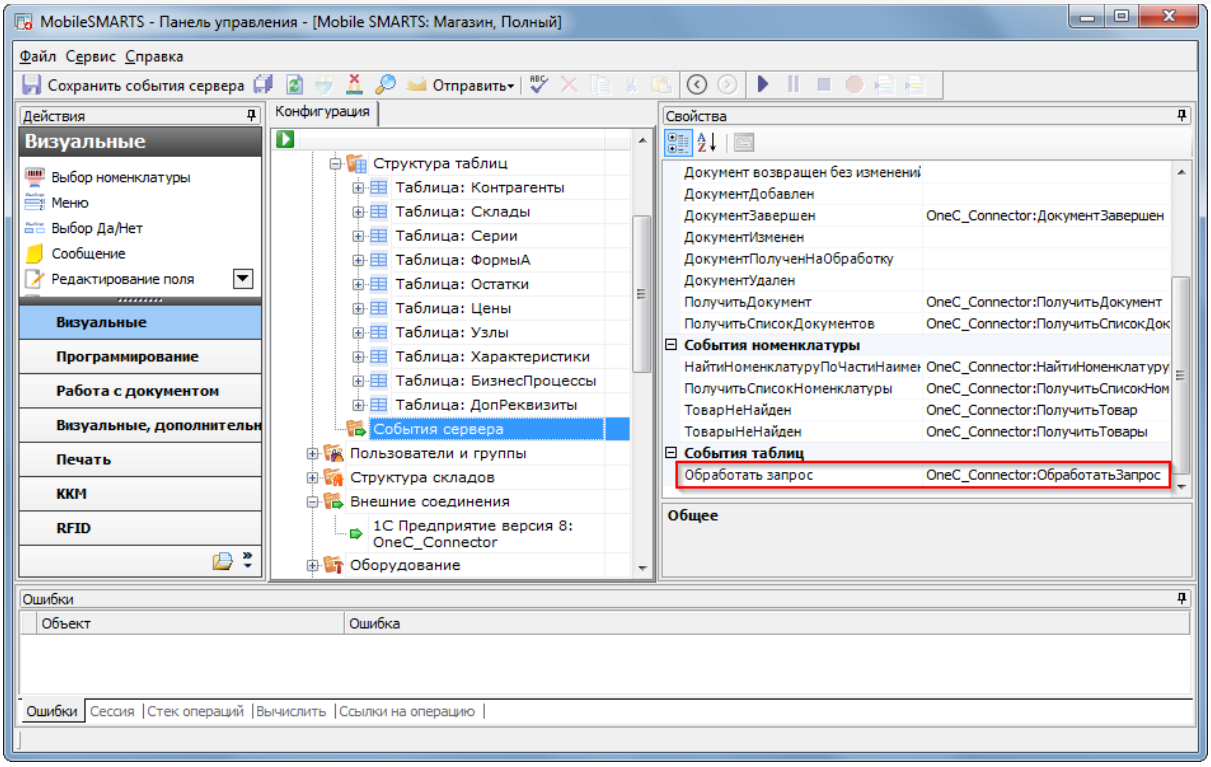
Описание в панели управления:

1C

<ид. коннектора>:ОбработатьЗапрос

Ид. коннектора — задается в панели управления.

Например: OneC_Connector:ОбработатьЗапрос



Пример функции

1С

```
Функция ОбработатьЗапрос(запросXML, userId, mXmlDoc=Неопределено) Экспорт
...
docQuery = connector.FromXML(запросXML);
ИмяМакетаСхемыКомпоновки = ПолучитьИмяМакета(docQuery.From);
СхемаКомпоновки = ЭтотОбъект.ПолучитьМакет(ИмяМакетаСхемыКомпоновки);
...
КомпоновщикНастроекКомпоновкиДанных = Новый
КомпоновщикНастроекКомпоновкиДанных;
...
ЗаполнитьОтбор(docQuery.From, docQuery.WhereRootElement,
КомпоновщикНастроекКомпоновкиДанных.Настройки.Отбор.Элементы);
ПроцессорВывода = Новый
ПроцессорВыводаРезультатаКомпоновкиДанныхВКоллекциюЗначений;
ТабВыгрузки = ПроцессорВывода.Вывести(ПроцессорКомпоновки);
Возврат ТабВыгрузки;
КонецФункции
Процедура ЗаполнитьОтбор(ИмяОтбора, queryElement, Элементы)
Если queryElement = NULL Или queryElement = Неопределено Тогда
Возврат;
КонецЕсли;
Если queryElement.IsValueItem Тогда
Возврат;
КонецЕсли;
Если Не queryElement.IsGroup Тогда
ЭлементОтбора = Элементы.Добавить(Тип("ЭлементОтбораКомпоновкиДанных"));
ЭлементОтбора.ВидСравнения = ПолучитьВидСравнения(queryElement.ComparisonTypeStr);
ЭлементОтбора.ЛевоеЗначение = Новый ПолеКомпоновкиДанных(queryElement.LeftValue);
ЭлементОтбора.ПравоеЗначение = ПолучитьЗначениеДляОтбора(ИмяОтбора,
queryElement.LeftValue, queryElement.RightValue);
Иначе
ЭлементОтбора = Элементы.Добавить(Тип("ГруппаЭлементовОтбораКомпоновкиДанных"));
ЭлементОтбора.ТипГруппы = ПолучитьТипГруппыОтбора(queryElement.GroupTypeStr);
Для Инд = 0 По queryElement.Elements.Count-1 Цикл
queryEl = queryElement.Elements.Item(Инд);
ЗаполнитьОтбор(ИмяОтбора, queryEl, ЭлементОтбора.Элементы);
КонецЦикла;
КонецЕсли;
КонецПроцедуры
```

события сервера, коннекторы

Не нашли что искали?

Задать вопрос в техническую поддержку

Ручная отправка событий для документов на сервере Mobile SMARTS

Последние изменения: 2024-03-26

Часто в процессе разработки или внедрения необходимо как-то добиться повторного возникновения события на сервере (например, «Документ завершен»).

Ранее разработчику приходилось повторно отправлять документ на ТСД и завершать его на мобильном устройстве, чтобы сервер заново вызвал обработку события.

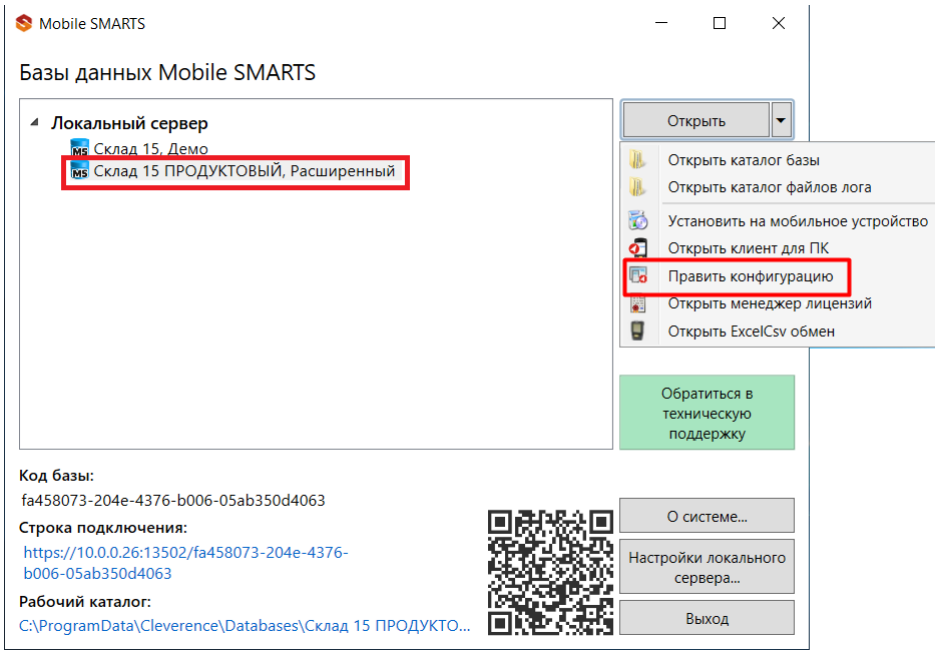
Начиная с версии 3.2 платформы Mobile SMARTS, в панели управления становится доступной функция ручной отправки основных событий для документов:

- ДокументЗавершен.
- ДокументДобавлен.
- ДокументИзменен.
- ДокументНазначен.

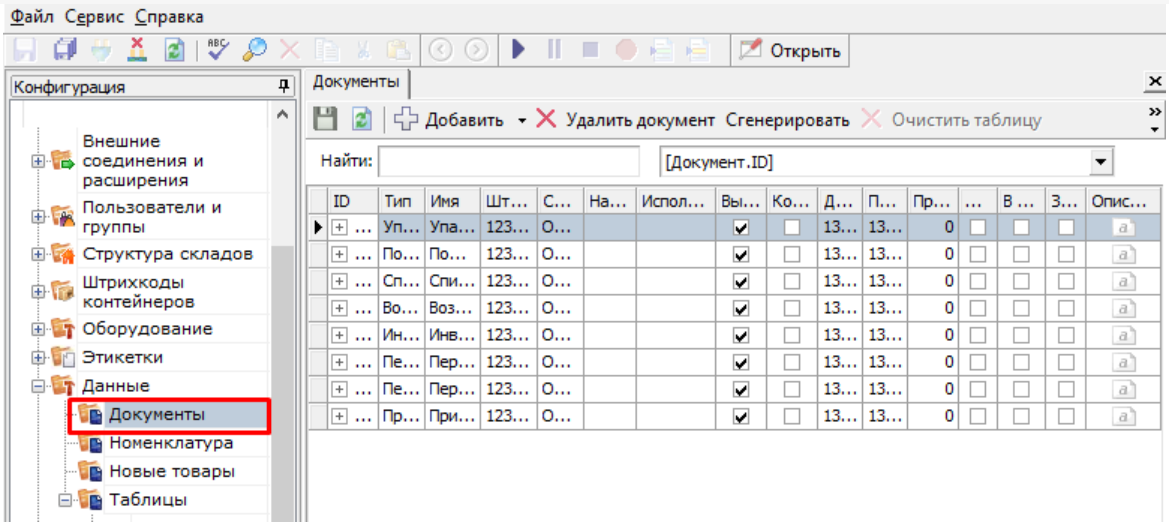
Это значит, что для вызова серверного события теперь не нужно производить какие-либо действия с документом на устройстве, достаточно просто нажать на одну кнопку в панели управления.

Для этого необходимо:

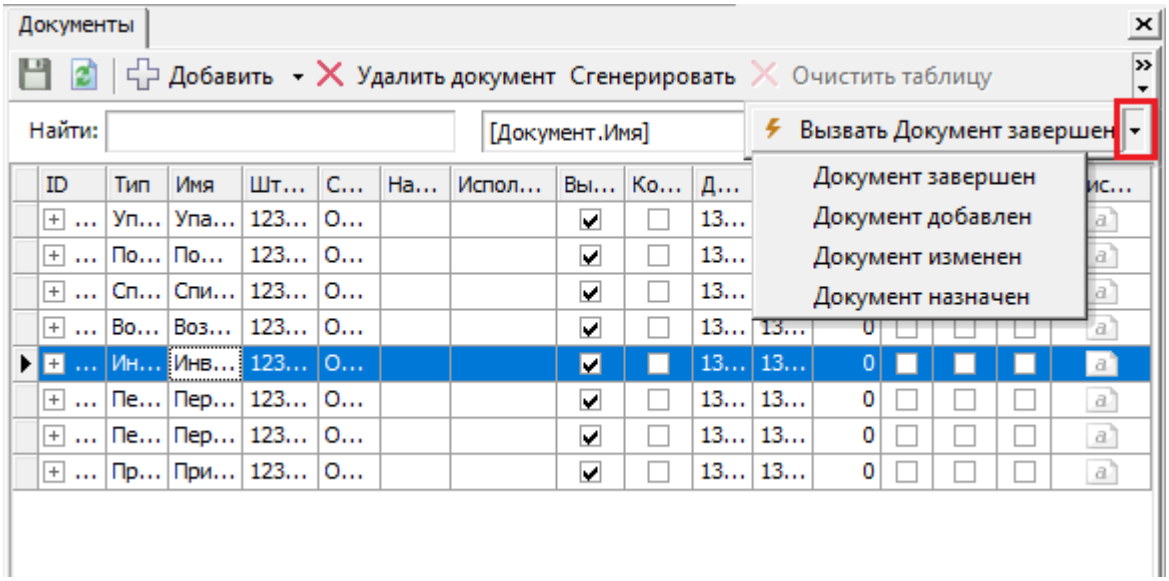
1. В менеджере баз Mobile SMARTS выбрать нужную базу и открыть для нее панель управления с помощью кнопки «Править конфигурацию».



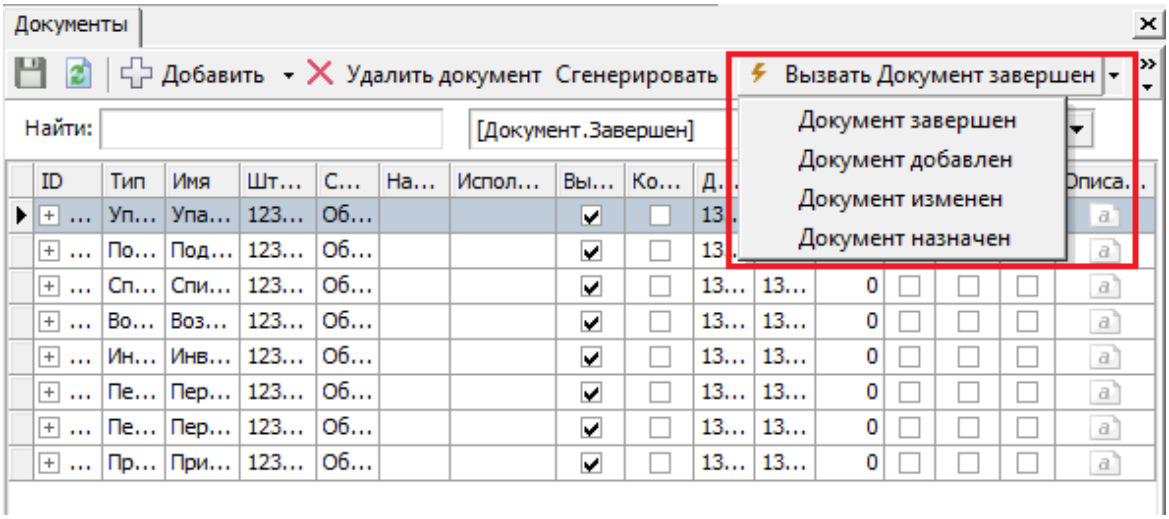
2. В панели управления открыть список документов данной базы.



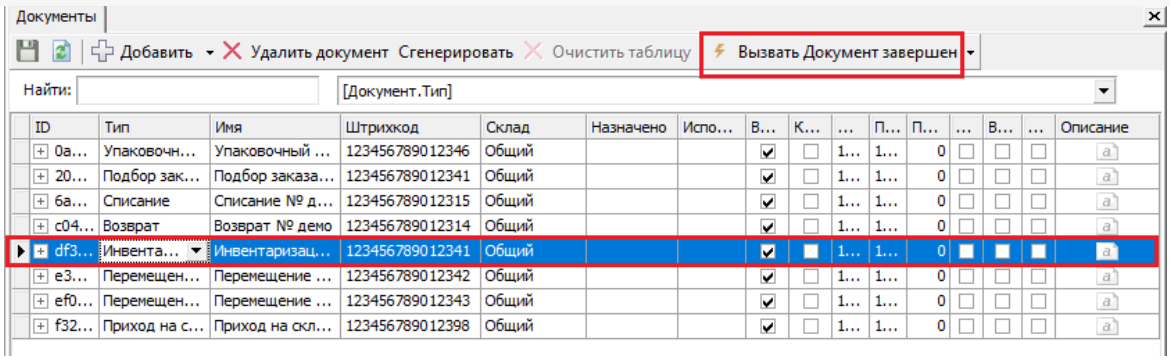
3. Добавить кнопку вызова события на панель инструментов с помощью кнопки  . После этого выбрать нужное событие из выпадающего списка.



4. Кнопка для вызова выбранного события появится на панели инструментов. Для того чтобы сменить вызываемое событие, выберите новое из выпадающего списка.



5. Далее для вызова события выберите нужный документ из списка, и нажмите на кнопку.



Не нашли что искали?

Задать вопрос в техническую поддержку

Объектная модель подключаемых модулей

Последние изменения: 2024-03-26

Рассмотрим объектную модель подключаемых модулей. Базовым интерфейсом для всех подключаемых модулей является IConnectivityBase:

[C#]

```
/// <summary>
/// Базовый интерфейс для подключаемых модулей
/// </summary>
public interface IConnectivityBase
{
/// <summary>
/// Уникальный идентификатор модуля.
/// </summary>
string Id
{
get;
}
/// <summary>
/// Флаг, указывающий разрешена или нет работа модуля.
/// </summary>
bool Enabled
{
get;
set;
}
/// <summary>
/// Инициализация (запуск) модуля. После вызова этой функции он должен перейти в  рабочее
состояние
/// </summary>
void Initialize();
/// <summary>
/// Инициализировано соединение или нет.
/// </summary>
bool Initialized
{
get;
}
/// <summary>
/// Функция проверки дополнительных лицензионных ограничений модуля
/// </summary>
param name="supportedSystems">Список внешних модулей, упомянутых в лицензионном
файле</param>
 void
CheckLicenseLimitations(System.Collections.Generic.List<Cleverence.Licensing.LicenseExternalSystem

supportedSystems);
}
```

От интерфейса IConnectivityBase наследуется интерфейс соединения с внешней системой IConnector и интерфейс расширения (плагина) IPlugin.

[C#]

```
/// <summary>
/// Интерфейс соединения с внешней системой. Важное отличие - наличие метода Invoke, позволяющего вызывать любые
какие-то методы из внешней системы
/// </summary>
public interface IConnector: IConnectivityBase
{
/// <summary>
/// Тайм-аут при вызове функций внешней системы через коннектор.
/// </summary>
int Timeout
{
get;
set;
}
/// <summary>
/// Поведение при наступлении тайм-аута. ThrowException - вызывать исключение, ReInvoke - завершить попытку вызова,
переинициализировать подключение и сделать снова вызов.
/// </summary>
TimeoutBehavaior TimeoutBehavaior
{
get;
set;
}
/// <summary>
/// Сообщает, что коннектор сам внутри обрабатывает таймауты, и внешняя обработка не требуется
/// </summary>
bool IsSelfTimeoutBehavior { get; }
/// <summary>
/// Для ActiveMQ коннектора добавляет в аргументы DeviceInfo.
/// </summary>
bool IsSupportDeviceInfoInArgs
{
get;
}
/// <summary>
/// Вызов метода внешней системы.
/// </summary>
/// <param name="methodName">Имя метода.</param>
/// <param name="args">Параметры.</param>
/// <returns>Результат (null, если метод ничего не возвращает).</returns>
object InvokeMethod(string methodName, object[] args);
}
/// <summary>
/// Базовый интерфейс плагина.
/// Немного расширяет стандартный IConnectivityBase явным методом остановки модуля.
/// </summary>
public interface IPlugin: IConnectivityBase
```

```
{
/// <summary>
/// Деинициализация (остановка) модуля. После вызова этой функции он должен остановить
свою работу.
/// </summary>
void Deinitialize();
}
```

Основное отличие между внешним соединением (коннектором) и расширением (плагином) в том, что коннектор имеет функцию InvokeMethod, с помощью которой выполняется вызов внешней системы при запросах с ТСД и при обработке событий сервера. Свойства Timeout, TimeoutBehavaior, IsSelfTimeoutBehavior позволяют установить тайм-аут при вызовах и задать поведение при наступлении тайм-аута. Плагин, в отличии от коннектора, не поддерживает вызовы. И коннектор и плагин запускаются при вызове функции Initialize сервером Mobile SMARTS. Вызов Initialize происходит при старте сервера базы данных Mobile SMARTS (если в настройках базы включена опция «Инициализировать подключения к внешним системам при старте») или при обращении к коннектору (вызов с ТСД или обработка события сервера). Конкретная реализация Initialize определяется особенностями системы, к которой выполняется подключение. Например, может быть выполнено создание COM-объекта для работы со внешней системой и выполнено подключение к базе данных системы с заданным логином/паролем. В случае плагина Initialize запускает модуль в работу. Например, может начаться слежение за папкой или запущен таймер для выполнения периодических операций. Плагин имеет функцию Deinitialize, при вызове которой работа плагина прекращается. Работа плагина может быть остановлена через панель управления (контекстное меню узла плагина -> «Остановить»), также остановка происходит при остановке сервера базы данных Mobile SMARTS.

Еще одним интерфейсом, производным от **IConnectivityBase**, является **IEventsProcessor**:

```
[C#]

/// <summary>
/// Базовый интерфейс для модуля, умеющего обрабатывать события.
/// Для реализации обработки событий необходимо пользоваться
/// атрибутами EventProcessorAttribute, DocumentEventProcessorAttribute
/// </summary>
public interface IEventsProcessor: IConnectivityBase
{
}
```

Интерфейс не добавляет каких-либо функций в **IConnectivityBase**, а служит маркером для модулей, которые используются при обработке событий сервера.

Существует базовая реализация интерфейса **IConnectivityBase**, от которой можно наследоваться при реализации своих коннекторов — класс **ConnectorBase**.

Если разрабатываемый коннектор будет использоваться для обработки событий сервера, рекомендуется в качестве базового использовать другой класс — **ConnectorTypical**, производный от **ConnectorBase**. Данный класс позволяет указывать имена обработчиков событий сервера в свойствах самого коннектора.

[C#]

Код класса приведен не полностью

```
/// <summary>
/// Базовый класс коннектора, позволяющий указывать имена обработчиков событий в свойствах самого коннектора.
/// При обработке событий вызываются функции класса, помеченные атрибутом EventProcessor или DocumentEventProcessor.
/// При разработке новых коннекторов рекомендуется наследоваться от данного класса.
/// </summary>
public abstract class ConnectorTypical : ConnectorBase
{
/// <summary>
/// Имя функции-обработчика события "Документ добавляется".
/// Событие вызывается перед добавлением документа в базу Mobile SMARTS из внешней системы.
/// </summary>
[EventHandlerProperty(EventType.DocumentAdding)]
public string DocumentAddingHandler { get; set; }
.....
/// <summary>
/// Обработчик вызывается перед добавлением документа на сервер из внешней системы.
/// </summary>
/// <param name="di"></param>
/// <param name="documentTypeName"></param>
/// <param name="doc"></param>
/// <returns>Возвращается переданный документ или null, если требуется отклонить добавление документа (будет вызвано исключение и документ не будет сохранен на сервере).
</returns>
[DocumentEventProcessor(DocumentType = "*", EventType = EventType.DocumentAdding)]
public virtual object DocumentAdding(DeviceInfo di, string documentTypeName, Document doc)
...
}
```

Имена обработчиков в панели управления указываются в свойствах коннектора:

Указание обработчиков событий сервера в свойствах коннектора позволяет организовать вызов нескольких обработчиков для одного события. Порядок вызова обработчиков редактируется в панели управления в свойствах узла «События сервера»:

Класс ConnectorTypical содержит виртуальные функции, вызываемые при обработке событий сервера, данные функции помечены атрибутом DocumentEventProcessor для событий документов и атрибутом EventProcessor для остальных событий (номенклатуры и таблиц). Например:

```
[C#]

<summary>
/// Обработчик вызывается сервером при запросе документа с ТСД.
/// </summary>
/// <returns>Возвращается документ (Document), может в сериализованном в xml-строку виде.
Если документ не был получен, то возвращается null. Также может возвращаться InvokeResult.
</returns>
[DocumentEventProcessor(DocumentType = "*", EventType = EventType.GetDocument)]
public virtual object GetDocument(DeviceInfo di, string documentTypeName, string  identity,
GetDocumentMode mode)
```

Атрибут DocumentEventProcessor имеет параметры: DocumentType, позволяет ограничить вызов данного обработчика определенными типами документов, «*» — вызов будет выполнен для всех типов документов, EventType — тип события. Атрибут EventProcessor имеет один параметр: EventType — тип события. При реализации своих обработчиков в производных классах также следует использовать атрибуты DocumentEventProcessor и EventProcessor.

Типы событий, задаваемые перечислением EventType:

События документов
DocumentAdding
Документ добавляется. **Событие, возникающее в процессе добавления документа на сервере. Вызывается при выгрузке документа в базу Mobile SMARTS из внешней системы.**
DocumentAdded
Документ добавлен. **Событие, возникающее после добавления документа на сервере. Вызывается при выгрузке документа в базу Mobile SMARTS из внешней системы.**
DocumentAppointing
Документ назначается пользователю. **Событие о том, что документ готов передаваться на мобильное устройство. Вызывается при запросе с ТСД получения документа для работы.**
DocumentAppointed
Документ назначен пользователю. **Событие о том, что документ захвачен на обработку. Вызывается в момент, когда документ был передан на ТСД для работы с ним.**
DocumentChanged
Документ изменен. **Событие об изменении документа. Вызывается при сохранении документа на сервер в процессе работы на ТСД при использовании в конфигурации Mobile SMARTS действия «Сохранение документа на сервер».**
DocumentFinished
Документ завершен. **Событие о завершении обработки документа. Вызывается при получении сервером завершенного документа с ТСД.**

DocumentReleased
Документ возвращен с ТСД без обработки. Вызывается когда документ с терминала был возвращен пользователем вызовом release (вернуть документ без изменений).
DocumentRemoved
Документ удален. Событие об удалении документа с сервера. Удаление выполняется внешней системой.
ListDocuments
Получить список документов. Вызывается при входе в список документов на ТСД. Используется в паре с событием ПолучитьДокумент для того, чтобы реализовать выбор документов прямо из учетной системы, без предварительной выгрузки.
GetDocument
Получить документ. Вызывается при выборе в списке документов на ТСД того документа, который еще не был выгружен (попал в этот список с помощью события ПолучитьСписокДокументов).
События номенклатуры
ProductNotFound
Получить товар. Событие о том, что продукт был запрошен терминалом по коду (штрихкод, артикул, код) с сервера и не был найден в серверном справочнике.
ProductsNotFound
Получить товары по списку Id. Событие о том, что несколько товаров были запрошены терминалом по идентификаторам с сервера и не были найден в серверном справочнике. Позволяет реализовывать получение сразу нескольких товаров из базы учетной системы без предварительной выгрузки.
GetProductByPartOfName
Получить товар по части наименования. Вызывается при вводе в строку поиска в списке номенклатуры на ТСД. Используется для того, чтобы реализовать поиск номенклатуры по части наименования прямо из учетной системы.

GetCellById
Получить ячейку по идентификатору
GetCellByBarcode
Получить ячейку по штрихкоду
ListProducts
Получить список товаров. Вызывается при входе в список номенклатуры на ТСД. Используется для того, чтобы реализовать выбор номенклатуры прямо из учетной системы, без предварительной выгрузки.
События таблиц
TableRequest
Обработать запрос. Событие вызывается при запросе данных из таблицы, определенной в конфигурации Mobile SMARTS. Позволяет получать строки таблицы он-лайн по запросу с ТСД.

Не нашли что искали?

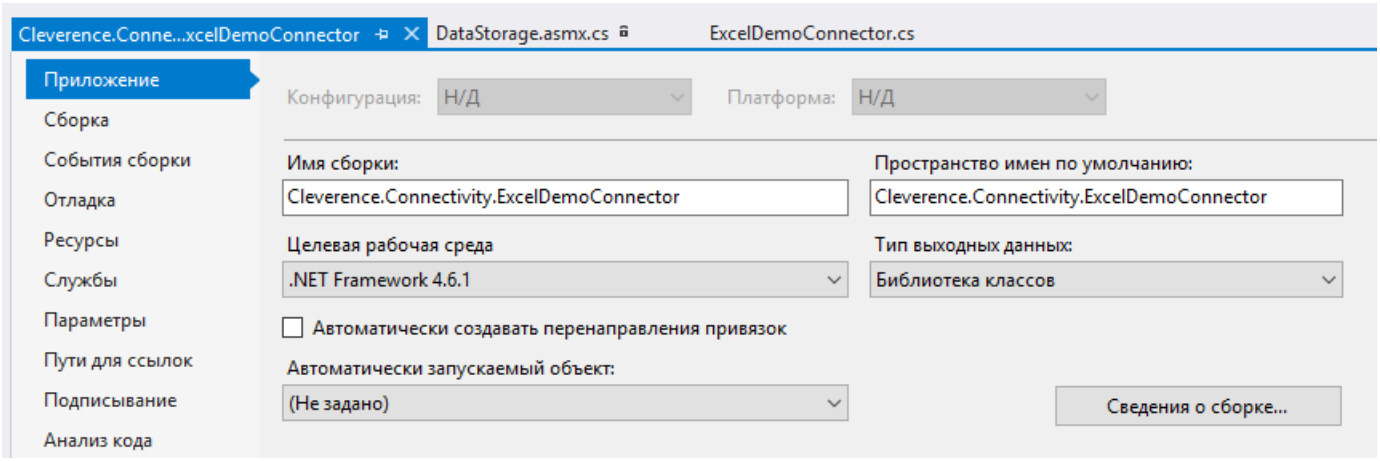
Задать вопрос в техническую поддержку

Разработка собственного коннектора к внешней системе

Последние изменения: 2024-03-26

Для работы коннектора под управлением сервера Mobile SMARTS и настройки параметров коннектора через панель управления создаются два файла dll: первая dll, предназначенная для сервера, размещается в <Папка базы Mobile SMARTS>\ Server\DataService\ bin\. Вторая dll, для панели управления — в <Папка базы Mobile SMARTS>\ Control panel\ Addins. Имена файлов должны иметь вид Cleverence.Connectivity.*.dll (например, Cleverence.Connectivity.MyConnector.dll — для сервера и Cleverence.Connectivity.MyConnector.Panel.dll — для панели управления).

В «Решении» (Solution) в Visual Studio нужно создать два проекта с типом выходных данных «Библиотека классов»:



Скачать заготовку коннектора (Решение Visual Studio 2019 с двумя проектами).

Пространство имен класса коннектора должно начинаться на Cleverence.Connectivity, целевая рабочая среда .NET Framework 4.6.1.

Для серверной версии коннектора в «Ссылки» (Reference Assemblies) нужно добавить следующие dll:

Cleverence.Barcoding.dll

Cleverence.Common.dll

Cleverence.Connectivity.dll

Cleverence.DataCollection.dll

Cleverence.MobileSMARTS.dll

Данные библиотеки находятся по пути <папка установки Mobile SMARTS>\ Server\DataService\ Bin (по умолчанию, C:\ Program Files (x86)\ Cleverence Soft\ Mobile SMARTS\ Server\ DataService\ Bin).

В проект коннектора для панели управления нужно добавить в «Ссылки» (Reference Assemblies):

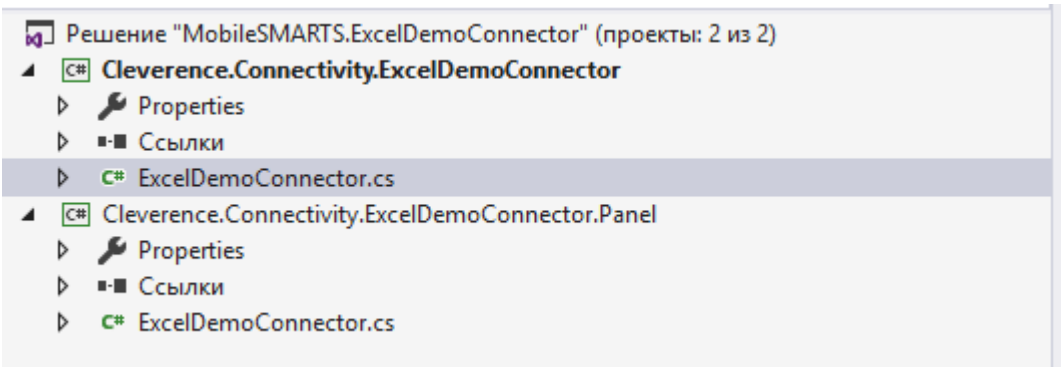
Cleverence.DataCollection.dll

Cleverence.MobileSMARTS.ComConnector.dll

Библиотека Cleverence.MobileSMARTS.ComConnector.dll находится по пути <папка установки Mobile SMARTS>\ Control Panel (по умолчанию C:\ Program Files (x86)\ Cleverence Soft\ Mobile SMARTS\ Control Panel).

Рекомендуется скопировать все нужные dll в отдельную папку рядом с файлом «Решения» (Solution) и добавить ссылки на библиотеки из этой папки.

В каждый из проектов (для сервера и для панели управления) нужно добавить по файлу *.cs для исходного кода класса коннектора.



Класс коннектора должен реализовывать интерфейс IConnector (находится в пространстве имен Cleverence.Connectivity). Можно наследовать класс своего коннектора от класса ConnectorTypical (также из Cleverence.Connectivity), который реализует IConnector и, кроме того, содержит свойства для указания имен обработчиков событий сервера и виртуальные функции для вызова обработчиков.

Обе версии класса коннектора (для сервера и для панели управления) должны иметь одинаковое имя и находится в одинаковых пространствах имен (namespace), начинающихся на Cleverence.Connectivity (например, Cleverence.Connectivity.MyConnector).

При наследовании от ConnectorTypical в серверной версии коннектора следует перегрузить функции: Initialize (выполняет инициализацию коннектора), Deinitialize (выполняет деинициализацию), InvokeMethod (выполняет вызов во внешнюю систему). Именно InvokeMethod реализует основной функционал коннектора по работе с внешней системой. Также следует перегрузить свойство Initialized (возвращает признак, инициализирован ли коннектор, true — инициализирован, false — нет). При необходимости, могут быть перегружены функции обработки событий сервера (например, GetProduct — получение товара при запросе с ТСД, DocumentFinished — на сервер с ТСД попал завершенный документ и др.). Если не перегружать функции обработки событий, при возникновении событий будет вызываться функция InvokeMethod, в которую передается имя обработчика события, указанное в настройках и соответствующие событию аргументы (см. События сервера).

Версия коннектора для панели управления также наследуется от ConnectorTypical. Перегружать какие-либо функции в этом случае не нужно. Для подключения коннектора к базе внешней системы обычно требуются определенные настройки (адрес базы внешней системы, имя пользователя/ пароль и т.п). Для того, чтобы иметь возможность редактировать нужные настройки и чтобы настройки сохранялись, требуется добавить свойства в классы коннектора для панели управления и сервера. Заготовка коннектора:

[C#]

```
Серверная часть:
namespace Cleverence.Connectivity.DemoConnector
{
public class DemoConnector : ConnectorTypical
{
public DemoConnector()
{
}
public string MyProperty
{
get;
set;
```

```
set;
}
public override bool Initialized
{
get
{
throw new NotImplementedException();
}
}
public override void Initialize()
{
throw new NotImplementedException();
}
public override void Deinitialize()
{
throw new NotImplementedException();
}
public override object InvokeMethod(string methodName, object[] args)
{
throw new NotImplementedException();
}
}
}
```

Часть для панели управления:

```
namespace Cleverence.Connectivity.DemoConnector
 {
public class DemoConnector : ConnectorTypical
{
public DemoConnector()
{
}
public string MyProperty
{
get;
set;
}
}
}
```

В серверной части нужно реализовать Initialized, Initialize, Deinitialize, InvokeMethod. Для примера добавлено свойство MyProperty.

Собранную dll серверной версии размещаем в <Папка базы Mobile SMARTS>\Server\DataService\bin\, версию для панели управления в <Папка базы Mobile SMARTS>\Control panel\Addins.

Сервер Mobile SMARTS в целях безопасности выполняет проверку цифровой подписи загружаемых сборок. Если после размещения неподписанного файла dll в <Папка базы Mobile SMARTS>\Server\DataService\bin\> перезапустить службу сервера Mobile SMARTS, в логе сервера базы данных dataserver_*.log (в C:\ProgramData\Cleverence\Logs) появится сообщение:

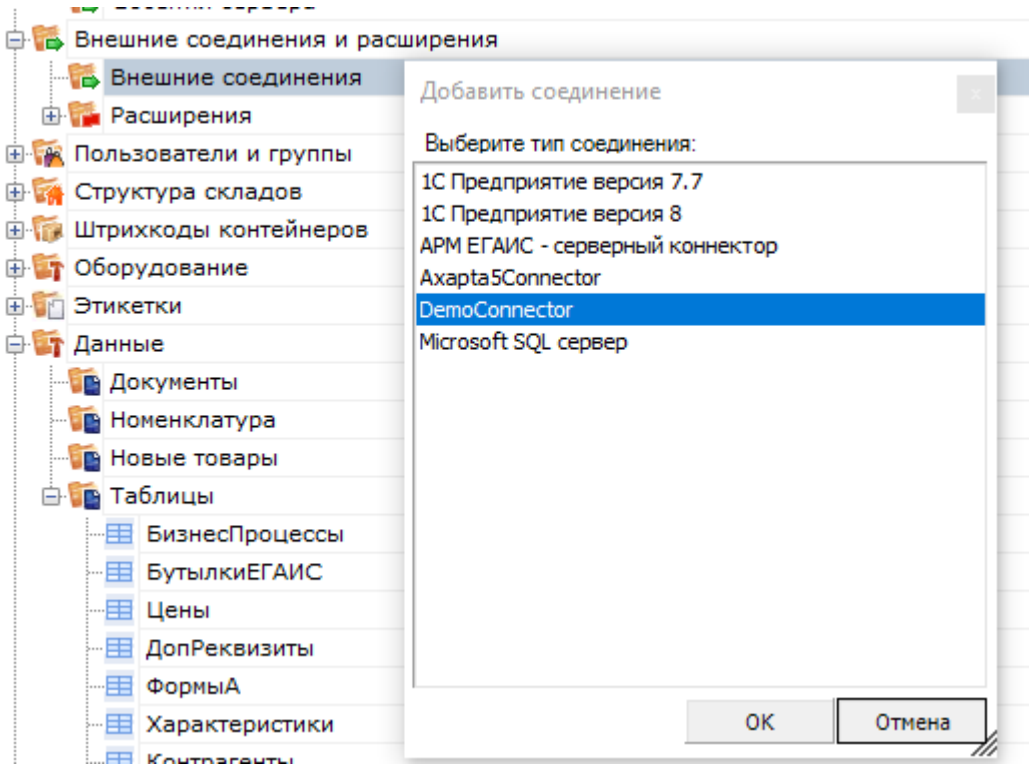
2019-08-05 10:08:08.6537|ERROR|NLogger.WriteNlogEvent| Коннекторы не загружены! Обнаружена неподписанная сборка: C:\ProgramData\Cleverence\Базы Mobile SMARTS\e3945857-308f-4829-92e2-720dc11d1bec\Server\DataService\bin\Cleverence.Connectivity.DemoConnector.dll

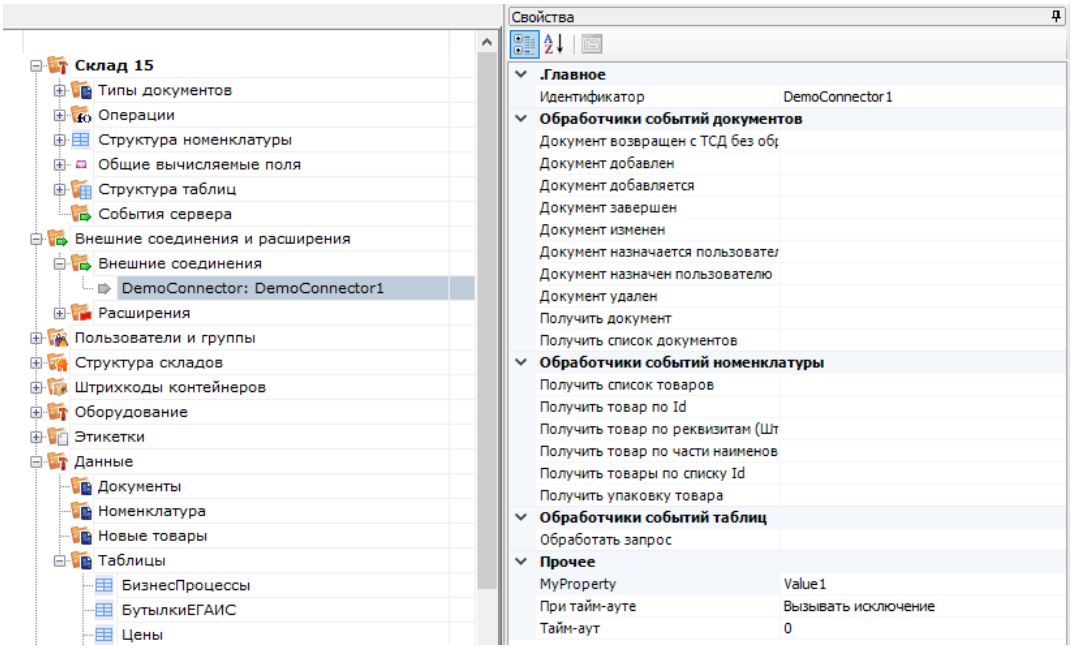
Видим, что dll коннектора не загрузилась. В некоторых случаях загрузка неподписанной сборки приводит

к остановке сервера. Для того, чтобы можно было проверить работу коннектора и выполнить отладку, сервер нужно запустить с ключом Debug из командной строки (службу сервера перед этим нужно остановить):

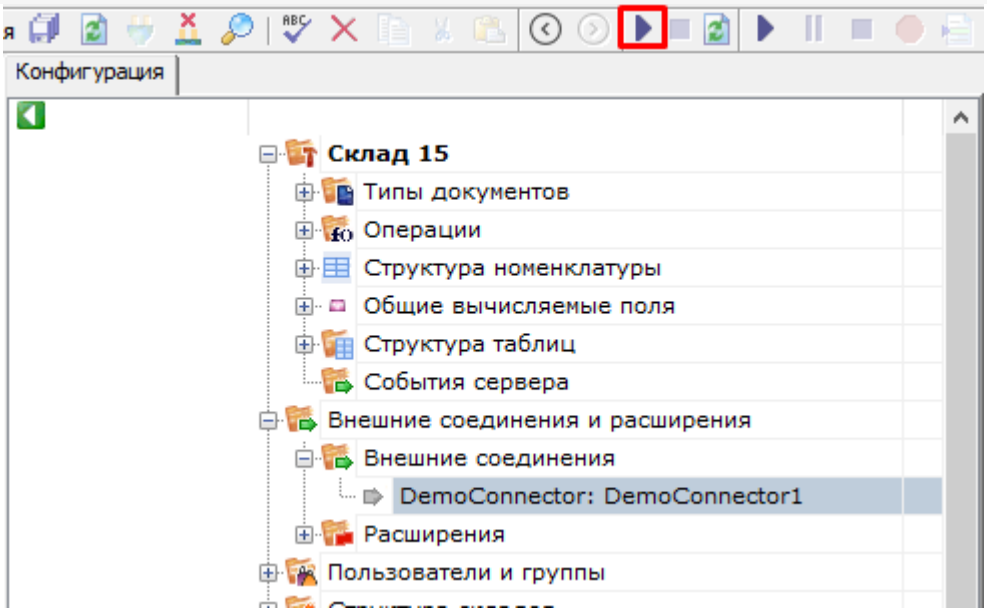
C:\Program Files (x86)\Cleverence Soft\Mobile SMARTS\Server\Cleverence.MobileSMARTS.Server.exe /debug

После этого в панели управления можно добавить коннектор в конфигурацию:





Настройка параметров коннектора выполняется через панель свойств. Когда настройка выполнена, сохраним конфигурацию. После этого можно запустить коннектор с помощью кнопки «Пуск»:



Если требуется отладка кода коннектора, рядом с файлом коннектора в <Папка базы Mobile SMARTS>\Server\DataService\bin\ следует разместить файл *.pdb (см. Тестирование и выпуск разработанного коннектора). Когда отладка закончена, обратитесь в техническую поддержку «Клеверенс» для подписания dll.

Скачать заготовку коннектора (Решение Visual Studio 2019 с двумя проектами).

Не нашли что искали?

Задать вопрос в техническую поддержку

Пример разработки коннектора к внешней системе

Последние изменения: 2024-03-26

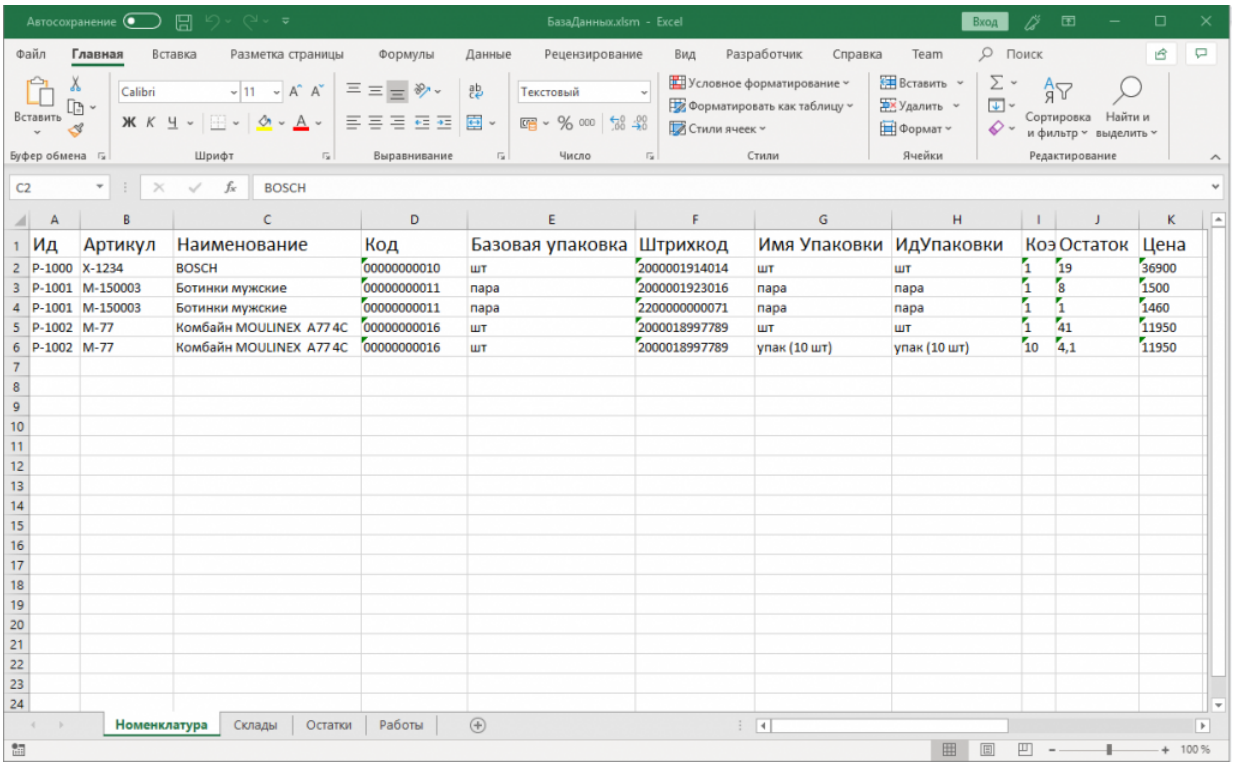
Для примера разработаем коннектор, который будет выполнять получение данных из файла Excel онлайн по запросам с ТСД. В примере будет рассмотрена, в том числе, перегрузка функций-обработчиков событий сервера для получения номенклатуры, документов и обработки запросов к таблицам.

Постановка задачи

Требуется разработать коннектор, который будет по запросам с ТСД получать данные из файлов Excel. Используются следующие справочники: «Номенклатура», «Склады», «Остатки». Должна быть обеспечена работа с документами «Поступление», на ТСД должен отображаться список документов для работы, подготовленных на сервере. Пользователь ТСД выбирает документ, принимает товар, после завершения работы данные записываются в исходный документ. Используется база Mobile SMARTS «Магазин 15».

Подготовка данных

Справочники будут храниться в одном файле (книге) Excel на листах: «Номенклатура», «Склады», «Остатки». Файл со справочниками имеет фиксированное название «БазаДанных.xlsm» (книга Excel с поддержкой макросов) и располагается в заданной папке (путь к папке указывается в настройках коннектора). Файлы документов «Поступления» будут находиться в этой же папке и называться «Поступление №00001.xlsx», «Поступление №00002.xlsx» и т. д. В архиве с исходниками данные находятся в папке Xlsx, состав полей см. в файлах.



	A	B	C	D	E	F	G	H	I	J	K
1	Ид	Артикул	Наименование	Код	Базовая упаковка	Штрихкод	Имя Упаковки	ИдУпаковки	Коэ	Остаток	Цена
2	P-1000	X-1234	BOSCH	00000000010	шт	2000001914014	шт	шт	1	19	36900
3	P-1001	M-150003	Ботинки мужские	00000000011	пара	2000001923016	пара	пара	1	8	1500
4	P-1001	M-150003	Ботинки мужские	00000000011	пара	2200000000071	пара	пара	1	1	1460
5	P-1002	M-77	Комбайн MOULINEX A77 4C	00000000016	шт	2000018997789	шт	шт	1	41	11950
6	P-1002	M-77	Комбайн MOULINEX A77 4C	00000000016	шт	2000018997789	упак (10 шт)	упак (10 шт)	10	4,1	11950

Разработка и тестирование коннектора

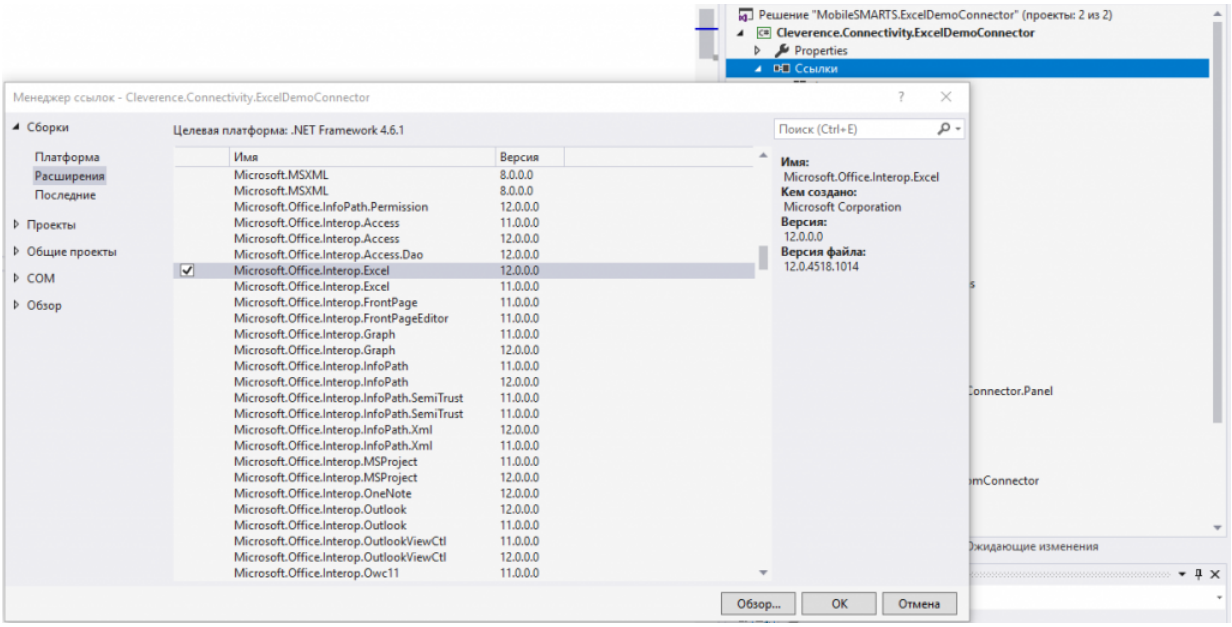
В Visual Studio создаем «Решение» (solution) и два проекта в нем:

Cleverence.Connectivity.ExcelDemoConnector (для сервера) и
Cleverence.Connectivity.ExcelDemoConnector.Panel (для панели управления).

Добавляем в ссылки (Reference assemblies) в серверный проект Cleverence.Connectivity.ExcelDemoConnector: Cleverence.Common.dll, Cleverence.Connectivity.dll, Cleverence.DataCollection.dll, Cleverence.MobileSMARTS.dll.

В проект для панели управления Cleverence.Connectivity.ExcelDemoConnector.Panel: Cleverence.DataCollection.dll, Cleverence.MobileSMARTS.ComConnector.dll.

Для работы с файлом Excel будем использовать COM-объект Excel.Application (на ПК должен быть установлен Microsoft Excel), добавим в ссылки в проект для сервера Microsoft.Office.Interop.Excel:



Добавим в каждый из проектов по файлу ExcelDemoConnector.cs, в котором будет находится класс коннектора ExcelDemoConnector. Базовым классом для ExcelDemoConnector является ConnectorTypical.

Добавим свойство, с помощью которого можно задать путь к папке с файлами Excel, используемыми для обмена данными:

[C#]

```
public string DatabaseFolder
{
get;
set;
}
```

Свойство должно быть как в серверном варианте коннектора, так и в варианте для Панели управления. Других настроек для нашего коннектора не требуется, займемся разработкой серверной части, которая будет выполнять обмен данными с Excel.

Добавим приватное поле Excel.Application excel для хранения ссылки на COM-объект Excel. Теперь первым делом нам нужно реализовать, функцию Initialize, которая переводит коннектор в рабочее состояние:

```
[C#]

public override void Initialize()
{
this.Dispose();
if (!this.Enabled)
throw new InvalidOperationException(this.GetType().Name + " is not enabled.");
this.excel = new Excel.Application();
this.excel.Visible = false;
}
```

Вызываем Dispose, чтобы выполнить деинициилиазацию, если ранее коннектор был инициализирован. Если вызовы коннектора запрещены через данель управления (Enabled == false), вызываем исключение. Если вызовы разрешены, создаем COM-объект.

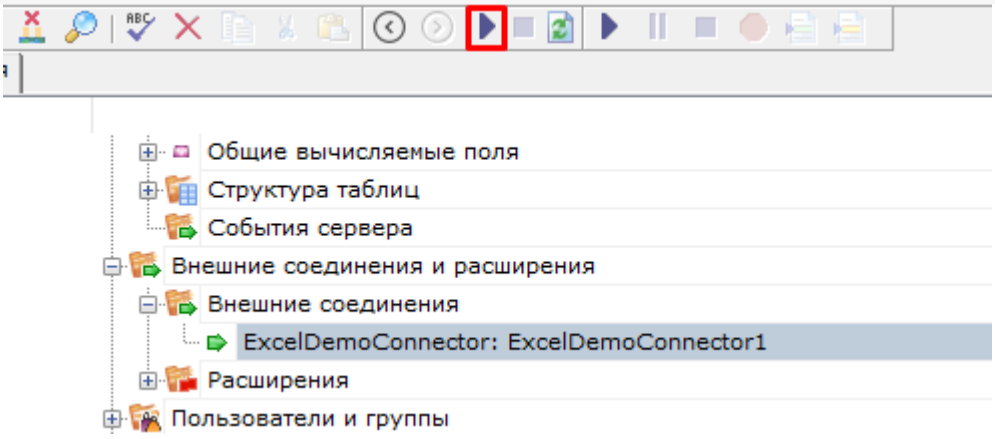
Деинициализация:

```
[C#]

public override void Deinitialize()
{
if (this.excel != null)
{
this.excel.Quit();
this.excel = null;
}
}
Признак того, что коннектор инициализирован:
public override bool Initialized
{
get
{
return this.excel != null;
}
}
```

Этого достаточно, чтобы проверить запуск коннектора через панель управления.

Соберем решение в Visual Studio. Развернем базу «Магазин 15 Расширенный» или «Мегамаркет». Скопируем dll для сервера в <Папка базы Mobile SMARTS>\Server\DataService\bin\, для панели управления в <Папка базы Mobile SMARTS>\Control panel\Addins. Запустим сервер в режиме отладки из командной строки: C:\Program Files (x86)\Cleverence Soft\Mobile SMARTS\Server\Cleverence.MobileSMARTS.Server.exe/debug.

Добавим коннектор в конфигурацию, сохраним и проверим, что выполняется запуск.



Реализуем получение номенклатуры онлайн из файла Excel. Когда товар запрашивается терминалом на сервере по каким-либо реквизитам (Ид, Штрихкод, Артикул) и товар не найден в выгруженном справочнике номенклатуры, выполняется вызов обработчика события «Получить товар»

В случае наследования от ConnectorTypical обработчик события может быть реализован как во внешней системе (вызов через InvokeMethod функции из внешней системы по имени), так непосредственно в коннекторе. Для реализации обработчика в самом коннекторе нужно перегрузить функцию GetProduct:

```
[C#]

[EventProcessor(EventType = EventType.ProductNotFound)]
public override object GetProduct(Cleverence.Warehouse.DeviceInfo di, string productId, string
packingId,
SearchProductMode searchMode, Cleverence.Barcoding.BarcodeData barcodeData)
{
CheckInit();
return GetProductInternal(productId, packingId, searchMode);
}
```

Не забываем указать атрибут EventProcessor (EventType = EventType.ProductNotFound). Реализацию поиска товаров по заданным реквизитам в зависимости от режима (searchMode) см. в GetProductInternal.

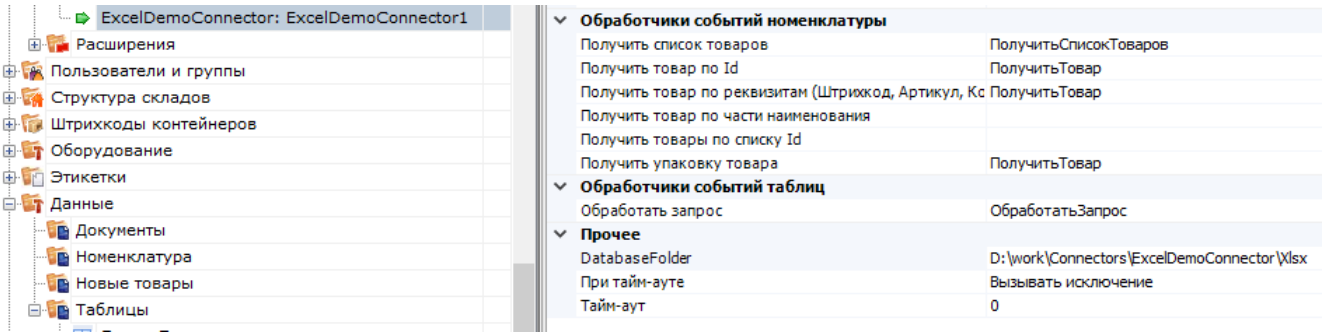
Событие «Получить список товаров» возникает, когда пользователь на ТСД заходит в список номенклатуры (выбор товара по 0 в действии «Выбор номенклатуры») и в базе нет выгруженного справочника. Перегрузим в коннекторе функцию GetProductsList:

```
[C#]

[EventProcessor(EventType = EventType.ListProducts)]
public override object GetProductsList(DeviceInfo di, string searchStr)
{
CheckInit();
return GetProductsListInternal(searchStr);
}
```

Функция возвращает PackedProductCollection со всем списком номенклатуры.

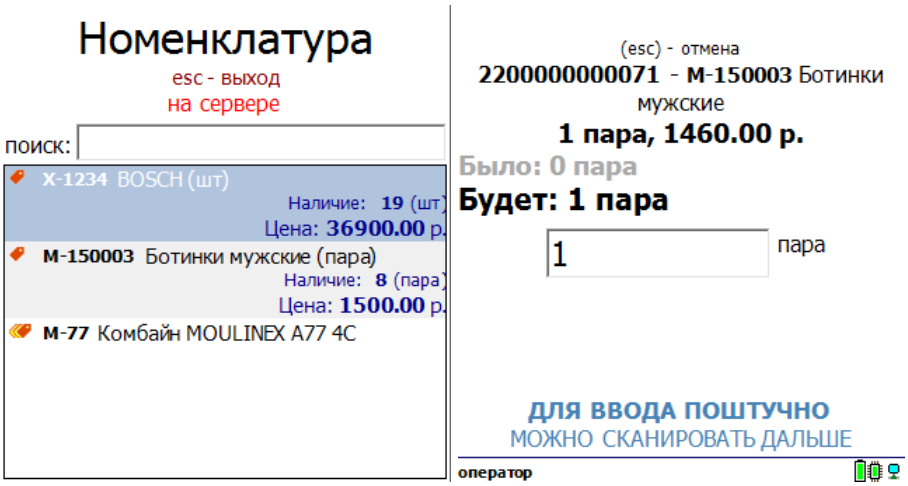
Проверим, как происходит получение номенклатуры. В настройках коннектора нужно указать путь к папке, в которой находится файл Excel со справочниками (DatabaseFolder). Нужно заполнить обработчики событий

номенклатуры: «Получить список товаров», «Получить товар по Id», «Получить товар по реквизитам» (штрихкод, артикул, код), «Получить упаковку товара».



Задание имен обработчиков требуется для того, чтобы указать, что коннектор имеет подписку на определенные события. При вызове обработчика через InvokeMethod указанные имена обработчиков передаются в InvokeMethod.

Проверить получение номенклатуры можно с помощью клиента Mobile SMARTS для ПК. Для получения списка товаров заходим в «Просмотр справочников -> Товары». Проверить получение по штрихкоду можно, например, в операции «Сбор штрихкодов»:



Реализуем получение списка документов. Обработчик события «Получить список документов» вызывается, когда пользователь на ТСД заходит в список выбора документов по кнопке типа документа.

```
[C#]

[DocumentEventProcessor(EventType = EventType.ListDocuments)]
public override object GetDocumentsList(DeviceInfo di, string documentTypeName)
{
...
}
```

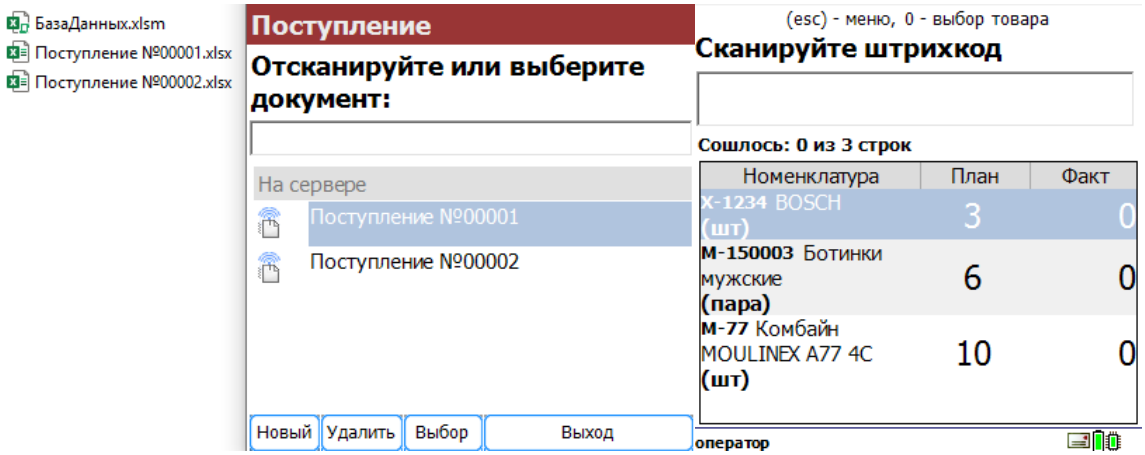
Функция возвращает DocumentDescriptionCollection со списком описаний документов (DocumentDescription), готовых для отдачи на ТСД. Список получаем на основе имен файлов xlsx в папке с данными.

Для получения документа в работу на ТСД нужно реализовать обработчик события «Получить документ». Обработчик вызывается, когда пользователь на ТСД выбирает документ из списка, полученного с помощью события «Получить список документов», или сканирует штрихкод документа в окне выбора документов.

```
[C#]

[DocumentEventProcessor(EventType = EventType.GetDocument)]
public override object GetDocument(DeviceInfo di, string documentTypeName, string identity,
GetDocumentMode mode)
{
...
}
```

Функция возвращает объект Document, если документ найден по переданным параметрам.

Проверим работу на примере документов «Поступление»:



Для загрузки завершенного на ТСД документа используем событие «Документ завершен».

```
[C#]

[DocumentEventProcessor(EventType = EventType.DocumentFinished)]
public override object DocumentFinished(DeviceInfo di, string documentTypeName, Document doc)
{
CheckInit();
string fileName = GetFileNameByDocId(doc.Id, documentTypeName);
if (!string.IsNullOrEmpty(fileName))
{
if(LoadDocumentFromTerminal(fileName, doc))
{
MessageCenter.AddNewMessage(string.Format("Документ '{0}' загружен.", doc.Name), null,
doc.UserId, false);
BaseRuntimeContext.Current.DocumentsManager.Delete(doc);
return doc;
}
}
return base.DocumentFinished(di, documentTypeName, doc);
}
```

В данной функции после загрузки документа на ТСД отправляется сообщение с помощью MessageCenter.AddNewMessage. Завершенный документ удаляется из базы Mobile SMARTS с помощью BaseRuntimeContext.Current.DocumentsManager.Delete (doc).

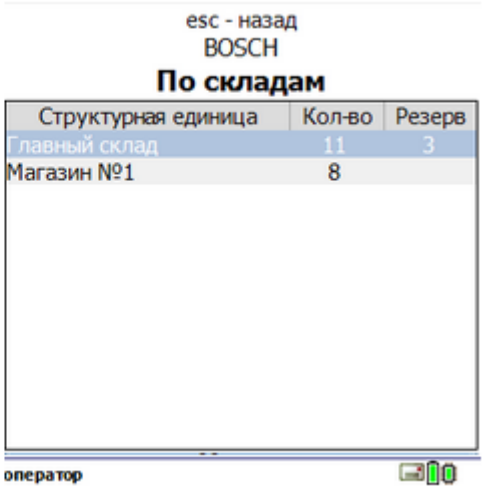
Реализуем получение данных онлайн при запросах к таблицам Mobile SMARTS. Например, в конфигурации «Магазин 15» есть таблицы «Склады», «Остатки» и др. На ТСД может выполняться получение списка всех складов, запрос остатков определенного товар и др. Если таблица хранится на сервере, в настройках таблицы включен поиск во внешней системе и задан обработчик события «Обработать запрос», то сервер вызывает указанный обработчик. Перегрузим в нашем коннекторе функцию ProcessTableRequest.

```
[C#]

[EventProcessor(EventType = EventType.TableRequest)]
public override object ProcessTableRequest(DeviceInfo di, DocumentQuery dq, DocumentTableInfo
tableInfo)
{
...
}
```

В функцию передается: DocumentQuery dq — объект запроса, в свойстве WhereRootElement содержится корень синтаксического дерева запроса. Обойдя дерево, можно сформировать запрос в том виде, который ожидает внешняя система. DocumentTableInfo tableInfo — описание таблицы, в свойстве Name содержится имя таблицы из конфигурации Mobile SMARTS. В простейшем случае можно возвращать все строки запрошенной таблицы, не накладывая условие из DocumentQuery, сервер Mobile SMARTS сам выполнит выборку данных по заданным условиям. Однако, в случае реальной работы с большими объемами данных, так делать не рекомендуется.

Функция возвращает коллекцию строк RowCollection.

Проверим получение складов и остатков:



Кроме обработки событий сервера, есть возможность вызова произвольных функций внешней системы, которые будут возвращать некоторые данные на ТСД или выполнять какую-то работу во внешней системе. В конфигурации Mobile SMARTS для этого используется действие «Вызов внешней системы». В коннекторе должна быть реализована функция InvokeMethod.

В нашем случае InvokeMethod будет вызывать макрос Excel с указанным именем, передавая полученные с ТСД аргументы:

[C#]

```
public override object InvokeMethod(string methodName, object[] args)
{
CheckInit();
string dbPath = Path.Combine(DatabaseFolder, DbFileName);
Excel.Workbook workbook = this.excel.Workbooks.Open(dbPath);
try
{
return RunMacro(methodName, args);
}
finally
{
workbook.Close();
}
}
```

Добавим в книгу Excel лист «Работы», на котором будет таблица с количеством собранных сотрудниками заказов за неделю:

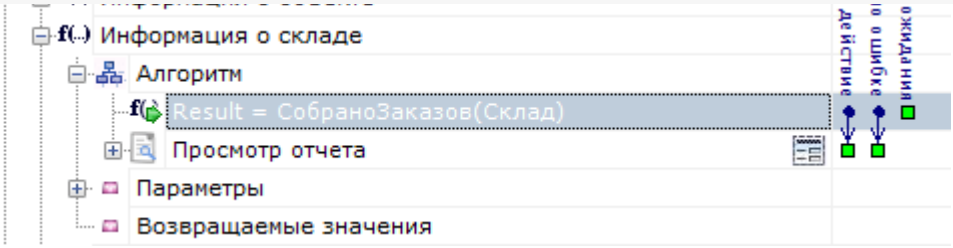
	A	B	C
1	Комплектовщик	Склад	Собрано заказов
2	Фахуртдинов	Главный склад	352
3	Петров	Главный склад	278
4	Коршунов	Магазин №1	94
5	Мухамедов	Магазин №2	65

На VB в Excel напишем функцию, которая будет возвращать таблицу с колонками «Комплектовщик», «Собрано заказов» по переданному складу.

[C#]

```
Function СобраноЗаказов(Склад As String)
Dim sh_src As Worksheet
Dim storageConnector As Object, rowCollection As Object, row As Object
Dim picker As String
Dim ordCnt As Integer
Dim rng As Excel.Range
Set storageConnector = CreateObject("Cleverence.Warehouse.StorageConnector")
Set rowCollection = CreateObject("Cleverence.Warehouse.RowCollection")
Set sh_src = Worksheets("Работы")
.....
СобраноЗаказов = storageConnector.ToXml(rowCollection)
End Function
```

Функция возвращает объект Cleverence.Warehouse.RowCollection, сериализованный в xml. В конфигурации Mobile SMARTS сделаем вызов данной функции при просмотре информации о складе:



В действии «Просмотр отчета» выведем полученную таблицу:

Не нашли что искали?

Задать вопрос в техническую поддержку

Тестирование и выпуск разработанного коннектора к внешней системе

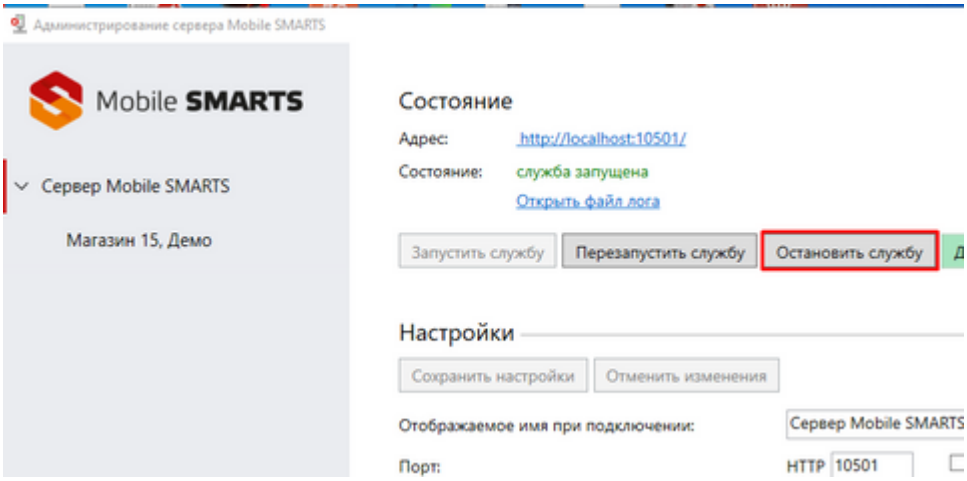
Последние изменения: 2024-03-26

В связи с повышением требований к безопасности сервера все разработанные коннекторы должны пройти процедуру проверки и подписания у «Клеверенс».

Попытка использования неподписанного коннектора в «продуктивном» режиме приводят к полной остановке сервера с записью в лог файле «Коннекторы не загружены! Обнаружена неподписанная сборка: xxxx.dll», либо «Коннекторы не загружены! Ошибка проверки подписи dll: xxxx.dll».

Для отладки коннектора при разработке следует использовать запуск сервера в тестовом режиме.

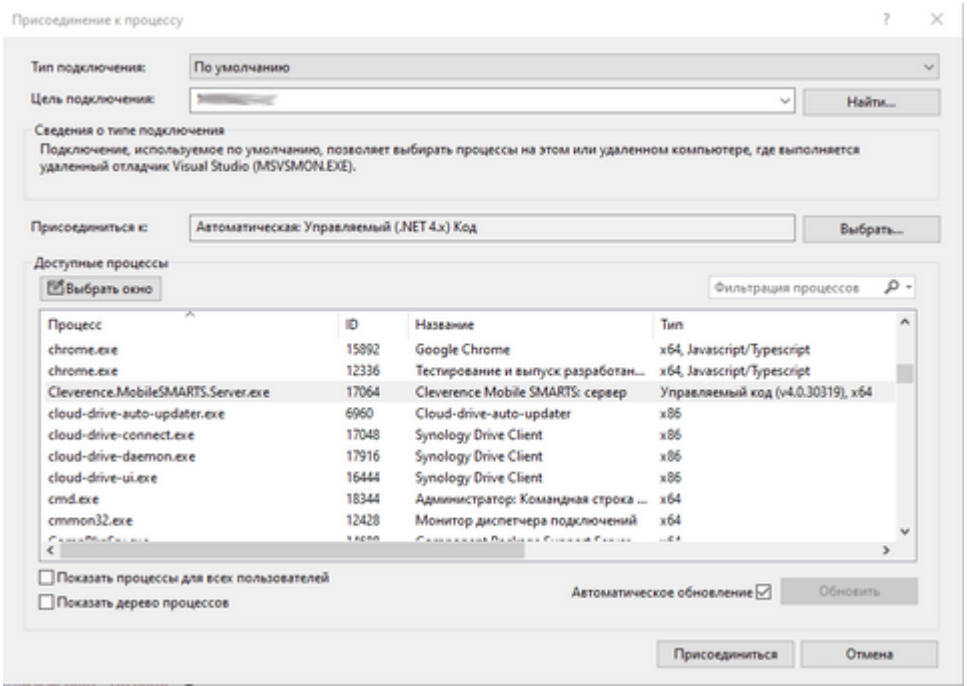
Для этого остановите сервис сервера:



И запускайте его из командной строки с параметром /debug:

c:\Program Files (x86)\Cleverence Soft\Mobile SMARTS\Server\Cleverence.MobileSMARTS.Server.exe /debug

Далее Вы можете просто подключаться отладчиком Visual Studio к запущенному процессу и вести отладку вашего коннектора.



После завершения разработки вашего коннектора создайте запрос в техническую поддержку «Клеверенс» о его проверке и подписании.

Не нашли что искали?

Задать вопрос в техническую поддержку