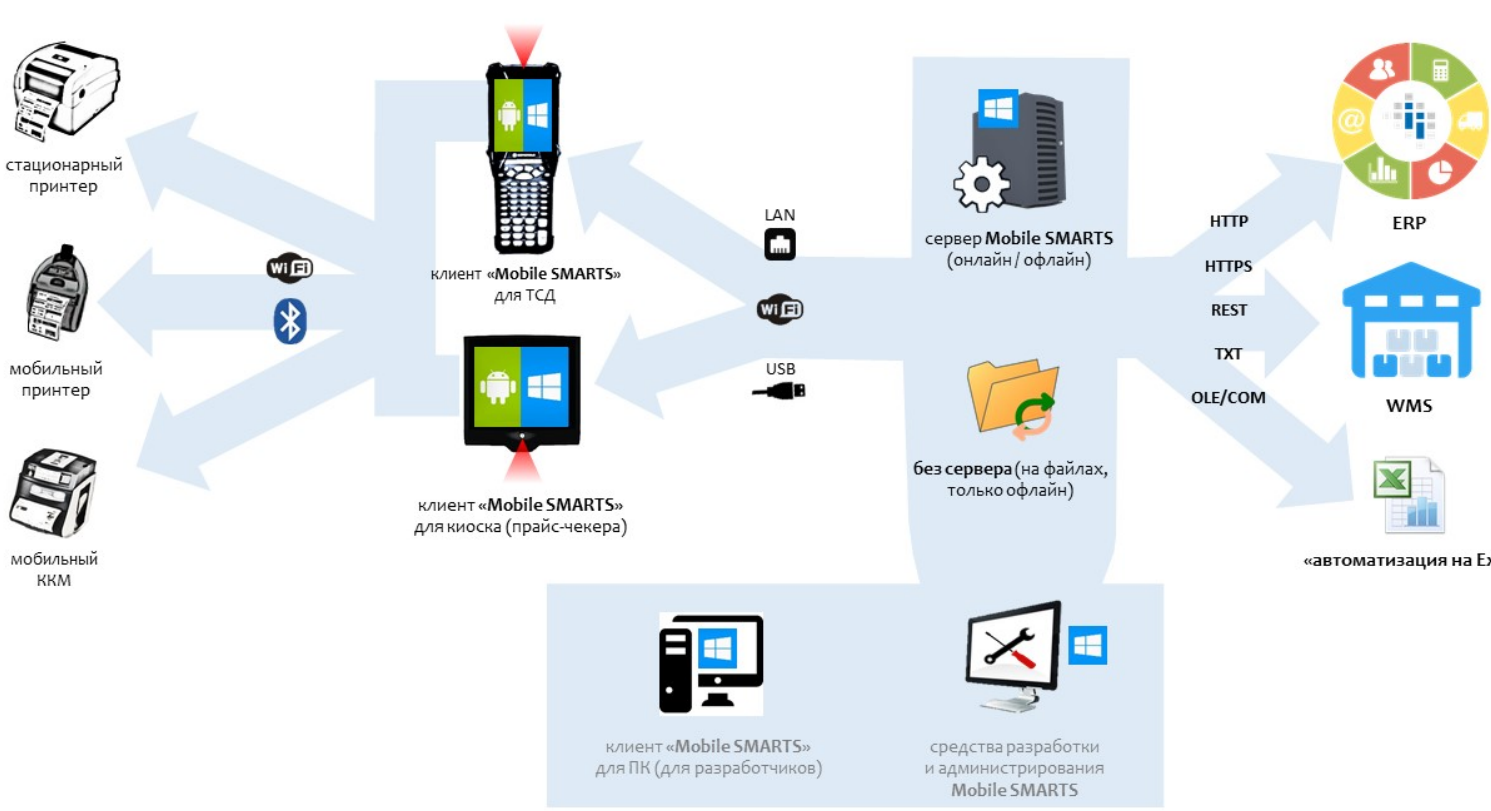
Схема взаимодействия ТСД с сервером Mobile SMARTS и сторонними учетными системами

Последние изменения: 2024-03-26



Данная схема взаимодействия ТСД с учетной системой позволяет производить обмен следующими способами:

1. либо через сервер Mobile SMARTS,
2. либо через с файлами через специальную программу, где ТСД может быть подключен к базе Mobile SMARTS через Wi-Fi, USB, или сетевой кабель.

ТСД может работать на ОС Android, WinCE, Windows Mobile, помимо подключения к серверу может взаимодействовать с кассами, мобильными принтерами через bluetooth или Wi-Fi.

Сервер Mobile SMARTS позволяет через разные типы коннекторов (в том числе и программируемые) обмениваться данными практически с любой учетной системой.

Учетные системы могут самостоятельно обращаться к серверу Mobile SMARTS через OLE/COM компоненту.

интеграция

Не нашли что искали?

Задать вопрос в техническую поддержку

Приложения и базы Mobile SMARTS

Последние изменения: 2024-03-26

Приложение Mobile SMARTS (см. App [ПриложениеСМАРТС]) представляет собой некоторое прикладное решение на платформе Mobile SMARTS ("Магазин 15", "Учет имущества", и т.д.).

Каждое приложение устанавливается из отдельного дистрибутива (если при этом платформа Mobile SMARTS на компьютере не установлена, сначала будет предложено загрузить и установить платформу). Например, с этой страницы можно скачать продукт Mobile SMARTS для ЕГАИС, предназначенный для учета алкогольной продукции при помощи терминалов сбора данных.

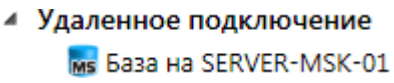
После установки приложения появляется возможность создавать базы данных на основе шаблона установленного приложения. См. подробнее.

База данных Mobile SMARTS (см. AppInstance [БазаСМАРТС]) - это экземпляр определенного приложения. База данных хранит конфигурацию Mobile SMARTS и текущие данные (справочники, документы, настройки). База данных находится в определенной папке на диске компьютера. В зависимости от того, какой способ обмена данными с терминалами требуется, база данных может работать в следующих режимах: обмен с сервером Mobile SMARTS, прямое подключение терминала к компьютеру через провод, обмен через папку на диске (Режим RDP - общая папка, терминал подключается физически к другому компьютеру, обмен непосредственно с терминалом выполняет специальная утилита, установленная на компьютере, к которому подключается терминал).

См. подробнее.

После установки Mobile SMARTS пользователь имеет возможность работать с базами данных (добавлять базы в список зарегистрированных баз, удалять, изменять настройки) с помощью Менеджера баз. Обмен данными возможен только с базами, зарегистрированными в списке баз на данном компьютере.

Удаленная база. Существует возможность добавить в список (зарегистрировать) не только базу, находящуюся на данном компьютере, но и работающую на сервере Mobile SMARTS, расположенном на другом компьютере в сети. В менеджере баз такая база будет находится в разделе Удаленное подключение:



На локальном компьютере удаленная база также, как и локальная, будет находится в заданной при добавлении базы папке, но в этой папке будут содержаться только настройки соединения с удаленным сервером Mobile SMARTS и пользовательские настройки, все данные (справочники, документы) при обмене будут загружаться (и выгружаться) с удаленного сервера. Сервер Mobile SMARTS представляет собой web-сервис, при обмене данными компонента вызывает функции web-сервиса, данные передаются по протоколу http. Организовывать общий доступ к папке базы на удаленном компьютере не нужно.

Объектам, служащим описанием удаленной базы, является также AppInstance [БазаСМАРТС] и для доступа к данным используется StorageConnector [Соединение]. При использовании функций обмена нет никакой разницы, удаленная база или локальная.

Создание и настройка новой базы данных

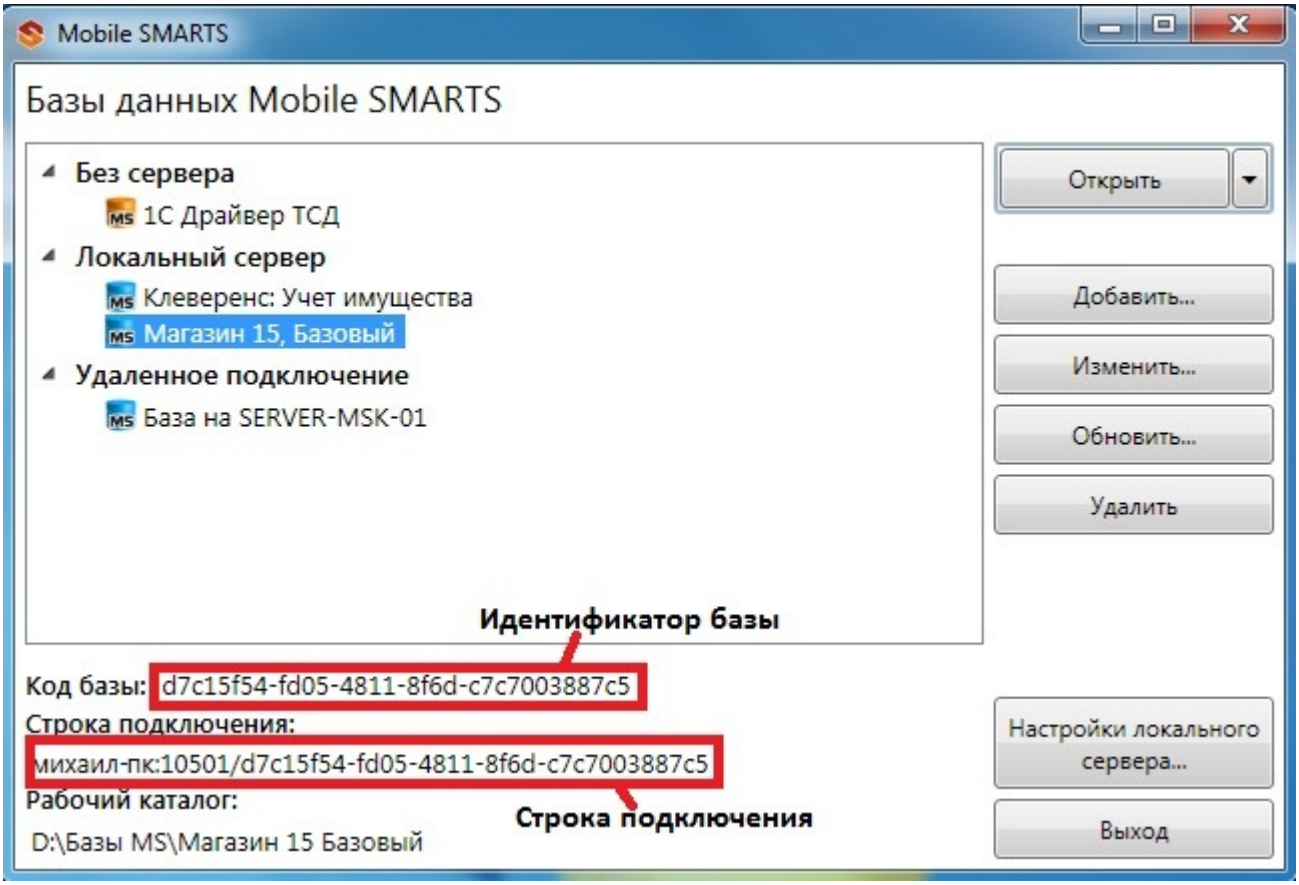
См. Заведение базы данных с помощью менеджера баз Mobile SMARTS

Для созданной базы данных пользователь указывает нужный режим работы (Прямое подключение к устройству, Прямая работа с папкой (RDP режим), Подключение к серверу). См. Настройка базы данных Mobile SMARTS.

Подключение к базе данных из учетной системы

Для того, чтобы была возможность производить обмен данными с базой Mobile SMARTS, необходимо сначала выполнить подключение к базе. Нам потребуется создать экземпляр COM-объекта Cleverence.Warehouse.StorageConnector и вызвать функцию SelectCurrentApp

[УстановитьПодключениеСБазойСМАРТС], передав в качестве аргумента Идентификатор (Код) базы или Строку подключения к базе.



Идентификатор и код в менеджере баз:

C#:

```
var connector  = new Cleverence.Warehouse.StorageConnector(); // Создаем экземпляр объекта
try

{

// Выполняем подключение

connector.SelectCurrentApp("михаил-пк:10501/d7c15f54-fd05-4811-8f6d-c7c7003887c5");

}

catch

{

// При подключении возникла  ошибка,  обрабатываем исключение

}
```

«1С:Предприятие 8»:

```
connector  =  новый  COMОбъект("Cleverence.Warehouse.StorageConnector");  //  Создаем
экземпляр  объекта Попытка

 connector.УстановитьПодключениеСБазойСМАРТС("михаил-пк:10501/d7c15f54-fd05-4811-8f6d-
c7c7003887c5");

 //Выполняем  подключение Исключение

 // При  подключении  возникла ошибка, обрабатываем  исключение КонецПопытки;
```

Получение списка баз, зарегистрированных на компьютере

Получить список баз может быть нужно, например, для того, чтобы дать пользователю учетной системы возможность выбирать базу, с которой будет выполняться обмен, из списка баз и хранить это как настройку в учетной системе, вместо того, чтобы жестко прописывать в коде строку подключения.

Воспользуемся для этого функцией GetAppsList [ПолучитьСписокБаз]. Перед вызовом этой функции выполнять подключение к какой-либо базе не нужно. Функция возвращает как локальные, так и удаленные базы, зарегистрированные на данном компьютере.

C#:

```
 var connector  = new Cleverence.Warehouse.StorageConnector(); // Создаем экземпляр объекта

 try

 {

 var appInstanceList = connector.GetAppsList(""); //Получаем список баз  - объект
AppInstanceCollection for(int i=0; i<appInstanceList.Count; i++) // Обходим в цикле полученный
список

 {

 var appInstance = appInstanceList[i]; //Получаем элемент списка - объект AppInstance var id =
appInstance.Id; //Идентификатор базы

 var connectionString = appInstance.ConnectionString; //Строка подключения var name  =
appInstance.Name; //Имя базы

   // Обрабатываем полученнные значения

 }

 }

 catch

 {

 // Обработка исключения



 }
```

«1С:Предприятие 8»:

```
connector = новый COMОбъект("Cleverence.Warehouse.StorageConnector"); // Создаем экземпляр объекта Попытка

СписокБаз = connector.ПолучитьСписокБаз(“”); //Получаем список баз - объект AppInstanceCollection Для Инд=0 По СписокБаз.Количество - 1 Цикл // Обходим в цикле полученный список

БазаСмартса = СписокБаз.Item(Инд); //Получаем элемент списка - объект AppInstance Ид = БазаСмартса.Ид; //Идентификатор базы

СтрокаПодключения = БазаСмартса.СтрокаПодключения; //Строка подключения Имя = БазаСмартса.Имя; //Имя базы

... // Обрабатываем полученнные значения КонецЦикла;

Исключение



// Обработка исключения КонецПопытки;
```

Получение и сохранение произвольных настроек

Есть возможность сохранять произвольные настройки в базе данных Mobile SMARTS из учетной системы. Например, это могут быть какие-либо настройки, используемые при выгрузке номенклатуры, параметры отборов документов и др. Настройки сохраняются в виде “Ключ-Значение”, Ключ - строка, задающее имя настройки, Значение - данные простого типа (строка, число, булево). Для работы с настройками необходимо получить объект базы AppInstance [БазаСМАРТС] и использовать функции

GetSettings [ПолучитьНастройки] и SaveSettings [СохранитьНастройки]. Объектом настроек является

AppInstanceSettings [НастройкиБазыСМАРТС].

C#:

```
 var connector  = new Cleverence.Warehouse.StorageConnector(); // Создаем экземпляр объекта try

 {

 // Получаем базу по строке подключения или идентификатору

 var appInstance  = connector.GetAppById("михаил-пк:10501/d7c15f54-fd05-4811-8f6d-c7c7003887c5");

   var appInstanceSettings = appInstance.GetSettings(); //Получаем объект настроек AppInstanceSettings var settingNode  = appInstanceSettings.GetSetting("ИмяНастройки"); //Получаем узел настройки по имени

 var value = settingNode.Value; //Значение настройки   settingNode.Value = newValue; // Присваиваем новое значение настройке

   appInstance.SaveSettings(appInstanceSettings); // Сохраняем настройки в базу

 }

 catch

 {

 // Обработка исключения

 }
```

«1С:Предприятие 8»:

```
 connector  =  новый  COMОбъект("Cleverence.Warehouse.StorageConnector");  //  Создаем экземпляр  объекта Попытка

 БазаСМАРТС =  connector.ПолучитьБазуСМАРТСПоИД("михаил-пк:10501/d7c15f54-fd05-4811-8f6d-c7c7003887c5");

 //Получаем базу по строке подключения  или  идентификатору

 НастройкиБазы  = БазаСМАРТС.ПолучитьНастройки(); //Получаем объект  настроек AppInstanceSettings УзелНастройки  = НастройкиБазы.Настройка("ИмяНастройки"); //Получаем узел настройки по  имени

 ЗначениеНастройки  = УзелНастройки.Значение; //Значение настройки УзелНастройки.Значение = НовоеЗначение; // Присваиваем новое  значение  настройке

   БазаСМАРТС.СохранитьНастройки(НастройкиБазы);  //  Сохраняем  настройки  в  базу Исключение

 // Обработка исключения КонецПопытки;
```

Получение режима работы базы

Рассмотрим, как получить, в каком режиме работает база данных (Прямое подключение к устройству, Прямая работа с папкой (RDP режим) или Подключение к серверу, см. Настройка базы данных Mobile SMARTS). Нам нужно будет получить объект базы AppInstance [БазаСМАРТС], получить настройки базы с помощью функции GetSettings [ПолучитьНастройки] и использовать следующие свойства объекта настроек AppInstanceSettings [НастройкиБазыСМАРТС]:

Свойство
Тип
Описание
ServerMode [РаботаССервером]
Булево
Режим работы с сервером. Истина - база работает в режиме с сервером, Ложь - без сервера.
LocalServerMode [РаботаСЛокальнымСервером]
Булево
Возвращает признак, работает ли серверная база на локальном компьютере. Истина - серверная база на локальном компьютере, Ложь - удаленная серверная база или база не серверная (прямой обмен с ТСД или с папкой).
RemoteServerMode
Булево
Возвращает признак того, что база работает на
[РаботаСУдаленнымСервером]
удаленном сервере. Истина - серверная база на удаленном компьютере, Ложь - локальная серверная база или база не серверная (прямой обмен с ТСД или с папкой).

FolderMode [РаботаСКаталогом]
Булево
Возвращает признак того, что база работает в режиме "с каталогом". Истина - база работает работает в режиме "с каталогом". Режим RDP - обмен данными через общую папку, терминал подключается физически к другому компьютеру, обмен непосредственно с терминалом выполняет специальная утилита, установленная на компьютере, к которому подключается терминал.
DeviceMode [РаботаСУстройствомНапрямую]
Булево
Возвращает признак того, что база работает в режиме напрямую с терминалом (батч-режим). Истина - база работает в режиме напрямую с терминалом (батч-режим). Обмен с терминалом происходит через проводное подключение.

C#:

```
 var connector  = new Cleverence.Warehouse.StorageConnector(); // Создаем экземпляр объекта соединения

 try

 {

 // Получаем базу по строке подключения или идентификатору

 var appInstance  = connector.GetAppById("михаил-пк:10501/d7c15f54-fd05-4811-8f6d-c7c7003887c5");

      var appInstanceSettings = appInstance.GetSettings(); //Получаем объект настроек AppInstanceSettings if(appInstanceSettings.ServerMode == true)

 {

 //Работа с сервером if(appInstanceSettings.LocalServerMode == true)

 {

 // Локальный сервер

 }

 else

 {
```

```
{
// Удаленный сервер
}
}
else  if(appInstanceSettings.FolderMode == true)
{
// Работа с папкой
}
else  if(appInstanceSettings.DeviceMode == true)
{
// Работа с ТСД напрямую
}
}
catch
{
// Обработка исключения
}
```

«1С:Предприятие 8»:

```
 connector = новый COMОбъект("Cleverence.Warehouse.StorageConnector"); // Создаем экземпляр объекта

  //соединения Попытка

  БазаСМАРТС = connector.ПолучитьБазуСМАРТСПоИД("михаил-пк:10501/d7c15f54-fd05-4811-8f6d-c7c7003887c5");

  //Получаем базу по строке подключения или идентификатору

    НастройкиБазы = БазаСМАРТС.ПолучитьНастройки(); //Получаем объект настроек AppInstanceSettings Если НастройкиБазы.РаботаССервером = Истина Тогда //Работа с сервером

  Если НастройкиБазы.РаботаСЛокальнымСервером = Истина Тогда // Локальный сервер

  ...

  Иначе // Удаленный сервер

  ...

  КонецЕсли;

  ИначеЕсли НастройкиБазы.РаботаСКаталогом = Истина Тогда //Работа с папкой

  ...

  ИначеЕсли НастройкиБазы.РаботаСУстройствомНапрямую = Истина Тогда //Работа с ТСД напрямую

  ...

  КонецЕсли;

    Исключение

  // Обработка исключения КонецПопытки;
```

COM-компонента

Не нашли что искали?

Задать вопрос в техническую поддержку

Выгрузка номенклатуры на ТСД через COM-компоненту

Последние изменения: 2024-03-26

Выгрузка справочника номенклатуры является в большинстве случаев первоочередной задачей при интеграции Mobile SMARTS с внешней учетной системой.

Выгрузка необходима, чтобы иметь возможность на терминале идентифицировать товар (находить по сканированному штрихкоду, выбирать из списка, находить по наименованию). Если просто сканировать штрихкоды без идентификации товара, то у оператора не будет возможности прямо на месте узнать: известен ли этот штрихкод учетной системе? верно ли заведена карточка товара? верная ли цена?

Сидя у компьютера будет очень трудно понять, к чему относятся ошибки загрузки результатов в учетную систему. Ошибки надо выдавать прямо на ТСД еще во время сканирования. Иначе на большом экране ПК будет список ошибок и будет написано “неизвестный штрихкод 1234567890”. А как понять, что это был за товар в торговом зале? Мы целые пол часа ходили сканировали собирали штрихкоды.

Кроме того, в каждой строке выгружаемых на терминал документов-заданий должен быть указан идентификатор товара и упаковки, которые обязаны соответствовать идентификаторам позиций выгруженного в базу Mobile SMARTS справочника номенклатуры.

В каждой учетной системе справочник номенклатуры имеет свою структуру. В общем случае из учетной системы может выгружаться не обязательно “номенклатура”, это могут быть “основные средства”, “инвентарные позиции”, “детали” и т.п. Любой большой список позиций, для которых требуется идентификация и быстрый поиск с занесением в строку документа. Перед написанием кода для выгрузки номенклатуры из учетной системы в Mobile SMARTS необходимо решить, как отобразится структура данных из учетной системы на объекты Mobile SMARTS.

Для «1С:Предприятия» выгрузку номенклатуры лучше всего проводить через методы TerminalConnector, которые не только принимают таблицы значений, но и поддерживают самые последние стандарты 1С для драйверов торгового оборудования.

Для остальных систем необходимо работать с ComConnector и объектами номенклатуры, которые в нем представлены.

Объектом, представляющим товар в Mobile SMARTS является Product [Товар], каждый товар содержит коллекцию упаковок Packing [Упаковка] (у товара должна быть хотя бы одна упаковка). У товара должна быть установлена базовая упаковка при помощи свойства BasePackingId [ИдБазовойУпаковки]. Базовая упаковка должна содержатся в коллекции упаковок Packings [Упаковки]. Базовая упаковка обычно имеет коэффициент (количество единиц UnitsQuantity [КоличествоБазовыхЕдиниц]) равный 1. Для остальных упаковок (например, коробки, палеты) UnitsQuantity [КоличествоБазовыхЕдиниц] определяет, сколько базовых упаковок содержится в данной (например, в коробке 10 шт, в палете 500 шт).

Если товар весовой и упаковок в действительности нет, при выгрузке в Mobile SMARTS должна быть создана одна упаковка (например, Килограммы (кг)) и она же назначена базовой.

Если различать упаковки товара нет необходимости (в учетной системе есть только сущность “Товар”), также нужно создать одну упаковку (например, “шт”) и назначить ее базовой.

В учетной системе могут быть различные сущности, связанные с товаром отношением “один ко многим” и идентифицируемые по штрихкоду, но не являющиеся упаковками. Например, серии и характеристики товара. Эти сущности в Mobile SMARTS также могут быть отражены как упаковки товара. (Примечание: можно использовать и дополнительные таблицы, которые определяются разработчиком конфигурации Mobile SMARTS, и связь с номенклатурой через ключ связи (Идентификатор товара), но при использовании упаковок уже будут работать все готовые механизмы по выбору номенклатуры, быстрому поиску по штрихкоду, записи в документ).

Объекты Product [Товар] и Packing [Упаковка] содержат как основные свойства, такие как Id [Ид], Name [Имя], Barcode [Штрихкод], Marking [Артикул], которые есть всегда в объектах данного типа, так и дополнительные, определяемые разработчиком конкретной конфигурации Mobile SMARTS. См. ниже Дополнительные поля.

Основные свойства и методы

Product [Товар]

Свойства
Наименование
Тип
Описание
Id [Ид]
Строка
Уникальный идентификатор товара.
Name [Имя]
Строка
Наименование товара.
Barcode [Штрихкод]
Строка
Основной штрихкод товара. Определяет только товар, без конкретной упаковки. Поиск при сканировании на терминале в первую очередь осуществляется по этому штрихкоду. Если товар найден и имеет несколько упаковок - то пользователю будет предложен выбор из списка, в какой упаковке сканирован штрихкод. Штрихкодов может быть задано несколько, с помощью разделителя '|'.
Marking [Артикул]
Строка
Общий артикул товара (без указания конкретной упаковки). Если необходимо, чтобы один и тот же товар в разных упаковках имел разные артикулы используйте такое же свойство у упаковки: Marking [Артикул].

BasePackingId **[ИдБазовойУпаковки]**
Строка
Идентификатор основного типа упаковки. Упаковка с данным идентификатором должна быть в коллекции упаковок товара Packings [Упаковки]. Подробнее см. Packing [Упаковка].
Packings [Упаковки]
PackingCollection **[КоллекцияУпаковок]**
Коллекция типов упаковок для товара. Подробнее см. Packing **[Упаковка].**
Методы
Наименование
Параметры и возвращаемое значение
Описание
GetField [ПолучитьПоле]
Параметр: string (имя поля) Возвращаемое значение: object (значение поля)
Возвращает значение дополнительного поля. Если поля с переданным именем в объекте нет, вернется null. **Пример:** **var value =** **product.GetField("ИмяПоля");** **значПоля =** **товар.ПолучитьПоле("ИмяПоля");**

SetField [УстановитьПоле]
Параметры: string (имя поля), object (значение) Возвращаемое значение: нет
Устанавливает значение дополнительного поля. Пример: product.SetField("ИмяПоля", value); товар.УстановитьПоле("ИмяПоля", значение);

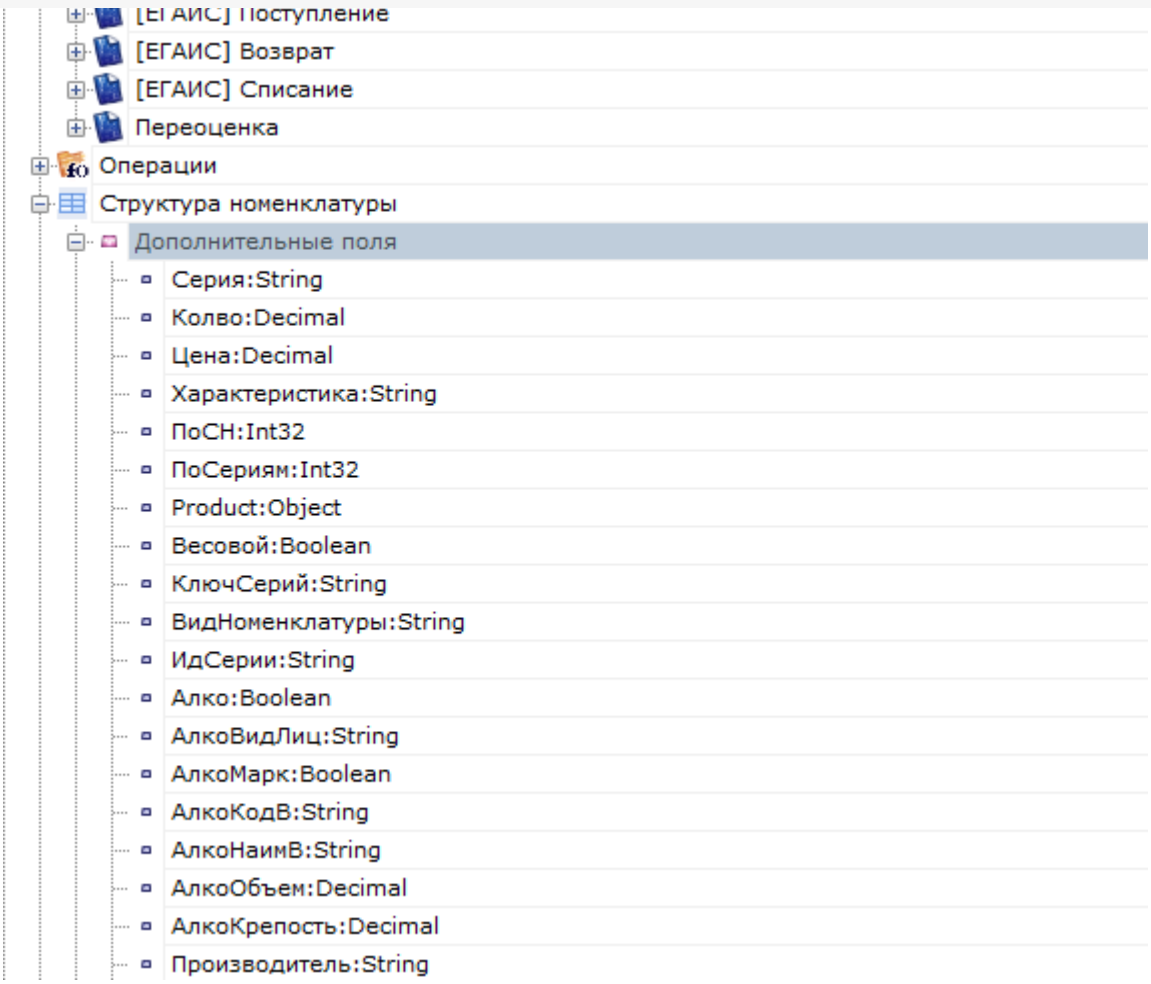
Packing [Упаковка]

Свойства
Наименование
Тип
Описание
Id [Ид]
Строка
Уникальный идентификатор типа упаковки. Уникальность в пределах упаковок одного товара.
Barcode [Штрихкод]
Строка
Штрихкод упаковки. Определяет товар в конкретной упаковке. При сканировании после поиска по основному штрихкоду товара, начинается поиск по этому штрихкоду. Если штрихкод найден, то выбирает товар сразу в конкретной упаковке. Кроме обычного задания в виде строки из цифр и букв, может задаваться в виде шаблона из комбинации символов и блоков {Template}. Такие блоки служат для вставки в штрихкод количества, срока годности и так далее. Подробнее про применение шаблонов штрихкодов см. тут. Штрихкодов и шаблонов может быть задано несколько, с помощью разделителя '|'.
Marking [Артикул]
Строка
Артикул товара в данной упаковке. Заполняется, если для упаковок товара используются артикулы, отличные от артикула товара.

Name [Имя]
Строка
Наименование упаковки.
UnitsQuantity [КоличествоБазовыхЕдиниц]
Число
Количество единиц товара в упаковке.
Методы
Наименование
Параметры и возвращаемое значение
Описание
GetField [ПолучитьПоле]
Параметр: string (имя поля) Возвращаемое значение: object (значение поля)
Возвращает значение дополнительного поля. Если поля с переданным именем в объекте нет, вернется null. Пример: var value = pack.GetField("ИмяПоля"); значПоля = упаковка.ПолучитьПоле("ИмяПоля")
SetField [УстановитьПоле]
Параметры: string (имя поля), object (значение) Возвращаемое значение: нет
Устанавливает значение дополнительного поля. Пример: pack.SetField("ИмяПоля", value); упаковка.УстановитьПоле("ИмяПоля", значение);

Дополнительные поля

При разработке конфигурации Mobile SMARTS могут быть определены произвольные дополнительные поля номенклатуры. Например, поле “Производитель”, в которое можно записывать при выгрузке номенклатуры наименование производителя товара или поле “Остаток”, в которое будет выгружаться количество данного товара на складе. Дополнительные поля задаются в Панеле управления Mobile SMARTS (см. Панель управления) в разделе Структура номенклатуры:



Заданные поля могут использоваться как для объектов Product [Товар], так и для Packing [Упаковка].

Для упаковки, например, могут выгружаться поля “Серия” и “Характеристика”, в которых будут наименование серии и характеристики товара (в случае, если упаковки определяют серии и характеристики товара).

Для выгрузки используются следующие функции объекта Cleverence.Warehouse.StorageConnector:

StorageConnector [Соединение]

Методы
Наименование
Параметры и возвращаемое значение
Описание

BeginUploadProducts

[НачатьПорционнуюВыгрузкуНом енклатуры]

Параметры: bool anyway,

bool overwriteExisting,

bool generateFullTextSearch

anyway

Начать новую выгрузку даже если какая-то другая выгрузка была открыта.

overwriteExisting

Флаг, задающий надо ли полностью заменить

номенклатуру в базе Mobile SMARTS или слить с существующей.

generateFullTextSearch

Флаг, задающий нужно ли генерировать индекс для полнотекстового поиска номенклатуры по наименованию.

Возвращаемое значение: нет

Функция открывает выгрузку номенклатуры в базу Mobile SMARTS.

UploadProducts

[ВыгрузитьПорциюНоменклатуры]

Параметры: ProductCollection products

products

Коллекция позиций номенклатуры, которые должны быть добавлены в текущую выгрузку.

Возвращаемое значение: нет

Выгружает номенклатуру в базу Mobile SMARTS в порционном режиме.

Учитывайте, что справочник номенклатуры реально изменится ТОЛЬКО после вызова EndUploadProducts [ЗавершитьПорционнуюВыгрузку Номенклатуры].

EndUploadProducts [ЗавершитьПорционнуюВыгрузку Номенклатуры]
Параметры: нет Возвращаемое значение: нет
Завершение порционной выгрузки номенклатуры. После этого вызова вся накопленная с помощью вызовов UploadProducts[ВыгрузитьПорци юНоменклатуры] номенклатура будет сохранена в базе Mobile SMARTS.
ResetUploadProducts
Параметры: нет
Сброс режима порционной
[СброситьПорционнуюВыгрузкуНоменклатуры]
Возвращаемое значение: нет
выгрузки. Имеет смысл использовать для серверной базы Mobile SMARTS. Все выгруженные, но не подтвержденные завершающим вызовом EndUploadProducts [ЗавершитьПорционнуюВыгрузкуНоменклатуры] товары не сохранятся в базе Mobile SMARTS на сервере. Если база не серверная, функция ничего не делает. Функцию нужно использовать, если выгрузка началась с BeginUploadProducts [НачатьПорционнуюВыгрузкуНоменклатуры] и далее в процессе выгрузки возникли ошибки. Функция сообщает серверу, что выгрузку нужно прервать (чтобы можно было впоследствии начать следующую выгрузку). Для несерверной базы прерывание выгрузки не требуется, данные в любом случае не сохранятся без вызова EndUploadProducts [ЗавершитьПорционнуюВыгрузкуНоменклатуры].

Выгрузка называется “порционной”, т.к. нет необходимости формировать весь список выгружаемых товаров сразу и передавать его в выгрузку целиком. Можно набрать некоторое количество товаров в коллекцию ProductCollection products, выполнить вызов UploadProducts [ВыгрузитьПорциюНоменклатуры], набрать еще и т.д. до обхода всех товаров, которые нужно выгрузить.

Общий алгоритм выгрузки номенклатуры. В общем виде алгоритм выгрузки выглядит следующим образом:

1. Создаем объект Cleverence.Warehouse.StorageConnector и устанавливаем подключение к нужной базе Mobile SMARTS при помощи вызова SelectCurrentApp

[УстановитьПодключениеСБазойСМАРТС];

2. Начинаем выгрузку с помощью вызова BeginUploadProducts

[НачатьПорционнуюВыгрузкуНоменклатуры];

3. Создаем объект ProductCollection [КоллекцияТоваров];

4. Получаем в учетной системе выборку записей, содержащую нужные поля для заполнения объектов

Product [Товар] и Packing [Упаковка];

5. Обходим в цикле выборку записей, создаем объекты Product [Товар] и Packing [Упаковка], упаковки добавляем в коллекцию упаковок Packings [Упаковки] объекта Product, обязательно указываем базовую упаковку (BasePackingId [ИдБазовойУпаковки]), добавляем объекты Product [Товар] в коллекцию товаров productCollection;

6. Если в коллекции товаров набралось достаточно много записей (например, 1000), вызываем выгрузку порции товаров UploadProducts [ВыгрузитьПорциюНоменклатуры]. Создаем новый объект

ProductCollection [КоллекцияТоваров], в который на следующих итерациях цикла будем накапливать товары;

7. После завершения цикла, если в коллекции товаров есть записи, вызываем UploadProducts [ВыгрузитьПорциюНоменклатуры];

8. Завершаем выгрузку, вызвав EndUploadProducts [ЗавершитьПорционнуюВыгрузкуНоменклатуры].

Пример

Допустим в результате некоторого отбора из различных таблиц в учетной системе мы получили выборку со следующими полями:

itemId (Идентификатор позиции номенклатуры)
name (наименование позиции номенклатуры)
marking (артикул)
unit (наименование единицы измерения, “шт”, “кг” и т.д.)
factor (коэффициент пересчета в базовые для данной единицы)
barcode (штрихкод)
descr (наименование характеристики номенклатуры)
qty (количество на складе)
price (цена)

В результате выборки получаем:

itemId
name
marking
unit
factor
barcode
descrl
qty
price

0001213
Джемпер
Д-0920
шт
1

0001213
Джемпер
Д-0920
шт
1
2500001780331
26/92, красный
5
359,00
0001213
Джемпер
Д-0920
шт
1
2500001780263
24/80, красный
7
359,00

0001213
Джемпер
Д-0920
шт
1
2500001780347
26/92, голубой
1
400,00

00000049
Бренди
шт
1
2200007761321
10
1200,00

00000049
Бренди
упак
6
2200007761321
7000,00

00000097
Крупа
Арт-8900
кг
1
10055
20
51,00

00000070
Печенье
Арт-4444
упак
1
2000020577788
8
62,00

00000070
Печенье
Арт-4444
упак
1
4607038271677
8
62,00

В выборке для одного и того же товара может быть несколько строк, т.к. может быть несколько характеристик, упаковок или штрихкодов одного товара.

Выгрузка:

C#:

```
 var connector  = new Cleverence.Warehouse.StorageConnector(); // Создаем экземпляр  объекта соединения

  // Выполняем подключение  к базе Mobile  SMARTS

  connector.SelectCurrentApp("михаил-пк:10501/d7c15f54-fd05-4811-8f6d-c7c7003887c5"); try

  {

  connector.BeginUploadProducts(true, true, true); //Начинаем выгрузку, в параметрах указываем, что

  //справочник перезаписываем и  создаем  индекс для  поиска по  наименованию

  var products = new Cleverence.Warehouse.ProductCollection(); //Создаем коллекцию для накопления

  //выгружаемых товаров

  SqlCommand command  = new SqlCommand();

  command.CommandText = query; //Запрос к  БД для  чтения данных

    using (SqlConnection connection  = new SqlConnection(connectionString)) //Устанавливаем соединение  с  //БД, из которой будем читать данные
```

```
 {

 connection.Open(); command.Connection = connection;

 SqlDataReader reader  = command.ExecuteReader(); while (reader.Read()) //Читаем данные

 {

 string itemId  = reader.GetString(0);
string name = reader.GetString(1);
string marking  = reader.GetString(2);
string unit = reader.GetString(3);
decimal factor  = reader.GetDecimal(4);
string barcode  = reader.GetString(5);
string descr = reader.GetString(6);
decimal qty = reader.GetDecimal(7);
decimal price  = reader.GetDecimal(8);

 //Получили все поля одной записи

 Cleverence.Warehouse.Product uploadProduct  = products.FindById(itemId); //Ищем товар по id
//среди набранных к выгрузке товаров

 if(uploadProduct == null)

 {

 //Если  товара  нет,  нужно  будет  создать  новый объект и  добавить  его  в коллекцию.

 //Перед этим проверяем, не набрали  ли  мы уже достаточное количество товаров.
if(products.Count >= 1000)

 {

 //Если  набрали  достаточно, выгружаем порцию товаров и  создаем  новый объект
коллекции,  чтобы

 //набирать  товары  дальше  в пустую  коллекцию. connector.UploadProducts(products);

 products = new Cleverence.Warehouse.ProductCollection();

 }

 uploadProduct  = new Cleverence.Warehouse.Product();

uploadProduct.Id = itemId; uploadProduct.Name = name; uploadProduct.Marking = marking;

 products.Add(uploadProduct);

 }

 }
```

```
Cleverence.Warehouse.Packing uploadPacking = null;

//Проверяем, если упаковка с той же единицей, характеристикой и ценой есть в упаковках данного

//товара, добавлять новую упаковку не нужно, нужно будет только добавить в эту же упаковку //штрихкод.
foreach(Cleverence.Warehouse.Packing pack in uploadProduct.Packings)

{

if(pack.Name == unit && pack.GetField("Характеристика") == descr && pack.GetField("Цена") == price)

{

uploadPacking = pack; break;

}

}

if(uploadPacking == null)

{

//Такой упаковки нет, создаем новую

uploadPacking = new Cleverence.Warehouse.Packing();

//Для упаковки нужно использовать уникальный идентификатор в пределах упаковок данного товара string packId = unit;

if(uploadProduct.Packings.FindById(unit) != null)

{

//Оказалось, что упаковка с Ид. равным единице измерения уже есть (могли быть разные

//характеристики товара с одной единицей).

//Генерируем новый Ид. вида [unit]_[номер] (например, "шт_1", "шт_2" и т.д.) int cnt = 0;

foreach (Packing p in uploadProduct.Packings) if (p.Id.StartsWith(unit + "_"))

cnt++;

packId = String.Format("{0}_{1}", unit, cnt + 1);

}
```

```
 uploadPacking.Id = packId;

 uploadPacking.Name = unit; //Имя равно единице измерения
uploadPacking.UnitsQuantity = factor; //Коэффициент пересчета в базовые единицы
uploadPacking.Barcode = barcode; //Штрихкод
uploadPacking.SetField("Характеристика", descr); //Устанавливаем доп. поля
uploadPacking.SetField("КоличествоНаСкладе", qty); uploadPacking.SetField("Цена", price);

   if(factor  == 1.0 && uploadProduct.BasePackingId == null)

 {

 //Если  коэффициент  равен  1 и базовую упаковку еще не установили, считаем  данную упаковку //базовой. uploadProduct.BasePackingId = packId;

 }

   uploadProduct.Packings.Add(uploadPacking); //Добавляем  упаковку в  коллекцию упаковок товара

 }

 else

 {

 //Если  такая упаковка уже есть, добавляем  к ней новый  штрихкод  (если он не пустой) if(!string.IsNullOrEmpty(barcode))

 uploadPacking.Barcode = uploadPacking.Barcode + "|"  + barcode;

}

  }

  if(products.Count > 0)

{

//Если  остались невыгруженные  товары,  их нужно  выгрузить connector.UploadProducts(products);

}

  connector.EndUploadProducts(); //Завершаем  выгрузку товаров.

}

catch

{
```

```
// При выгрузке возникла ошибка, обрабатываем исключение и сбрасываем начавшуюся выгрузку.

// В реальном приложении нужна отдельная обработка исключений, связанных с чтением данных из базы SQL // и с выгрузкой в Mobile SMARTS.

connector.ResetUploadProducts();

}
```

«1С:Предприятие 8»:

```
connector = новый COMОбъект("Cleverence.Warehouse.StorageConnector"); // Создаем экземпляр объекта

//соединения Попытка

connector.УстановитьПодключениеСБазойСМАРТС("михаил-пк:10501/d7c15f54-fd05-4811-8f6d-c7c7003887c5");

//Выполняем подключение

  Запрос = Новый Запрос(ТекстЗапроса);

ТаблицаТоваров = Запрос.Выполнить().Выгрузить(); //Выполняем запрос

  connector.BeginUploadProducts(Истина, Истина, Истина); //Начинаем выгрузку, в параметрах указываем, //что справочник перезаписываем и создаем индекс для поиска по наименованию

products = новый COMОбъект("Cleverence.Warehouse.ProductCollection"); //Создаем коллекцию для //накопления выгружаемых товаров

  Для Каждого СтрокаТаблицы из ТаблицаТоваров Цикл

ИдНоменклатуры = XMLСтрока(СтрокаТаблицы.НоменклатураСсылка); //В качестве идентификатора товара //будем выгружать Guid позиции номенклатуры

Наименование = СтрокаТаблицы.НоменклатураНаименование; Артикул = СтрокаТаблицы.Артикул;

ЕдиницаИзмерения = СтрокаТаблицы.УпаковкаНаименование; Коэффициент = СтрокаТаблицы.УпаковкаКоэффициент; Штрихкод = СтрокаТаблицы.Штрихкод;

Характеристика = СтрокаТаблицы.ХарактеристикаНаименование; Остаток = СтрокаТаблицы.ОстатокНаСкладе;

Цена = СтрокаТаблицы.Цена;

uploadProduct = products.НайтиПоИд(ИдНоменклатуры); //Ищем товар по id среди
```

```
  uploadProduct  = products.НайтиПоИд(ИдНоменклатуры); //ищем  товар по id  среди
набранных  к выгрузке

 //товаров

 Если uploadProduct  = Неопределено  Тогда

 //Если товара  нет, нужно будет создать новый объект и  добавить его в коллекцию.

 //Перед этим проверяем, не набрали  ли  мы уже достаточное количество товаров.

 Если products.Количество >= 1000 Тогда

 //Если  набрали  достаточно, выгружаем порцию товаров и  создаем  новый объект
коллекции,  чтобы

 //набирать  товары  дальше  в пустую  коллекцию.
connector.ВыгрузитьПорциюНоменклатуры(products);

 products = новый COMОбъект("Cleverence.Warehouse.ProductCollection"); КонецЕсли;



  uploadProduct  = новый COMОбъект("Cleverence.Warehouse.Product");



  uploadProduct.Ид = ИдНоменклатуры;

 uploadProduct.Имя = Наименование;

 uploadProduct.Артикул = Артикул;

  products.Добавить(uploadProduct); КонецЕсли;

    uploadPacking  = Неопределено;

  //Проверяем, если  упаковка с той же единицей, характеристикой  и ценой есть в упаковках
данного

  //товара,  добавлять новую упаковку  не  нужно, нужно будет только добавить  в эту  же
упаковку  //штрихкод.

  Для Инд=0  По uploadProduct.Упаковки.Количество-1  Цикл pack =
uploadProduct.Упаковки.Элемент(Инд);

  Если pack.Name = unit И pack.ПолучитьПоле("Характеристика") ==  Характеристика И
pack.ПолучитьПоле("Цена") == Цена Тогда

  uploadPacking = pack; Прервать;

  КонецЕсли; КонецЦикла;
```

```
    Если uploadPacking  = Неопределено  Тогда

  //Такой упаковки нет, создаем  новую

  uploadPacking  = новый COMОбъект(“Cleverence.Warehouse.Packing”);

  //Для  упаковки  нужно использовать  уникальный идентификатор  в пределах упаковок
данного  товара packId  = ЕдиницаИзмерения;

  Если uploadProduct.Упаковки.НайтиПоИд(ЕдиницаИзмерения) <>  Неопределено Тогда

  //Оказалось, что упаковка с Ид. равным  единице  измерения уже есть  (могли  быть  разные

  //характеристики товара с одной единицей).

  //Генерируем  новый  Ид.  вида  [ЕдиницаИзмерения]_[номер]  (например,  "шт_1",  "шт_2"  и
т.д.) Сч = 0;

  Для Инд=0 По uploadProduct.Упаковки.Количество-1  Цикл р =
uploadProduct.Упаковки.Элемент(Инд);

  Если (р.Ид.СтрНачинаетсяС(ЕдиницаИзмерения  + "_"))  Тогда Сч++;

  КонецЕсли; КонецЦикла;

  packId = ЕдиницаИзмерения + “_” +  (Сч  + 1); КонецЕсли;

    uploadPacking.Ид = packId;

  uploadPacking.Имя = ЕдиницаИзмерения; //Имя равно единице  измерения
 uploadPacking.КоличествоБазовыхЕдиниц = Коэффициент; //Коэффициент  пересчета в
базовые  единицы uploadPacking.Штрихкод = Штрихкод;

  uploadPacking.УстановитьПоле("Характеристика",  Характеристика); //Устанавливаем доп.
поля
 uploadPacking.УстановитьПоле("КоличествоНаСкладе", Остаток);
uploadPacking.УстановитьПоле("Цена", Цена);

    Если Коэффициент = 1.0 И uploadProduct.ИдБазовойУпаковки = Неопределено Тогда

  //Если  коэффициент  равен  1 и базовую упаковку еще не установили, считаем  данную
упаковку //базовой. uploadProduct.ИдБазовойУпаковки = packId;

  КонецЕсли;

    uploadProduct.Упаковки.Добавить(uploadPacking); //Добавляем  упаковку  в  коллекцию
упаковок  //товара Иначе

  //Если  такая упаковка уже есть, добавляем  к ней новый  штрихкод  (если он не пустой)

  Если ЗначениеЗаполнено(Штрихкод) Тогда
```

```
 uploadPacking.Штрихкод  = uploadPacking.Штрихкод + "|" +  Штрихкод; КонецЕсли;

 КонецЕсли; КонецЦикла;

   Если products.Количество > 0 Тогда

 //Если  остались невыгруженные  товары,  их нужно  выгрузить
connector.ВыгрузитьПорциюНоменклатуры(products);

 КонецЕсли;

  connector.EndUploadProducts (); //Завершаем выгрузку товаров.

  Исключение

// При  выгрузке возникли  ошибки, обрабатываем исключение
connector.СброситьПорционнуюВыгрузкуНоменклатуры();

КонецПопытки;
```

COM-компонента

Не нашли что искали?

Задать вопрос в техническую поддержку

Как происходит обмен документами между учетной системой и Mobile SMARTS через COM-компоненту

Последние изменения: 2024-03-26

При выполнении любых операций на терминале (Приемка, Отгрузка, Инвентаризация и др.) пользователь работает с некоторым документом Mobile SMARTS. Документ может быть выгружен из учетной системы и содержать список товаров, которые нужно отсканировать (или предполагается, что они будут отсканированы). Например, должна произойти приемка товара на склад и из учетной системы выгружается накладная со списком товаров, при этом по факту приемки могут быть расхождения (какой-то товар не привезли или привезли не в том количестве, а может придти товар, которого нет по накладной). Фактически отсканированный товар фиксируется в документе Mobile SMARTS. После завершения работы с документом на терминале, документ загружается в учетную систему и происходит регистрация принятого товара. Кроме выгрузки из учетной системы, документ может быть создан на терминале пользователем (если это позволяют настройки данного типа документа, заданные в конфигурации Mobile SMARTS). В таком документе будут только фактически отсканированные товары. Например, приемка на склад может производится по факту без выгрузки какого-либо документа из учетной системы. В этом случае при загрузке завершенного документа с терминала, как правило, создается новый документ учетной системы.

Таким образом, документ Mobile SMARTS служит заданием для выполнения на терминале (если он выгружается из учетной системы) и является результатом работы пользователя на терминале, который требуется отразить в учетной системе.

Алгоритм работы с документом каждого типа (какие данные должен вводить пользователь и как они будут обрабатываться), задается в Панели управления (см. Панель управления, Тип документа).

Документ Mobile SMARTS содержит два списка товаров (Плановая часть - товары, которые должны быть или предполагается отсканировать, Фактическая часть - фактически отсканированные товары), а также, если необходимо, другие данные (поля шапки, дополнительные табличные части).

Объектом документа является Document [Документ]. С помощью этого объекта происходит обмен данными между учетной системой и Mobile SMARTS (выгрузка документов-заданий и загрузка результатов работы с терминалов).

Document [Документ] содержит следующие основные поля и методы:

Свойства
Наименование
Тип
Описание
Свойства шапки документа, назначаемые при выгрузке из учетной системы:
Id [Ид]
Строка
Уникальный идентификатор документа. При выгрузке из учетной системы должен идентифицировать исходный выгружаемый документ. При создании документа на терминале назначается автоматически.

Name [Имя]
Строка
Название документа. Произвольное наименование, которое отображается на экране терминала. Назначается при выгрузке из учетной системы (например, “Приемка №327 от 12.07.16”), при создании документа на терминале присваивается автоматически.
DocumentTypeName [ИмяТипаДокумента]
Строка
Должно соответствовать имени одного из типов документов, имеющихся в конфигурации Mobile SMARTS.
WarehouseId [ИдСклада]
Строка
Идентификатор склада Mobile SMARTS, к которому привязан документ. При выгрузке из учетной системы должен быть равен Ид. одного из складов, имеющихся в конфигурации Mobile SMARTS. В типовых конфигурациях по умолчанию существует единственный склад “Общий” с Ид. равным “1”.
Appointment [Назначение]
Строка
Назначение документа - код пользователя или имя группы пользователей, которым данный документ назначается на исполнение. Если значение пустое, то документ попадает к первому свободному пользователю, которому разрешен тип документа. В типовых конфигурациях Mobile SMARTS по умолчанию заведен единственный пользователь с Ид. “оператор”.
AutoAppointed [ВыдаватьАвтоматически]
Булево
Флаг автоматической выдачи. Если стоит Истина - документ выдается на ТСД, автоматически, не дожидаясь явного выбора его пользователем из списка или по ШК. Имеет смысл только для серверной базы Mobile SMARTS. По умолчанию Истина. Выдача автоматически не означает, что документ будет сразу же открыт без участия пользователя. Сервер назначает документ для исполнения, он отправляется на терминал, пользователь получает звуковое уведомление, дальше документ нужно будет открыть, выбрав его из списка или по ШК.

Barcode [Штрихкод]
Строка
Штрихкод документа. Документ на терминале может быть открыт путем сканирования штрихкода с бумажной копии документа (если это разрешено настройками типа документа в конфигурации Mobile SMARTS).
ServerHosted [ИсполняемыйНаСервере]
Булево
Признак того, что документ должен выполняться "на сервере". Такой документ могут одновременно открыть на редактирование несколько пользователей. Все изменения в документе будут происходить одновременно для всех работающих с ним пользователей. Работа в таком режиме требует наличия постоянной связи с сервером. По умолчанию Ложь.
CreateDate [ДатаСоздания]
ДатаВремя
Дата создания документа. Если не назначать, проставится текущая дата.
Priority [Приоритет]
Целое
Приоритет документа. Более приоритетные документы раньше отдаются на терминал для обработки. Имеет смысл только для серверной базы Mobile SMARTS. Назначать не обязательно.

Description [Описание]
Строка
Описание документа, назначать не обязательно.
Свойства шапки документа, которые могут использоваться при загрузке в учетную систему:
Finished [Завершен]
Булево
Признак того, что обработка документа пользователем была завершена, и его можно забирать назад в учетную систему.
Modified [Изменен]
Булево
Признак того, что документ был изменен.

InProcess [ВОбработке]
Булево
Признак того, что документ захвачен пользователем на обработку.
Completed **[ВсеСтрокиПланаВыполнены]**
Булево
Свойство, позволяющее проверить все ли строки задания выполнены **(фактическое количество товара равно плановому). Проверка происходит по** **плановым строкам DeclaredItems.**
Overloaded [ЕстьПереполнение]
Булево
Признак, есть ли превышение количества товара в строках документа.
Underloaded [ЕстьНедобор]
Булево
Признак, есть ли недобор количества товара в строках документа.
CreatedOnPDA [СозданНаТСД]
Булево
Признак того, что документ был создан на ТСД. Истина - создан на ТСД, Ложь - выгружен из учетной системы.
UserId [ИдПользователя]
Строка
Ид. пользователя ТСД, который выполнил документ.
UserName [ИмяПользователя]
Строка
Имя пользователя ТСД, который выполнил документ.
DeviceId [ИдУстройства]
Строка
Ид. устройства (уникальный код терминала), на котором был завершен документ.

DeviceIP [ИпУстройства]
Строка
IP-адрес устройства, на котором был завершен документ.
DeviceName [ИмяУстройства]
Строка
Имя устройства, на котором был завершен документ.
Коллекции строк документа:
DeclaredItems [СтрокиПлан]
DocumentItemCollection [КоллекцияСтрокДокумента]
Коллекция строк документа DocumentItem [СтрокаДокумента], содержащая данные о товарах, которые предполагается отсканировать по плану. Заполняется при выгрузке из учетной системы. При этом на терминале при работе с документом также может происходить запись в плановые строки.
CurrentItems [СтрокиФакт]
DocumentItemCollection [КоллекцияСтрокДокумента]
Коллекция строк документа DocumentItem [СтрокаДокумента], содержащая данные о фактически отсканированных товарах. Заполняется при работе на терминале.
Tables [Таблицы]
DocumentTableCollection [КоллекцияТаблиц]
Коллекция дополнительных таблиц документа. В конфигурации Mobile SMARTS для типа документа могут быть определены дополнительные табличные части с произвольным набором полей. Дополнительные табличные части могут быть заполнены при выгрузке документа из учетной системы, также они могут заполняться при работе на терминале.
Методы

Наименование
Параметры и возвращаемое значение
Описание

GetField [ПолучитьПоле]
Параметр: string (имя поля) Возвращаемое значение: object (значение поля)
Возвращает значение дополнительного поля шапки документа. Если поля с переданным именем в объекте нет, вернется null. Пример: var value = document.GetField("ИмяПоля");   значПоля = документ.ПолучитьПоле("ИмяПоля")
SetField [УстановитьПоле]
Параметры: string (имя поля), object (значение) Возвращаемое значение: нет
Устанавливает значение дополнительного поля шапки документа. Пример: document.SetField("ИмяПоля", value); документ.УстановитьПоле("ИмяПоля", значение);

Объектом строки документа является DocumentItem [СтрокаДокумента]. Данные объекты содержатся в коллекции плановых (DeclaredItems [СтрокиПлан]) и фактических (CurrentItems [СтрокиФакт]) строк документа.

DocumentItem [СтрокаДокумента] содержит следующие основные поля и методы:

Свойства
Наименование
Тип
Описание

ProductId [ИдТовара]
Строка
Идентификатор товара. Товар с таким идентификатором должен быть в выгруженном в справочнике номенклатуры.

PackingId [ИдУпаковки]
Строка
Идентификатор упаковки товара. Упаковка с таким ид. должна быть в коллекции упаковок товара, заданного в строке документа.
DeclaredQuantity [КоличествоПлан]
Число
Плановое количество товара. Значение заполняется при выгрузке документа из учетной системы.
CurrentQuantity [КоличествоФакт]
Число
Фактическое количество товара. Заполняется при работе на терминале и отражает, сколько фактически было отсканировано данного товара.
SSCC [КодЕдиницыХранения]
Строка
Serial shipping container code - уникальный номер единицы хранения. Поле может использоваться для занесения номера контейнера, палеты и т.д. Может заполняться из штрихкода EAN-128 (см. Интерпретация кода EAN-128 в Mobile SMARTS).
FirstStorageBarcode [ШтрихкодПервогоМеста]
Строка
Штрихкод ячейки, палеты и т.п., в которых находится или из которых перемещается товар.
SecondStorageBarcode [ШтрихкодВторогоМеста]
Строка
Штрихкод ячейки, палеты и т.п. в которые перемещается товар из первого места хранения.
ExpiredDate [СрокГодности]
ДатаВремя
Срок годности товара. Может заполняться из штрихкода EAN-128 (см. Интерпретация кода EAN-128 в Mobile SMARTS).

RegistrationDate [ДатаРегистрации]
ДатаВремя
Дата регистрации товара. Может заполняться на терминале.
BindedLine [СвязаннаяСтрока]
DocumentItem [СтрокаДокумента]
Связанная строка документа. Заполняется на терминале и задает связь строки из фактической части со строкой из плановой части документа.
Overload [Переполнение]
Число
Возвращает превышение фактического количества товара над плановым (разницу между фактическим и плановым количеством).
Underload [Недобор]
Число
Возвращает разницу между плановым и фактическим количеством.
UnderloadedOrOverloaded [ЕстьНедоборИлиПереполнение]
Булево
Позволяет проверить совпадают ли заявленное и фактическое количество в строке. Истина - количества разные, Ложь - количества одинаковые.
Методы
Наименование
Параметры и возвращаемое значение
Описание

GetField [ПолучитьПоле]
Параметр: string (имя поля) Возвращаемое значение: object (значение поля)
Возвращает значение дополнительного поля шапки документа. Если поля с переданным именем в объекте нет, вернется null. Пример: var value = documentItem.GetField("ИмяПоля"); значПоля = строкаДокумента.ПолучитьПоле("ИмяПол я")
SetField [УстановитьПоле]
Параметры: string (имя поля), object (значение) Возвращаемое значение: нет
Устанавливает значение дополнительного поля шапки документа. Пример: document.SetField("ИмяПоля", value); документ.УстановитьПоле("ИмяПоля", значение);

Для обмена документами используются следующие функции объекта StorageConnector:

StorageConnector [Соединение]

Методы
Наименование
Параметры и возвращаемое значение
Описание
Выгрузка документов:

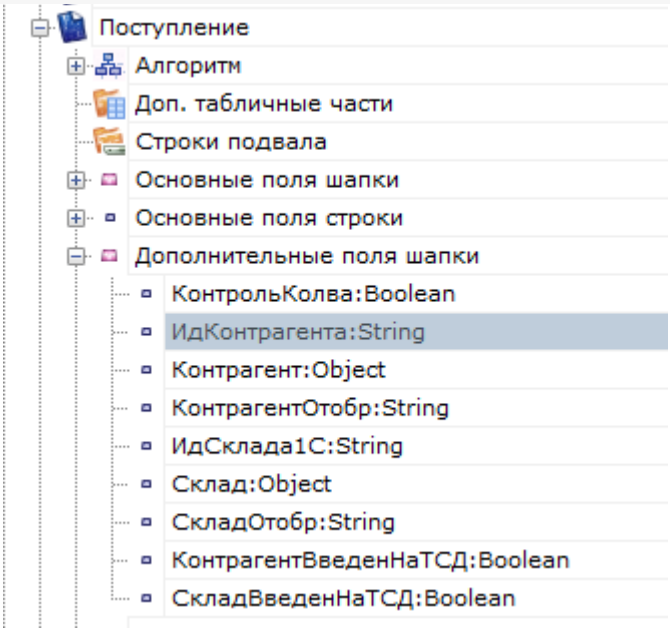
SetDocument [ВыгрузитьДокумент]
Параметры: Document document document Выгружаемый документ. Возвращаемое значение: нет
Выгружает документ в базу Mobile SMARTS.
SetDocuments [ВыгрузитьДокументы]
Параметры: DocumentCollection documents documents Коллекция выгружаемых документов. Возвращаемое значение: нет
Выгружает сразу несколько документов в базу Mobile SMARTS.
Получение документов:
GetDocuments [ПолучитьДокументы]
Параметры: string docType, bool checkForFinish docType Наименование типа документов, которые следует получить. Если передать пустую строку, возвращает все документы, независимо от типа. checkForFinish Если Истина, вернутся только завершенные на терминале документы. Если Ложь, все документы. Возвращаемое значение: Коллекция документов, отобранных в соответствии с заданными условиями.
Получает документы из базы Mobile SMARTS в соответствии с заданными параметрами. Если документов, удовлетворяющих условиям нет, вернется пустая коллекция.

GetDocument **[ПолучитьДокумент]**
Параметры: **string documentId** **documentId** **Идентификатор документа.** **Возвращаемое значение: объект документа Document или null.**
Получает документ из базы Mobile SMARTS по идентификатору.
GetDocumentsByIds **[ПолучитьДокументыПоИдентиф икаторам]**
Параметры: **StringCollection idList** **idList** **Коллекция идентификаторов документов.** **Возвращаемое значение: коллекция документов DocumentCollection.**
Получает из базы Mobile SMARTS документы с заданными идентификаторами.
GetDocument **[ПолучитьДокумент]**
Параметры: **string idocumentId** **idocumentId** **Идентификатор документа.** **Возвращаемое значение: объект документа Document или null, если документ не найден.**
Получает документ из базы Mobile SMARTS по идентификатору.
Удаление документов:

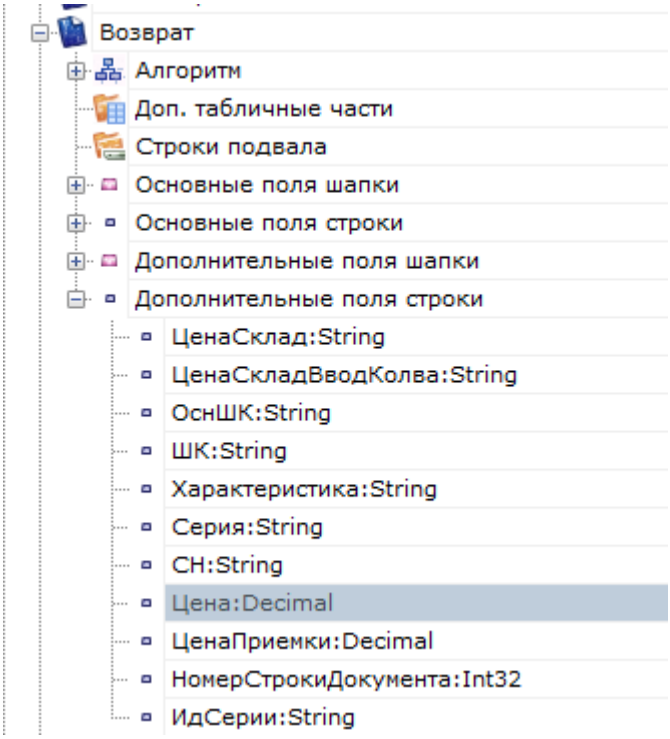
RemoveDocument [УдалитьДокумент]
Параметры: string documentId documentId Идентификатор документа. Возвращаемое значение: нет.
Удаляет документ из базы Mobile SMARTS по идентификатору. Если документ не найден, функция ничего не делает.
RemoveDocuments [УдалитьДокументы]
Параметры: DocumentCollection documents documents Коллекция документов. Возвращаемое значение: нет.
Удаляет заданные документы из базы.

Выгрузка документов

Выгрузка из учетной системы используется, если требуется отправить на терминал определенное задание для работы. В учетной системе, как правило, имеется некоторый документ, содержащий список товаров с количествами, а также, возможно, другие данные (поля шапки и строк документа, табличные части). Например, накладная на приемку. Чтобы реализовать выгрузку документа из учетной системы, нужно определить, какие данные нужны на терминале при работе с документом, т.е. определить состав выгружаемых полей и соответствие полей документа из учетной системы полям документа Mobile SMARTS. Кроме основных полей, объекты Document [Документ] и DocumentItem [СтрокаДокумента] могут содержать произвольные дополнительные поля, доступ к которым осуществляется с помощью функций SetField [УстановитьПоле] и GetField [ПолучитьПоле]. Состав дополнительных полей шапки и строки определяется для каждого типа документа в Панеле управления Mobile SMARTS (см. Тип документа):



Поля шапки



Поля строки

Примечание: некоторые поля, задаваемые в конфигурации Mobile SMARTS являются вычислимыми, т.е. их значение определяется путем некоторых операций над другими полями. Для вычислимого поля задается шаблон значения:

Тип поля	Decimal
Шаблон значения	Item.НоваяЦена > 0? Item.НоваяЦена:Item.Цена

Присваивать значения вычислимых полей при выгрузке нельзя, однако при загрузке получить значение такого поля можно.

Общий алгоритм выгрузки документа. В общем виде алгоритм выгрузки выглядит так:

1. Создаем объект Cleverence.Warehouse.StorageConnector и устанавливаем подключение к нужной базе Mobile SMARTS при помощи вызова SelectCurrentApp[УстановитьПодключениеСБазойСМАРТС];
2. Получаем в учетной системе необходимые для выгрузки документа данные (объект документа учетной системы, некую выборку и т.п.);
3. Создаем объект Document [Документ];
4. Устанавливаем у созданного объекта Document [Документ] свойства:
   - Id [Ид] - обычно равен идентификатору или номеру документа в учетной системе. Может быть так, что в учетной системе документы различных типов имеют одинаковый номер и эти разные типы документов выгружаются в Mobile SMARTS. В этом случае Id [Ид] нужно формировать так, чтобы он однозначно определял документ в Mobile SMARTS и для разных выгружаемых документов не мог повторяться (например, <Имя типа документа><разделитель><номер> - “Приемка#БРД000003”);

   - Name [Имя] - название документа, которое отображается на терминале при выборе документа из списка и в шапках окон (если не отключено настройками в конфигурации). Можно использовать наименование документа из учетной системы (например, “Приемка БРД000003 от 12.06.16”). Примечание: иногда в списке документов на терминале нужно отобразить дополнительную информацию (например, контрагента, от которого пришел товар), выделить какую-то часть текста в списке цветом и т.п., чтобы пользователю было проще найти нужный документ. В этом случае следует использовать свойство “Шаблон отображения документов в списке” типа документа в Панеле управления, через шаблон можно вывести значения любых основных и дополнительных полей шапки документа (см. Шаблоны);

   - DocumentTypeName [ИмяТипаДокумента] - должно быть равно имени одного из типов документов в конфигурации Mobile SMARTS. Например, “Приемка”;

   - WarehouseId [ИдСклада] - должен быть равен Ид. одного из складов, имеющихся в конфигурации Mobile SMARTS. В типовых конфигурациях по умолчанию существует единственный склад “Общий” с Ид. равным “1”;

   - Appointment [Назначение] - код пользователя или имя группы пользователей, которым данный документ назначается на исполнение. Может быть пустым (“общий” документ, имеет смысл только для серверной базы Mobile SMARTS). В типовых конфигурациях Mobile SMARTS по умолчанию заведен единственный пользователь с Ид. “оператор”. В самом простом случае присваиваем “оператор”. Если пользователей терминалов несколько, в учетной системе может быть реализован выбор пользователя, которому будет отправлен документ. Для этого нужно получить список пользователей из базы Mobile SMARTS (см. Получение пользователей) и предложить выбор. В этом случае присваивается выбранное значение;

   - AutoAppointed [ВыдаватьАвтоматически] - по умолчанию Истина (документ сразу отправляется назначенному пользователю). Если нужно, чтобы документ оставался на сервере Mobile SMARTS, пока пользователь на терминале не выберет его из списка или по ШК, свойство следует проставить в Ложь. Например, оставив поле Appointment [Назначение] пустым, а AutoAppointed [ВыдаватьАвтоматически] выставив в Ложь, можно добиться чтобы документ, выгруженный на сервер, был виден всем пользователям терминалов (“общий” документ). Документ остается видным для всех пользователей, пока кто-то один не откроет его на обработку. Имеет смысл только для серверной базы Mobile SMARTS;

5. Проставляем объекту документа необходимые дополнительные поля шапки при помощи SetField

[УстановитьПоле]. Состав полей зависит от типа документа. Например, для приемки может потребоваться наименование или идентификатор контрагента, которые берем из исходного документа учетной системы;

6. Обходим в цикле строки исходного документа учетной системы (при необходимости перед этим может быть получена какая-то выборка строк. Например, нужно выгружать не все строки или выполнить группировку). На каждой итерации цикла создаем объект DocumentItem[СтрокаДокумента], в строке документа проставляем поля:
   - **ProductId [ИдТовара]** - должен соответствовать ид. одного из товаров из выгруженного справочника номенклатуры. Для “неизвестного товара” (см. ниже) равно “*”. Если номенклатура не выгружается из учетной системы и реализовано он-лайн получение товара с помощью события сервера ТоварНеНайден (см. События сервера, ТоварНеНайден), функция- обработчик события должна возвращать товар по такому идентификатору;

   - **PackingId [ИдУпаковки]** - упаковка с таким Ид. должна быть в коллекции упаковок данного товара в справочнике товаров базы Mobile SMARTS. Для “неизвестного товара” (см. ниже) равно “шт”. Если номенклатура не выгружается из учетной системы и реализовано он-лайн получение товара с помощью события сервера ТоварНеНайден (см. События сервера, ТоварНеНайден), функция-обработчик события должна возвращать товар, содержащий упаковку с таким идентификатором;

   - **DeclaredQuantity [КоличествоПлан]** - плановое количество товара;

   - При необходимости заполняем поля SSCC [КодЕдиницыХранения], FirstStorageBarcode [ШтрихкодПервогоМеста] и др.

   - Проставляем нужные дополнительные поля строки при помощи SetField [УстановитьПоле]. Например, характеристика товара: documentItem.SetField(“Характеристика”, descr) [строкаДок.УстановитьПоле(“Характеристика”, знач)]

   - **Добавляем** созданную и заполненную строку в коллекцию **DeclaredItems [СтрокиПлан]** документа: document.DeclaredItems.Add(documentItem) [документ.СтрокиПлан.Добавить(строкаДок)];

7. Если кроме заполнения плановых строк требуется выгрузка дополнительных табличных частей документа, подготавливаем данные для выгрузки. Для каждой выгружаемой дополнительной таблицы документа:

   7.1. Создаем объект Table [Таблица], проставляем наименование Name [Имя], как оно задано в конфигурации Mobile SMARTS;
   7.2. Обходим в цикле выгружаемый набор данных (табличную часть документа учетной системы), на каждой итерации цикла создаем объект Row [СтрокаТаблицы];
   - Проставляем поля в строке при помощи SetField [УстановитьПоле], наименования полей должны соответствовать заданным в конфигурации Mobile SMARTS для даннойтабличной части;
   - Добавляем созданную строку в коллекцию строк таблицы: documentTable.Rows.Add(row) [таблицаДокумента.Строки.Добавить(строка)];

   7.3. Добавляем созданную и заполненную таблицу к таблицам документа: document.Tables.Add(table) [документ.Таблицы.Добавить(таблица)];
8. **Выгружаем** заполненный документ: storageConnector.SetDocument(document) [соединение.ВыгрузитьДокумент(документ)].

“Неизвестный товар”. Строка документа Mobile SMARTS, как правило, определяет конкретный товар с упаковкой (единицей измерения). В этом случае поля ProductId [ИдТовара] и PackingId [ИдУпаковки] соответствуют идентификатору товара и упаковки из справочника номенклатуры. Но возможна ситуация, когда справочник номенклатуры вообще не используется при работе на терминале, а вся необходимая информация содержится в строках документа. Например, в строке документа записан номер рабочего инструмента (без привязки к номенклатуре), а документ представляет собой список всех инструментов, которые необходимо

выдать. Тогда номер инструмента записывается в дополнительное поле строки, а при помощи ProductId [ИдТовара] и PackingId [ИдУпаковки] задается “неизвестный товар”. В случае “неизвестного товара” ProductId [ИдТовара] должен быть равен “*”, PackingId [ИдУпаковки] – “шт”.

Пример выгрузки

C#:

```
var connector = new Cleverence.Warehouse.StorageConnector(); // Создаем экземпляр
 объекта соединения



// Выполняем подключение к базе Mobile SMARTS

connector.SelectCurrentApp("михаил-пк:10501/d7c15f54-fd05-4811-8f6d-c7c7003887c5"); try

{

var document = new Cleverence.Warehouse.Document(); //Создаем объект документа Mobile
SMARTS

var acceptanceDoc = AcceptanceJournal.Find(acceptanceDocId); //Получаем документ учетной
системы



document.Id = acceptanceDoc.Id; //Ид. документа document.Name = acceptanceDoc.Name;
//Имя документа

document.DocumentTypeName = "Поступление"; //Имя
типа документа document.WarehouseId = "1"; //Ид. склада, к которому привязан документ



if(sharedDoc)

{

//"Общий" документ - назначение пустое, остается на сервере до явного выбора.
```

```
document.Appointment = "";

document.AutoAppointed = false;

}

else

{

document.Appointment = selectedUserId; ///Ид. выбранного пользователя Mobile SMARTS или можно

//присвоить "оператор"
для типовых конфигураций Mobile SMARTS

}



//Проставляем дополнительные поля шапки документа document.SetField("ИдКонтрагента", acceptanceDoc.Contractor.Id); document.SetField("ИдСклада", acceptanceDoc.Warehouse.Id);



// Обходим в цикле строки табличной части исходного
докумнта foreach(var acceptanceDocItem in acceptanceDoc.InventItems)

{

//Создаем строку документа Mobile SMARTS и присваиваем Ид. товара, Ид. упаковки и плановое //количество товара

var documentItem = new Cleverence.Warehouse.DocumentItem(); documentItem.ProductId = acceptanceDocItem.InventItem.Id; documentItem.PackingId = acceptanceDocItem.Unit.Name; documentItem.DeclaredQuantity = acceptanceDocItem.Quantity;
```

```
//Проставляем дополнительные поля строки документа if(acceptanceDocItem.InventDim != null)

documentItem.SetField("Характеристика", acceptanceDocItem.InventDim.Name);



documentItem.SetField("Цена", acceptanceDocItem.Price);



//Добавляем строку в коллекцию плановых строк

document.DeclaredItems.Add(documentItem);

}



//Создание дополнительной таблицы документа

var boxesTable = new Cleverence.Warehouse.DocumentTable(); boxesTable.Name = "Короба"; //Присваиваем имя



foreach(var boxItem in acceptanceDoc.Boxes)

{

//Создаем в цикле строки доп. таблицы

var row = new Cleverence.Warehouse.Row(); row.SetField("Номер", boxItem.Id); row.SetField("Описание", boxItem.Description);

boxesTable.Rows.Add(row); //Добавление строки в
 коллекцию строк таблицы

}
```

```
}



//Добавляем таблицу в
 коллекцию таблиц документа document.Tables.Add(boxesTable);



connector.SetDocument(document); //Выгружаем
 документ



}

catch

{

//Обработка ошибки

}
```

«1С:Предприятие 8»:

```
connector = новый COMОбъект("Cleverence.Warehouse.StorageConnector"); //
Создаем экземпляр объекта
//соединения Попытка
connector.УстановитьПодключениеСБазойСМАРТС("михаил-пк:10501/d7c15f54-fd05-4811-8f6d-
c7c7003887c5");
//Выполняем подключение
ДокументТСД = новый COMОбъект("Cleverence.Warehouse.Document"); Документ1С =
Документы.ПоступлениеТоваров.НайтиПоНомеру(НомерДокумента);
 ДокументТСД.Ид = “Поступление#”+Документ1С.Номер; //Ид. документа ДокументТСД.Имя
= Строка(Документ1С); //Имя документа ДокументТСД.ИмяТипаДокумента = "Поступление";
//Имя типа документа ДокументТСД.ИдСклада = "1"; //Ид. склада, к которому привязан
документ
 Если
ОбщийДокумент = Истина Тогда
//"Общий" документ - назначение пустое, остается на сервере до явного выбора.
ДокументТСД.Назначение =
"";
ДокументТСД.ВыдаватьАвтоматически = Ложь; Иначе
ДокументТСД.Назначение = ИдПользователяТСД; //Ид. выбранного пользователя Mobile
```

```
SMARTS или можно
//присвоить "оператор"
для типовых конфигураций Mobile SMARTS КонецЕсли;
 //Проставляем дополнительные поля шапки документа
ДокументТСД.УстановитьПоле("ИдКонтрагента", ДокументТСД.Контрагент.Код);
ДокументТСД.УстановитьПоле("ИдСклада", ДокументТСД.Склад.Код);
 Для каждого строкаДок1С из Документ1С.Товары Цикл
//Создаем строку документа Mobile SMARTS и присваиваем Ид. товара, Ид. упаковки и
плановое //количество товара



СтрокаДокументаТСД = Новый COMОбъект("Cleverence.Warehouse.DocumentItem");
СтрокаДокументаТСД.ИдТовара = XMLСтрока(строкаДок1С.Номенклатура);

СтрокаДокументаТСД.ИдУпаковки = строкаДок1С.Упаковка.Наименование;
СтрокаДокументаТСД.КоличествоПлан = строкаДок1С.Количество;
 //Проставляем дополнительные поля строки документа
СтрокаДокументаТСД.УстановитьПоле(“Характеристика”,
Строка(строкаДок1С.Характеристика)); СтрокаДокументаТСД.УстановитьПоле(“Цена”,
строкаДок1С.Цена);
 //Добавляем строку в коллекцию плановых строк
ДокументТСД.СтрокиПлан.Добавить(СтрокаДокументаТСД);
КонецЦикла;
 //Создание дополнительной таблицы документа
ТаблицаКоробовТСД = Новый COMОбъект("Cleverence.Warehouse.Table");
ТаблицаКоробовТСД.Имя =
 “Короба”;
 Для каждого строкаКоробаДок1С из Документ1С.Короба Цикл
//Создаем в цикле строки доп. таблицы
СтрокаКоробаТСД = Новый COMОбъект("Cleverence.Warehouse.Row");
СтрокаКоробаТСД.УстановитьПоле(“Номер”, строкаКоробаДок1С.НомерКороба);
СтрокаКоробаТСД.УстановитьПоле(“Описание”, строкаКоробаДок1С.Описание);
 ТаблицаКоробовТСД.Строки.Добавить(СтрокаКоробаТСД); //Добавление строки в коллекцию
строк таблицы КонецЦикла;
 //Добавляем таблицу в
 коллекцию таблиц документа ДокументТСД.Таблицы.Добавить(ТаблицаКоробовТСД);
connector.ВыгрузитьДокумент(ДокументТСД); //Выгружаем документ Исключение
// При выгрузке возникли ошибки, обрабатываем исключение




КонецПопытки;
```

Загрузка документов

Для того, чтобы отразить результаты работы пользователей терминалов в учетной системе, следует загрузить завершенные документы, содержащие фактические данные. Например, после приемки товара, кладовщик завершает документ на терминале, в документе содержится список фактически принятых товаров с количествами, этот список загружается в учетную систему и на его основе заполняется табличная часть документа учетной системы, после чего документ проводится и в учетной системе фиксируется принятый товар.

Общий алгоритм загрузки документов.

1. **Создаем объект Cleverence.Warehouse.StorageConnector и устанавливаем подключение к нужной базе Mobile SMARTS при помощи вызова SelectCurrentApp [УстановитьПодключениеСБазойСМАРТС];**

2. **Получаем из базы Mobile SMARTS документы для загрузки. Если нужно загрузить все завершенные документы (возможно, с фильтром по типу документа), используем функцию GetDocuments [ПолучитьДокументы]. Если известен идентификатор документа, который требуется загрузить, получаем один этот документ с помощью функции GetDocument [ПолучитьДокумент];**

3. **Для загрузки данных из документа Mobile SMARTS может быть создан новый документ учетной системы или загрузка может выполняться в существующий. Если загружается документ ранее выгруженный из учетной системы, загрузка может выполняться в исходный документ (при этом заполняется фактический список товаров и количества) или может быть создан новый документ другого типа (например, выгружали Заказ поставщику, при загрузке создаем Поступление товаров), также пользователь учетной системы может выбрать документ, в который будет происходить загрузка. Конкретный вариант определяется принятым бизнес-процессом. Перед загрузкой в существующий документ нужно обнулить количества товара в табличной части документа, в который происходит загрузка, или очистить табличную часть (Если для количества используется только одно поле и нет двух полей, к примеру, Количество и КоличествоФакт).**

4. **Получаем из документа Mobile SMARTS поля шапки, которые требуются, с помощью вызова GetField [ПолучитьПоле], и записываем нужные данные в поля документа учетной системы;**

5. **Обходим в цикле строки фактической части документа Mobile SMARTS CurrentItems [СтрокиФакт], на каждой итерации цикла:**
   5.1. На основании значений полей ProductId [ИдТовара], PackingId [ИдУпаковки] строки документа Mobile SMARTS находим в учетной системе товар и упаковку (единицу измерения);
   5.2. Получаем из строки документа Mobile SMARTS необходимые при загрузке дополнительные поля при помощи GetField [ПолучитьПоле] (например, Характеристика, Цена);
   5.3. Ищем в табличной части документа учетной системы строку с данным товаром и другими соответствующими полями (например, характеристикой). Если строка найдена, прибавляем количество товара. Если нет - создаем новую строку, записываем в нее товар, упаковку, другие поля если нужно (например, характеристику, цену);

6. **Когда все данные загружены, записываем документ в учетной системе;**

7. **Если загруженный документ Mobile SMARTS больше не нужен, удаляем его из базы при помощи функции RemoveDocument [УдалитьДокумент].**

Пример загрузки

C#:

```
var connector = new Cleverence.Warehouse.StorageConnector(); // Создаем экземпляр
 объекта соединения
 // Выполняем подключение к базе Mobile SMARTS
connector.SelectCurrentApp("михаил-пк:10501/d7c15f54-fd05-4811-8f6d-c7c7003887c5");
try
{
// Получаем из базы все завершенные документы поступления
var documents = connector.GetDocuments("Поступление", true);
 //Обходим в цикле полученные документы и обрабатываем каждый документ foreach(var
document in documents)
```

```
{
var acceptanceDoc = AcceptanceJournal.Find(document.Id); //Ищем в учетной системе
исходный //документ приемки, который был выгружен в Mobile SMARTS
if(acceptanceDoc == null) //Если документ не найден, создаем новый acceptanceDoc =
AcceptanceJournal.CreateDocument();
else
acceptanceDoc.InventItems.Clear(); //Очистим строки, чтобы загрузить фактически
набранный //товар
 //Получаем поля шапки документа Mobile SMARTS
string contractorId = document.GetField("ИдКонтрагента") as string; string warehouseId
= document.GetField("ИдСклада") as string;
 //На основании полей шапки документа
 Mobile SMARTS заполняем
 поля шапки документа учетной системы if(!String.IsNullOrEmpty(contractorId))
{
acceptanceDoc.Contractor = ContractorCatalog.FindById(contractorId);
}
if(!String.IsNullOrEmpty(warehouseId))
{
acceptanceDoc.Warehouse = WarehouseCatalog.FindById(warehouseId);
}
 //Обходим в цикле фактические строки документа
Mobile SMARTS foreach(var documentItem in document.CurrentItems)
{
//Получаем из строки документа Mobile SMARTS Ид. товара, Ид. упаковки и
 количество string productId = documentItem.ProductId;
string packingId = documentItem.PackingId;




decimal currentQuantity = documentItem.CurrentQuantity;

//Получаем дополнительные поля строки документа
string descr = documentItem.GetField("Характеристика"); decimal price =
documentItem.GetField("Цена");
 //Находим в учетной системе
 товар, единицу и характеристику var inventItem = InventItemCatalog.FindById(productId); var
unit = UnitCatalog.FindByName(packingId);
 var inventDim = (from inventDim in InventDimCatalog
where inventDim.Owner == inventItem && inventDim.Name == descr select
inventDim).FirstOrDefault();
 //Ищем строку документа учетной системы
 с полученными товаром, единицей и характеристикой var acceptanceDocItem = (from
docItem in acceptanceDoc.InventItems
where docItem.InventItem == inventItem
&& docItem.Unit == unit &&
 docItem.InventDim == inventDim
select docItem).FirstOrDefault();
 if(acceptanceDocItem != null)
{
//Если строка найдена, прибавляем количество acceptanceDocItem.Quantity +=
```

```
currentQuantity;
}
else
{
//Строка не найдена, добавляем строку и заполняем в ней нужные поля
acceptanceDocItem = acceptanceDoc.InventItems.AddNew(); acceptanceDocItem.InventItem =
inventItem;
acceptanceDocItem.Unit = unit; acceptanceDocItem.InventDim = inventDim;
acceptanceDocItem.Price = price; acceptanceDocItem.Quantity = currentQuantity;
}
}
 //Все строки загрузили, сохраняем документ acceptanceDoc.Save();
 //Удаляем из базы Mobile SMARTS загруженный документ
connector.RemoveDocument(document.Id);
}
}
catch
{
//Обработка ошибки



}
```

«1С:Предприятие 8»:

```
connector = новый COMОбъект("Cleverence.Warehouse.StorageConnector"); //
Создаем экземпляр объекта
//соединения Попытка
connector.УстановитьПодключениеСБазойСМАРТС("михаил-пк:10501/d7c15f54-fd05-4811-8f6d-
c7c7003887c5");
//Выполняем подключение
 // Получаем из базы все завершенные документы поступления ДокументыТСД =
connector.ПолучитьДокументы("Поступление", Истина);



//Обходим в цикле полученные документы и обрабатываем каждый документ


  Для ИндДок = 0 По ДокументыТСД.Количество-1 Цикл ДокументТСД =
ДокументыТСД.Элемент(ИндДок);


Документ1С = НайтиДокумент1СПоИдТСД(ДокументТСД.Ид); //Получаем документ 1С по
ид. документа ТСД. //При выгрузке из 1С назначали ид. вида
“Поступление#”+Документ1С.Номер, для созданного на //терминале документ 1С =
Неопределено, создаем новый документ 1С
```

```
Если Документ1С = Неопределено
Тогда

Документ1С = Документы.ПоступлениеТоваров.СоздатьДокумент(); Иначе

Документ1С.Товары.Очистить(); КонецЕсли;



//Получаем поля шапки документа Mobile SMARTS КодКонтрагента =
ДокументТСД.ПолучитьПоле("ИдКонтрагента"); КодСклада =
ДокументТСД.ПолучитьПоле("ИдСклада");



//На основании полей шапки документа Mobile SMARTS заполняем поля шапки
документа 1С Если
ЗначениеЗаполнено(КодКонтрагента) Тогда

Документ1С.Контрагент =
 Справочники.Контрагенты.НайтиПоКоду(КодКонтрагента); КонецЕсли;



Если ЗначениеЗаполнено(КодСклада) Тогда

Документ1С.Склад = Справочники.Склады.НайтиПоКоду(КодСклада); КонецЕсли;



//Обходим в цикле фактические строки документа Mobile SMARTS Для ИндСтроки = 0 По
ДокументТСД.СтрокиФакт.Количество-1 Цикл

СтрокаДокументаТСД = ДокументТСД.СтрокиФакт.Элемент(ИндСтроки); //Получаем строку
документа
```

```
//Получаем из строки документа Mobile SMARTS Ид. товара, Ид. упаковки и
 количество ИдТовара  =  СтрокаДокументаТСД.ИдТовара;


ИмяУпаковки  =  СтрокаДокументаТСД.ИдУпаковки; Количество  =
СтрокаДокументаТСД.КоличествоФакт;




//Получаем дополнительные поля строки документа


ХарактеристикаИмя  =  СтрокаДокументаТСД.ПолучитьПоле(“Характеристика”); Цена  =
СтрокаДокументаТСД.ПолучитьПоле(“Цена”);




//Находим в 1С товар, единицу и характеристику


Номенклатура  =  Справочники.Номенклатура.ПолучитьСсылку(новый
УникальныйИдентификатор(ИдТовара  ));




Упаковка  =
Справочники.ЕдиницыИзмерения.НайтиПоНаименованию(ИмяУпаковки,,,Номенклатура);




Если  ЗначениеЗаполнено(ХарактеристикаИмя)  Тогда


Характеристика  =  Справочники.Характеристики.НайтиПоНаименованию(ХарактеристикаИмя
,,,Номенклатура); КонецЕсли;




//Ищем строку документа 1С с полученными товаром, упаковкой и характеристикой
ПараметрыОтбора  =  Новый  Структура;


ПараметрыОтбора.Вставить("Номенклатура",
 Номенклатура); ПараметрыОтбора.Вставить("Упаковка",
 Упаковка); ПараметрыОтбора.Вставить("Характеристика",  Характеристика);
```

```
НайденныеСтроки = Документ1С.Товары.НайтиСтроки(ПараметрыОтбора);



Если НайденныеСтроки.Количество() > 0 Тогда СтрокаТабличнойЧасти =
НайденныеСтроки[0];


//Если строка найдена, прибавляем количество


СтрокаТабличнойЧасти.Количество = СтрокаТабличнойЧасти.Количество + Количество;
Иначе
//Строка не найдена, добавляем строку и заполняем в ней нужные поля
СтрокаТабличнойЧасти = Документ1С.Товары.Добавить();
СтрокаТабличнойЧасти.Номенклатура = Номенклатура; СтрокаТабличнойЧасти.Упаковка =
Упаковка; СтрокаТабличнойЧасти.Характеристика = Характеристика;
СтрокаТабличнойЧасти.Количество = Количество; СтрокаТабличнойЧасти.Цена = Цена;


КонецЕсли;



//Записываем документ в
 1С Документ1С.Записать(РежимЗаписиДокумента.Проведение);



//Удаляем загруженный документ Mobile SMARTS из базы
connector.УдалитьДокумент(ДокументТСД.Ид);


КонецЦикла; КонецЦикла;


Исключение


// Обработка

ошибки



КонецПопытки;
```

COM-компонента, обмен данными

Не нашли что искали?

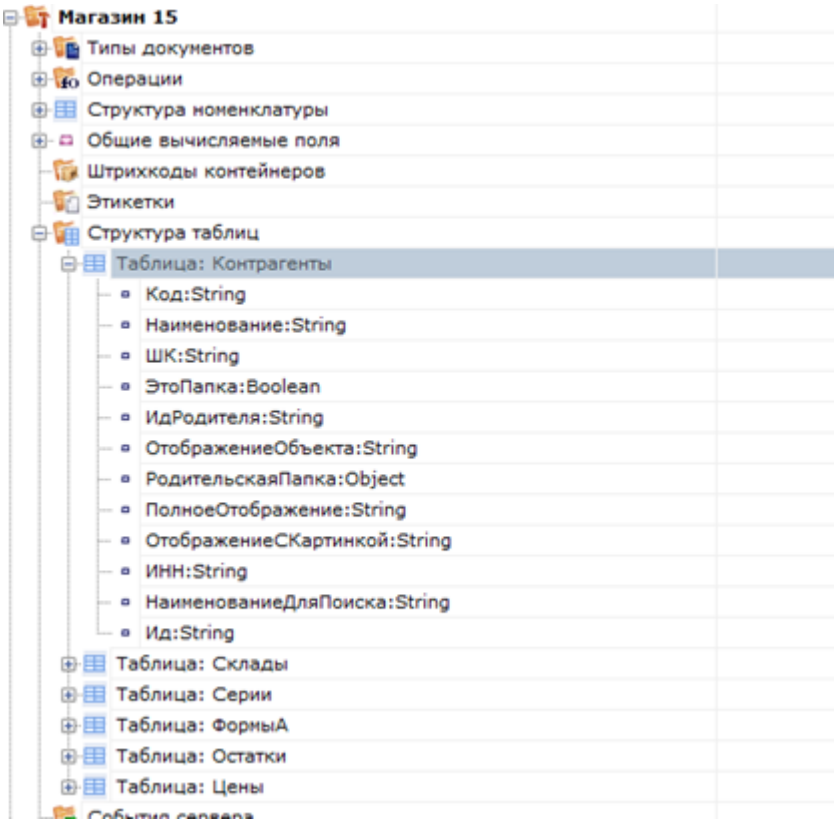
Задать вопрос в техническую поддержку

Дополнительные таблицы Mobile SMARTS

Последние изменения: 2024-03-26

Дополнительная таблица представляет собой “плоский” набор данных, строки таблицы содержат поля простых типов (числа, строки, булевы). Структура данных дополнительных таблиц создается и редактируется с помощью Панели управления Mobile SMARTS. Разработчик конфигурации может создавать произвольные дополнительные таблицы. В таблице кроме полей, которые содержат данные, могут быть заданы вычислимые поля (значения вычислимых полей определяется путем операций над значениями других полей), для вычислимых полей задается Шаблон значения. Таблица может как загружаться на терминал, так и хранится на сервере (в случае серверной базы Mobile SMARTS).

Например, таблица Контрагенты используется для хранения списка сторонних организаций, на терминале при работе с документом Приемки пользователь может выбирать организацию, от которой пришел товар. В таблице Остатки хранятся остатки товаров в разрезе складов, привязка к номенклатуре выполняется с помощью поля ИдНоменклатуры.



Дополнительные таблицы

Для работы с дополнительными таблицами используется объект TableAccessor [ДоступКТаблице], с помощью функций этого объекта можно выгружать данные в таблицу, читать данные, добавлять и удалять строки таблицы. Для получения объекта TableAccessor [ДоступКТаблице] следует использовать функцию GetTableAccessor [ПолучитьДоступКТаблице] из StorageConnector [Соединение]. Перед вызовом GetTableAccessor [ПолучитьДоступКТаблице] должно быть установлено подключение к базе данных Mobile SMARTS с помощью SelectCurrentApp [УстановитьПодключениеСБазойСМАРТС].

Наименование
Параметры и возвращаемое значение
Описание

GetTableAccessor [ПолучитьДоступКТаблице]
string tableName Имя таблицы, для которой нужно получить объект доступа. Должно соответствовать имени одной из таблиц из конфигурации Mobile SMARTS. Возвращаемое значение: объект TableAccessor [ДоступКТаблице].
Возвращает объект TableAccessor [ДоступКТаблице], позволяющий выгружать данные в таблицу, читать данные, добавлять и удалять строки таблицы.

Объект TableAccessor [ДоступКТаблице] имеет следующие функции:

Наименование
Параметры и возвращаемое значение
Описание
Выгрузка
BeginUpload [НачатьВыгрузку]
bool overwrite Истина - полностью перезаписать таблицу при выгрузке, Ложь - добавлять строки Возвращаемое значение: нет
Открывает выгрузку данных в таблицу.
Upload [ДобавитьВВыгрузку]
RowCollection rows rows - коллекция выгружаемых строк Возвращаемое значение: нет
Добавляет порцию строк в выгрузку.

EndUpload [ЗавершитьВыгрузку]
Параметры: нет Возвращаемое значение: нет
Завершает выгрузку. Данные сохраняются в базу Mobile SMARTS.
ResetUpload [СброситьВыгрузку]
Параметры: нет Возвращаемое значение: нет
Сбрасывает начатую выгрузку без сохранения данных.
Чтение данных Свойства:
Count [Количество]
Возвращаемое значение: Число
Количество строк в таблице.
Item [Элемент]
int index index - индекс строки Возвращаемое значение: нет
Получает строку по индексу.
Поиск
FindByField [НайтиПоЗначениюПоля]
string fieldName, object value fieldName Наименование поля для отбора. value Значение для отбора Возвращаемое значение: TableAccessor [ДоступКТаблице]
Выполняет поиск в таблице по значению заданного поля. Возвращает объект TableAccessor [ДоступКТаблице], из которого можно читать строки, удовлетворяющие отбору.

FindFirstByField [НайтиПервуюЗаписьПоЗначени юПоля]
string fieldName, object value   fieldName Наименование поля для отбора. value Значение для отбора Возвращаемое значение: Row Найденная строка таблицы или null.
Выполняет поиск в таблице по значению заданного поля и возвращает первую найденную строку или null, если строка не найдена.
Редактирование
Add [Добавить]
Row item item - добавляемая строка Возвращаемое значение: нет
Добавляет строку в таблицу. Чтобы зафиксировать сделанные изменения, следует вызвать CommitChanges [СохранитьИзменения].
Remove [Удалить]
Row item
Удаляет строку из таблицы.
item - строка таблицы для удаления Возвращаемое значение: bool
Чтобы зафиксировать сделанные изменения, следует вызвать CommitChanges [СохранитьИзменения]. .
CommitChanges [СохранитьИзменения]
Параметры: нет Возвращаемое значение: нет
Сохраняет сделанные изменения в таблицу в базе данных Mobile SMARTS.

UndoChanges [ОтменитьИзменения]
Параметры: нет Возвращаемое значение: нет
Отменяет сделанные изменения (добавленные, удаленные строки).

Пример выгрузки

C#:

```
 var connector  = new Cleverence.Warehouse.StorageConnector(); // Создаем экземпляр  объекта соединения

  // Выполняем подключение  к базе Mobile  SMARTS

  connector.SelectCurrentApp("михаил-пк:10501/d7c15f54-fd05-4811-8f6d-c7c7003887c5");

  // Получаем объект для доступа к таблице

  var tableContractors = connector.GetTableAccessor("Контрагенты");

  try

  {

  tableContractors.BeginUpload(true); // Начинаем выгрузку

  var rows = new Cleverence.Warehouse.RowCollection(); // Коллекция строк таблицы, которую будем

  // использовать при выгрузке

    using (SqlConnection connection  = new SqlConnection(connectionString)) //Устанавливаем соединение  с  //БД, из которой будем читать данные

  {

  connection.Open(); command.Connection = connection;

  SqlDataReader reader  = command.ExecuteReader(); while (reader.Read()) //Читаем данные

  {

  string id  = reader.GetString(0); string name = reader.GetString(1); string inn  = reader.GetString(1);

    if(rows.Count >= 1000)

  {

  //Если  набрали  достаточно строк, выгружаем  порцию строк и  создаем  новый объект
```

```
коллекции,

  //чтобы набирать строки дальше  в пустую коллекцию. tableContractors.Upload(rows);

  rows = new Cleverence.Warehouse.RowCollection();

  }

  var row = new Cleverence.Warehouse.Row(); //Создаем строку таблицы row.SetField("Ид", id);
//Проставляем поля row.SetField("Наименование", name);

  row.SetField("ИНН", inn);

  rows.Add(row); //Добавляем строку в коллекцию строк

  }

    //Если  есть строки  для  выгрузки, выгружаем

  } if(rows.Count > 0)

  tableContractors.Upload(rows);

  }

    tableContractors.EndUpload(); //Завершаем выгрузку

  }

  catch

  {

  //Обработка ошибки

  tableContractors.ResetUpload(); //Если  возникла ошибка, сбрасываем  выгрузку
```

«1С:Предприятие 8»:

```
 connector  = новый  COMОбъект("Cleverence.Warehouse.StorageConnector"); //  Создаем
экземпляр  объекта

  //соединения

  connector.УстановитьПодключениеСБазойСМАРТС("михаил-пк:10501/d7c15f54-fd05-4811-8f6d-
c7c7003887c5");

  //Выполняем подключение

    таблицаКонтрагенты  = connector.ПолучитьДоступКТаблице("Контрагенты"); //  Получаем
объект для доступа к  // таблице
```

```
  Попытка

 таблицаКонтрагенты.НачатьВыгрузку(Истина);  // Начинаем  выгрузку

 строкиДляВыгрузки  = новый COMОбъект("Cleverence.Warehouse.RowCollection"); //
Коллекция строк //таблицы, которую будем  использовать при  выгрузке

  Запрос = Новый Запрос(ТекстЗапроса);

 ТаблицаРезультата  = Запрос.Выполнить().Выгрузить(); //Выполняем запрос

  Для каждого СтрокаТаблицы из ТаблицаРезультата Цикл

 Ид = XMLСтрока(СтрокаТаблицы.КонтрагентСсылка); //Получаем  необходимые  поля
Наименование = СтрокаТаблицы.Наименование;

 ИНН = СтрокаТаблицы.ИНН;

  Если строкиДляВыгрузки.Количество  >= 1000  Тогда

 //Если  набрали  достаточно строк, выгружаем  порцию и очищаем коллекцию строк
таблицаКонтрагенты.ДобавитьВВыгрузку(строкиДляВыгрузки);
таблицаКонтрагенты.Очистить();

 КонецЕсли;

  строкаДляВыгрузки  = новый COMОбъект("Cleverence.Warehouse.Row"); //Создаем строку
таблицы строкаДляВыгрузки.УстановитьПоле(“Ид”,  Ид); //Проставляем  поля
строкаДляВыгрузки.УстановитьПоле(“Наименование”,  Наименование);
строкаДляВыгрузки.УстановитьПоле(“ИНН”, ИНН);

  строкиДляВыгрузки.Добавить(строкаДляВыгрузки);  //Добавляем  строку  в коллекцию
строк КонецЦикла;

  //Если  есть строки  для  выгрузки, выгружаем Если строкиДляВыгрузки.Количество  > 0
Тогда

 таблицаКонтрагенты.ДобавитьВВыгрузку(строкиДляВыгрузки); КонецЕсли;

  таблицаКонтрагенты.ЗавершитьВыгрузку();  //Завершаем  выгрузку Исключение

 // Обработка ошибки



 таблицаКонтрагенты.СброситьВыгрузку(); //Если возникла  ошибка,  сбрасываем выгрузку
КонецПопытки;
```

COM-компонента

Не нашли что искали?

Задать вопрос в техническую поддержку

Работа с компонентой AddIn.Cl.TerminalConnector

Последние изменения: 2024-03-26

Компонента AddIn.Cl.TerminalConnector предназначена для использования в среде «1С:Предприятие» (версий 7.7, 8.x). Компонента реализует стандарт 1С на внешние компоненты, а также стандарты 1С на компоненты драйверов торгового оборудования (терминалов сбора данных). Компонента позволяет выполнять соединение с базой данных Mobile SMARTS, выгружать справочник номенклатуры и дополнительные таблицы, выгружать документы Mobile SMARTS для исполнения на терминале, загружать завершенные документы Mobile SMARTS в 1С. Отличие от Cleverence.Warehouse.StorageConnector заключается в том, что для обмена данными используются объекты 1С (массивы, списки значений, таблицы значений и др.), что существенно упрощает написание кода обмена в 1С. При этом возможно совместное использование Add.Cl.TerminalConnector и Cleverence.Warehouse.StorageConnector, а также всех других COM-объектов Mobile SMARTS (Document [Документ], DocumentItem [СтрокаДокумента], Product [Товар] и др.).

Подключение компоненты и создание объекта

«1С:Предприятие 8»:

```
    ПрогИД = "AddIn.Cl.TerminalConnector";

Если ПодключитьВнешнююКомпоненту(ПрогИД) Тогда КомДляMS = Новый (ПрогИД);

Иначе

//Ошибка подключения компоненты КонецЕсли;
```

«1С:Предприятие 7.7»:

```
    ПрогИД = "AddIn.Cl.TerminalConnector";

Если ПодключитьВнешнююКомпоненту(ПрогИД) <> 0 Тогда ОбъектТСД =
СоздатьОбъект(Компонента);

Иначе

//Ошибка подключения компоненты КонецЕсли;
```

Если по каким-то причинам использование ПодключитьВнешнююКомпоненту с передачей программного идентификатора невозможно (например, на сервере «1С:Предприятие»), компонента может быть создана как COM-объект:

«1С:Предприятие 8»:

```
    ПрогИД = "AddIn.Cl.TerminalConnector";

  КомДляMS = Новый COMОбъект(ПрогИД);

  КомДляMS.УстановитьВерсию1С("v8"); //Необходимо сообщить компоненте версию
платформы 1С (“v7” или “v8”)
```

«1С:Предприятие 7.7»:

```
    ПрогИД = "AddIn.Cl.TerminalConnector";

  КомДляMS = СоздатьОбъект(ПрогИД);

  КомДляMS.УстановитьВерсию1С("v7"); //Необходимо сообщить компоненте версию
платформы 1С (“v7” или “v8”)
```

Подключение к базе данных Mobile SMARTS

Перед выполнением обмена данными (выгрузка номенклатуры, документов и др.) необходимо установить подключение к базе данных Mobile SMARTS, для этого используется функция Open [Подключить].

Наименование
Параметры и возвращаемое значение
Описание
Open [Подключить]
Параметры: **ValuesArray [МассивЗначений],** **DeviceID [ИДУстройства]** **ValuesArray [МассивЗначений] - Массив (для 1С 8.x) или** **Список значений (для 1С 7.7), содержащий один элемент, в котором должен находится идентификатор базы Mobile SMARTS (или строка подключения), к которой происходит подключение.** **DeviceID [ИДУстройства] - Возвращается идентификатор ТСД или пустая строка.** **Возвращаемое значение: Истина (1) — подключение успешно выполнено, Ложь (0) — возникла ошибка. Для получения описания ошибки используйте функцию GetLastError [ПолучитьОшибку].**
Выполняет подключение к базе данных Mobile SMARTS и инициализацию компоненты драйвера для последующего обмена с терминалом, папкой обмена или сервером Mobile SMARTS. **Функция входит в один из стандартов 1С на Драйвер для ТСД (См. Описание стандарта).**

«1С:Предприятие 8»:

```
ПрогИД = "AddIn.Cl.TerminalConnector";

Если ПодключитьВнешнююКомпоненту(ПрогИД) Тогда КомДляMS = Новый (ПрогИД);

МассивЗначений = Новый Массив;

МассивЗначений.Добавить(“михаил-пк:10501/3cb8cb43-ca60-42c5-94d9-bb352dcba9e1”);
//Строка

// подключения или ид. базы


ИДУстройства = “”;

Если КомДляMS.Подключить(МассивЗначений, ИДУстройства) Тогда

//Успешное подключение Иначе

//Ошибка при подключении к базе ОписаниеОшибки = “”;
КомДляMS.ПолучитьОшибку(ОписаниеОшибки);

КонецЕсли;


Иначе

//Ошибка подключения компоненты КонецЕсли;
```

«1С:Предприятие 7.7»:

```
ПрогИД = "AddIn.CI.TerminalConnector";

Если ПодключитьВнешнююКомпоненту(ПрогИД) <> 0 Тогда КомДляMS =
СоздатьОбъект(ПрогИД);


Параметры = СоздатьОбъект("СписокЗначений");

Параметры.ДобавитьЗначение(“михаил-пк:10501/3cb8cb43-ca60-42c5-94d9-bb352dcba9e1”);
//Строка

// подключения или ид. базы


ИДУстройства = “”;

Если КомДляMS.Подключить(Параметры, ИДУстройства) <> 0 Тогда

//Успешное подключение Иначе

//Ошибка при подключении к базе ОписаниеОшибки = “”;

КомДляMS.ПолучитьОшибку(ОписаниеОшибки); КонецЕсли;


Иначе


//Ошибка подключения компоненты КонецЕсли;
```

Выгрузка номенклатуры

Для того, чтобы реализовать выгрузку справочника номенклатуры из 1С в Mobile SMARTS нужно определить, какие данные потребуются на терминале для работы и как отобразятся эти данные на объекты Mobile SMARTS Product [Товар] и Packing [Упаковка] при выгрузке. См. также Выгрузка

номенклатуры. Самое очевидное, что должны выгружаться штрихкоды товаров и наименования (для того, чтобы при сканировании на терминале происходила идентификация товара по штрихкоду). Штрихкод в базе 1С, как правило, привязан не только к позиции номенклатуры, но и к упаковке (единице измерения), а часто и к характеристике номенклатуры. Таким образом, позиции номенклатуры соответствует объект

Product [Товар], упаковкам и характеристикам — Packing [Упаковка]. Наименование характеристики записывается в дополнительное поле Характеристика (descr) упаковки Packing [Упаковка]. Для объектов

Product [Товар] требуются уникальные идентификаторы (поле Id), лучше всего использовать Guid-ы позиций справочника номенклатуры (для 1С 7.7 — коды). Кроме того, при выгрузке могут заполняться различные дополнительные поля номенклатуры, определенные в конфигурации Mobile SMARTS. Например, Колво (qty) — остаток товара на складе, Алко — признак, что товар является алкоголем и на терминале следует сканировать ШК акцизной марки. Когда определены все нужные для выгрузки данные, можно приступить к написанию запроса для их получения. Чаще всего запрос выполняется к справочнику номенклатуры и используются соединения для регистров штрихкодов, остатков, цен и др. Товары, не имеющие штрихкодов, тоже обычно выгружаются в Mobile SMARTS, т.к. могут выгружаться документы, содержащие такие товары и чтобы видеть на терминале наименование и другие данные по товару, нужно, чтобы он присутствовал в справочнике номенклатуры. Кроме того, штрихкод к товару можно присвоить на самом терминале.

Для выгрузки используются следующие функции объекта AddIn.Cl.TerminalConnector:

Наименование
Параметры и возвращаемое значение
Описание
BeginUploadProducts [НачатьВыгрузкуТоваров]
Параметры: schema [СхемаВыгрузки] Массив (для 1С 8.х) или Список значений (для 1С 7.7), описывающий схему выгрузки. Содержит наименования полей объектов Product [Товар] и Packing [Упаковка] в виде «Product.ИмяПоля» или «Packing.ИмяПоля». Значения полей передаются с помощью функции AddProductToUpload [ДобавитьВВыгрузкуТоваров]. Возвращаемое значение: Истина (1) — выполнено успешно, Ложь (0) - возникла ошибка. Для получения описания ошибки используйте функцию GetLastError [ПолучитьОшибку].
Переводит коннектор в режим выгрузки товаров. Выгрузка выполняется с помощью функции AddProductToUpload [ДобавитьВВыгрузкуТоваров].

AddProductToUpload [ДобавитьВВыгрузкуТоваров]

Параметры:

object row [Данные]

Массив (для 1С 8.х) или

Список значений (для 1С 7.7), содержащий поля объектов

Product [Товар] и Packing

[Упаковка], заданные при помощи функции BeginUploadProducts [НачатьВыгрузкуТоваров].

Возвращаемое значение: Истина

(1) — выполнено успешно, Ложь (0)

- возникла ошибка. Для

получения описания ошибки используйте функцию

GetLastError [ПолучитьОшибку].

Добавляет в выгрузку товаров товар с упаковкой.

EndUploadProducts [ЗавершитьВыгрузкуТоваров]

Параметры: нет

Возвращаемое значение: Истина

(1) — выполнено успешно, Ложь (0)

- возникла ошибка. Для

получения описания ошибки используйте функцию

GetLastError [ПолучитьОшибку].

Завершает выгрузку товаров.

ResetUploadProducts [СброситьВыгрузкуТоваров]

Параметры: нет

Возвращаемое значение: Истина

(1) — выполнено успешно, Ложь (0)

- возникла ошибка. Для

получения описания ошибки используйте функцию

GetLastError [ПолучитьОшибку].

Сбрасывает начатую выгрузку. Функция применяется в случае ошибки в процессе выгрузки номенклатуры.

«1С:Предприятие 8»:

```
ПрогИД = "AddIn.Cl.TerminalConnector"; ПодключитьВнешнююКомпоненту(ПрогИД);
Соединение = Новый (ПрогИД);



МассивЗначений = Новый Массив;

МассивЗначений.Добавить(“михаил-пк:10501/3cb8cb43-ca60-42c5-94d9-bb352dcba9e1”);
//Строка

// подключения или ид. базы



ИДУстройства = “”;

Если КомДляMS.Подключить(МассивЗначений, ИДУстройства) Тогда Запрос = Новый
Запрос(ТекстЗапроса);

ТаблицаТоваров = Запрос.Выполнить().Выгрузить(); //Выполняем запрос



//Описываем схему выгрузки мДанные = Новый Массив(26);

//Основные поля товара и упаковки

мДанные.Установить( 0, "Product.Id" );

мДанные.Установить( 1, "Product.Marking" );

мДанные.Установить( 2, "Product.Barcode" );
```

```
мДанные.Установить(    3,  "Packing.Barcode"  );
мДанные.Установить(    4,  "Product.Name"  );
мДанные.Установить(   5, "Product.BasePackingId" );
 мДанные.Установить(   6, "Packing.Id" );
мДанные.Установить(   7, "Packing.Name"  );
 мДанные.Установить(   8, "Packing.UnitsQuantity" );
//Дополнительные поля
мДанные.Установить(   9, "Packing.descr" );
 мДанные.Установить( 10, "Packing.serial" );
 мДанные.Установить( 11, "Packing.price" );
мДанные.Установить( 12, "Packing.qty"  );
мДанные.Установить( 13, "Product.withserial" );
 мДанные.Установить( 14, "Product.withsn" );
мДанные.Установить( 15, "Product.КлючСерий" );
//Дополнительные поля для работы с  алкоголем
мДанные.Установить( 16, "Product.Алко"  );
мДанные.Установить( 17, "Packing.АлкоОбъем" );
мДанные.Установить( 18, "Packing.АлкоКодВ"  );
мДанные.Установить( 19, "Packing.АлкоНаимВ" );
 мДанные.Установить( 20, "Product.АлкоКрепость"  );
мДанные.Установить( 21, "Product.Производитель" );
мДанные.Установить( 22, "Product.ПроизвИНН" );
 мДанные.Установить( 23, "Product.ПроизвКПП" );
мДанные.Установить( 24, "Packing.АлкоКод"  );
мДанные.Установить( 25, "Packing.АлкоМарк"  );
Соединение.УстановитьПоискПоНаименованиюИАртикулу(Истина);  //Устанавливаем
признак - что будет //доступен поиск по наименованию товара на терминале
```

признак, что будет //доступен поиск по наименованию товара на терминале

```
//Начинаем выгрузку

Если Не Соединение.НачатьВыгрузкуТоваров(мДанные) Тогда

//Ошибка при начале выгрузки ОписаниеОшибки = “”;

Соединение.ПолучитьОшибку(ОписаниеОшибки); Соединение.ОсвободитьРесурсы();

Сообщить(ОписаниеОшибки); Возврат;

КонецЕсли;

Для Каждого СтрокаТовара из ТаблицаТоваров Цикл

ИдТовара = XMLСтрока(СтрокаТовара[“НоменклатураСсылка”]); //В качестве идентификатора товара //будем выгружать Guid позиции номенклатуры

Наименование = Строка(СтрокаТовара["Номенклатура"]); Код = СокрЛП(СтрокаТовара["Код"]);

ШК = СокрЛП(СтрокаТовара["Штрихкод"]);

ЕдиницаИзмерения = Строка(СтрокаТовара["Упаковка"]); Характеристика = СокрЛП(СтрокаТовара["Характеристика"]); Серия = СокрЛП(СтрокаТовара["Серия"]);

Цена = СтрокаТовара["Цена"]; Количество = СтрокаТовара["Количество"]; Артикул = СокрЛП(СтрокаТовара["Артикул"]); Коэфф = СтрокаТовара["Коэффициент"];

БазоваяЕдиница = ?(СтрокаТовара["ФлагБазовойЕдиницы"], Строка(СтрокаТовара["Упаковка"]), ""); ЕстьСерии = СтрокаТовара["ЕстьСерии"];

СтатусУказанияСерий = СтрокаТовара["СтатусУказанияСерий"];

мДанные.Установить( 0, ИдТовара ); //Product.Id мДанные.Установить( 1, Артикул ); //Product.Marking

мДанные.Установить( 2, Код ); //Product.Barcode - выгружаем Код позиции номенклатуры, чтобы на

//терминале была возможность поиска по коду

мДанные.Установить( 3, ШК ); //Packing.Barcode - штрихкод упаковки мДанные.Установить( 4, Наименование ); //Product.Name

мДанные.Установить( 5, БазоваяЕдиница); //Product.BasePackingId - ид. базовой упаковки (единицы) //или

пустая строка, если данная единица не является базовой
```

```
  мДанные.Установить(   6, ЕдиницаИзмерения ); //Packing.Id

  мДанные.Установить(   7, ЕдиницаИзмерения ); //Packing.Name  - в  качестве имени  и ид.
упаковки

  //используем сокращенное  наименование  (например,  “шт”,  “кг”) мДанные.Установить(   8,
Коэфф  ); //Packing.UnitsQuantity

  мДанные.Установить(   9, Характеристика  );  //Packing.descr  - наименование  характеристики
или пустая

  //строка,  если  нет  характеристики

  мДанные.Установить( 10, Серия  ); //Packing.serial - наименование серии  или пустая  строка,
если  нет

  //серии

  мДанные.Установить( 11, Цена ); //Packing.price - цена мДанные.Установить( 12, Количество
);  //Packing.qty -  остаток  на складе

  мДанные.Установить( 13, ЕстьСерии  ); //Product.withserial  - ведется учет  по сериям
мДанные.Установить( 14, СтатусУказанияСерий );  //Product.withsn -  есть  серийные номера
мДанные.Установить( 15, ИдТовара ); //Product.КлючСерий -  поле  для  связи  с таблицей
Серии  (если

  //таковая  выгружается), ид. позиции номенклатуруры или  ид.  вида номенклатуры

    //Алко  поля

  мДанные.Установить( 16, СтрокаТовара[“Алко”]);

  мДанные.Установить( 17, СтрокаТовара[“АлкоОбъем”]);

  мДанные.Установить( 18, СтрокаТовара[“АлкоКодВ”]);

  мДанные.Установить( 19, СтрокаТовара[“АлкоНаимВ”]);

  мДанные.Установить( 20, СтрокаТовара[“АлкоКрепость”]);

  мДанные.Установить( 21, СтрокаТовара[“Производитель”]);

  мДанные.Установить( 22, СтрокаТовара[“ПроизвИНН”]);

  мДанные.Установить( 23, СтрокаТовара[“ПроизвКПП”]);

  мДанные.Установить( 24, СтрокаТовара[“АлкоКод”]);

  мДанные.Установить( 25, СтрокаТовара["АлкоМарк"]);

    //Добавляем массив  данных в выгрузку
```

```
 Если Не Соединение.ДобавитьВВыгрузкуТоваров(мДанные) Тогда

 //Произошла  ошибка ОписаниеОшибки = “”;

 Соединение.ПолучитьОшибку(ОписаниеОшибки); Сообщить(ОписаниеОшибки);

  Соединение.СброситьВыгрузкуТоваров(); //Прерываем выгрузку
Соединение.ОсвободитьРесурсы();

 Возврат; КонецЕсли;

  КонецЦикла;

  //Завершение выгрузки

 Если  Не  Соединение.ЗавершитьВыгрузкуТоваров()  Тогда ОписаниеОшибки = “”;
Соединение.ПолучитьОшибку(ОписаниеОшибки); Сообщить(ОписаниеОшибки);

 КонецЕсли;

  Соединение.ОсвободитьРесурсы();

  Иначе

 //Ошибка  при  подключении  к  базе ОписаниеОшибки = “”;
Соединение.ПолучитьОшибку(ОписаниеОшибки); Сообщить(ОписаниеОшибки);

 КонецЕсли;
```

«1С:Предприятие 7.7»:

```
ПрогИД  =  "AddIn.Cl.TerminalConnector";

 ПодключитьВнешнююКомпоненту(ПрогИД); Соединение  = СоздатьОбъект(ПрогИД);

 Параметры  = СоздатьОбъект("СписокЗначений");

 Параметры.ДобавитьЗначение(“михаил-пк:10501/3cb8cb43-ca60-42c5-94d9-bb352dcba9e1”);
//Строка

 // подключения  или  ид.  базы

 ИДУстройства = “”;

 Если Соединение.Подключить(Параметры,  ИДУстройства) <>  0 Тогда

 //Успешное  подключение

  ТаблицаНоменклатуры = ПолучитьТаблицуНоменклатурыДляВыгрузки();
```

```
  //Описываем схему  выгрузки
 мДанные =  СоздатьОбъект("СписокЗначений");
мДанные.ДобавитьЗначение("Product.Id"  );
мДанные.ДобавитьЗначение("Product.Marking"  );
 мДанные.ДобавитьЗначение("Product.Barcode"  );
 мДанные.ДобавитьЗначение("Packing.Barcode"  );
 мДанные.ДобавитьЗначение("Product.Name" );
 мДанные.ДобавитьЗначение("Product.BasePackingId"  );
 мДанные.ДобавитьЗначение("Packing.Id"  );
мДанные.ДобавитьЗначение("Packing.Name" );
мДанные.ДобавитьЗначение("Packing.UnitsQuantity"  );
 //Дополнительные поля
 мДанные.ДобавитьЗначение("Packing.descr" );
мДанные.ДобавитьЗначение("Packing.serial"  );
 мДанные.ДобавитьЗначение("Packing.price" );
 мДанные.ДобавитьЗначение("Packing.qty"  );
мДанные.ДобавитьЗначение("Product.withserial" );
 мДанные.ДобавитьЗначение("Product.withsn"  );
мДанные.ДобавитьЗначение("Product.КлючСерий" );
 //Дополнительные поля для работы с  алкоголем
мДанные.ДобавитьЗначение("Product.Алко" );
мДанные.ДобавитьЗначение("Packing.АлкоОбъем" );
 мДанные.ДобавитьЗначение("Packing.АлкоКодВ" );
мДанные.ДобавитьЗначение("Packing.АлкоНаимВ" );
 мДанные.ДобавитьЗначение("Product.АлкоКрепость" );
 мДанные.ДобавитьЗначение("Product.Производитель"  );
```

```
 мДанные.ДобавитьЗначение("Product.ПроизвИНН" );

 мДанные.ДобавитьЗначение("Product.ПроизвКПП" );

 мДанные.ДобавитьЗначение("Packing.АлкоКод"  );

 мДанные.ДобавитьЗначение("Packing.АлкоМарк" );

   Соединение.УстановитьПоискПоНаименованиюИАртикулу(1);  //Устанавливаем признак,  что будет

 //доступен  поиск по наименованию товара на  терминале

   //Начинаем  выгрузку

 Если Соединение.НачатьВыгрузкуТоваров(мДанные)  =  0 Тогда

 //Ошибка  при  начале  выгрузки ОписаниеОшибки = “”;

 Соединение.ПолучитьОшибку(ОписаниеОшибки); Соединение.ОсвободитьРесурсы();

   Сообщить(ОписаниеОшибки); Возврат;

 КонецЕсли;

 Пока ТаблицаНоменклатуры.ПолучитьСтроку()  = 1 Цикл

 мДанные.УдалитьВсе();

 Код  =  СокрЛП(ТаблицаНоменклатуры.Номенклатура.Код);  //В  качестве  идентификатора товара  будем  выгружать код позиции номенклатуры

 Наименование                    = СокрЛП(ТаблицаНоменклатуры.Номенклатура.Наименование); ШК   =  Запрос.Штрихкод;

 ЕдиницаИзмерения                = СокрЛП(ТаблицаНоменклатуры.ЕдиницаИзмерения); Характеристика                    = СокрЛП(ТаблицаНоменклатуры.Характеристика); Серия                = СокрЛП(ТаблицаНоменклатуры.Серия);

 Цена                    = ТаблицаНоменклатуры.Цена; Количество          = ТаблицаНоменклатуры.Количество; Артикул          =  СокрЛП(ТаблицаНоменклатуры.Артикул);

 ….

 мДанные.ДобавитьЗначение(Код); мДанные.ДобавитьЗначение(Артикул); мДанные.ДобавитьЗначение(Код); мДанные.ДобавитьЗначение(ШК); мДанные.ДобавитьЗначение(Наименование);

 …. мДанные.ДобавитьЗначение(СокрЛП(ТаблицаНоменклатуры.АлкоМарк));

 //Добавляем массив данных в выгрузку
```

```
//Добавляем массив данных в выгрузку

Если Соединение.ДобавитьВВыгрузкуТоваров(мДанные) = 0  Тогда

//Произошла  ошибка ОписаниеОшибки = "";

Соединение.ПолучитьОшибку(ОписаниеОшибки); Сообщить(ОписаниеОшибки);

  Соединение.СброситьВыгрузкуТоваров(); //Прерываем выгрузку
Соединение.ОсвободитьРесурсы();

Возврат; КонецЕсли;

  КонецЦикла;

//Завершение выгрузки

Если  Соединение.ЗавершитьВыгрузкуТоваров()  =  0  Тогда ОписаниеОшибки = "";
Соединение.ПолучитьОшибку(ОписаниеОшибки); Сообщить(ОписаниеОшибки);

КонецЕсли;

Соединение.ОсвободитьРесурсы();

Иначе

//Ошибка  при  подключении  к  базе ОписаниеОшибки = "";
Соединение.ПолучитьОшибку(ОписаниеОшибки);



КонецЕсли;
```

COM-компонента

Не нашли что искали?

Задать вопрос в техническую поддержку

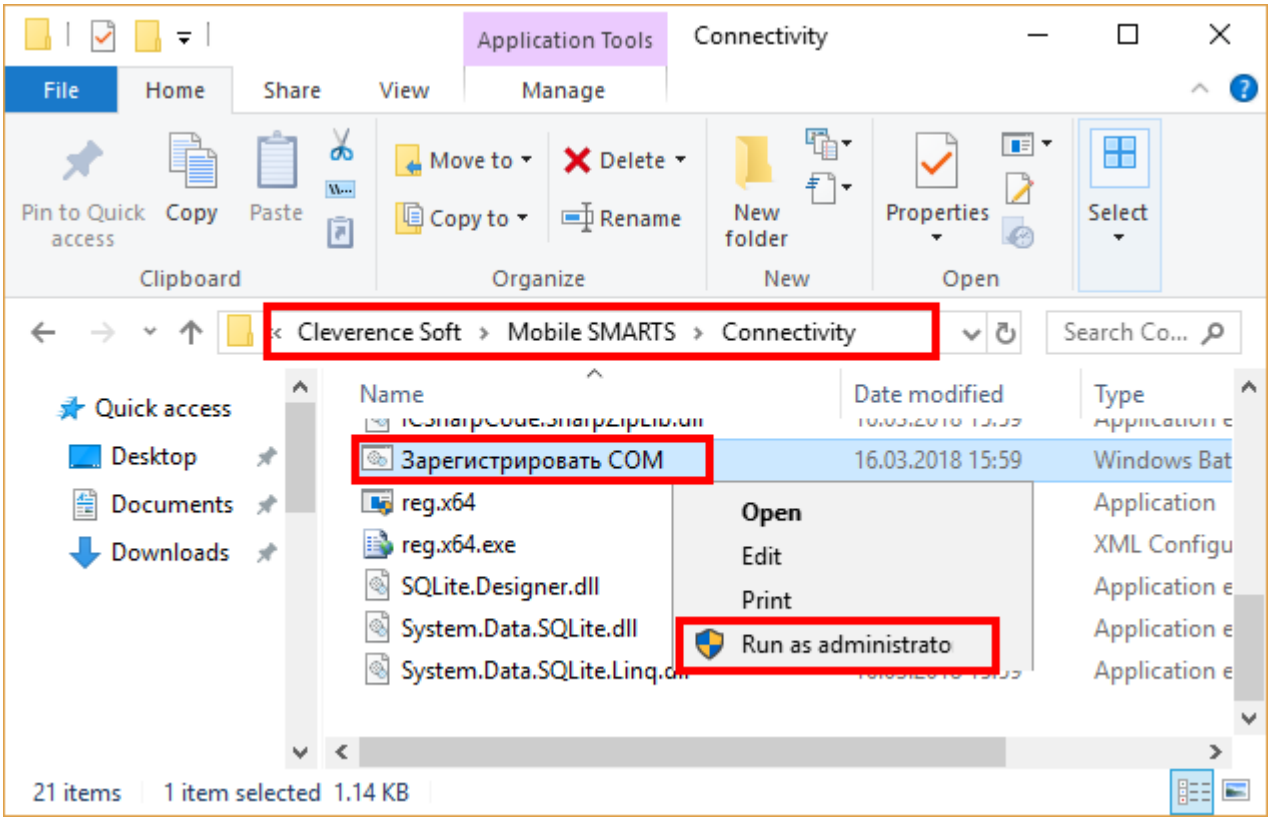
Процедура регистрации COM-объектов Mobile SMARTS

Последние изменения: 2024-03-26

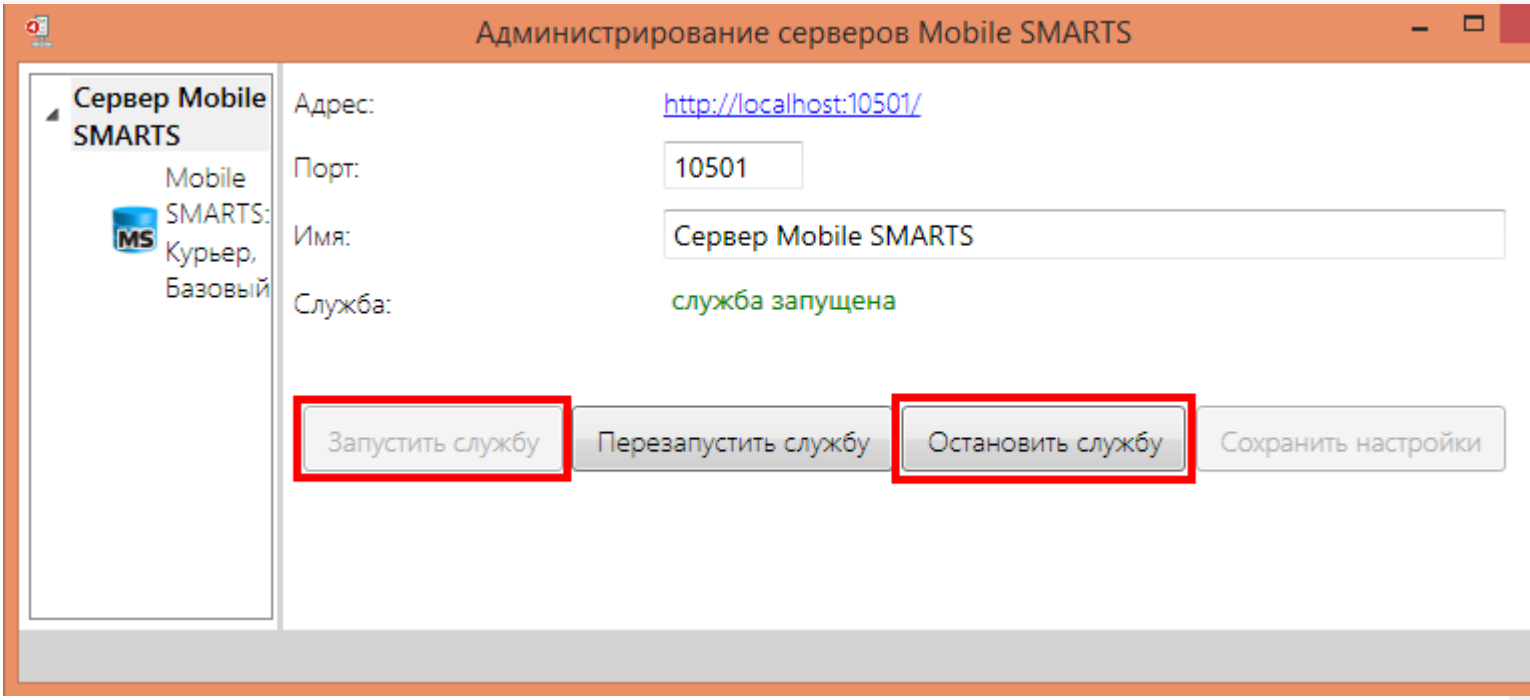
Если у вас не зарегистрированы COM объекты, то не будет обмена данными с учетной системой, например, 1С.

Для регистрации COM объекта необходимо запустить с правами администратора файл "Зарегистрировать COM.bat" из папки C:\Program Files (x86)\Cleverence Soft\Mobile SMARTS\Connectivity.

> **Запустите файл "Зарегистрировать COM.bat" от администратора именно правой кнопкой мыши!**



Перезапустите службу драйвера. Запустите Mobile SMARTS, нажмите "Настройки локального сервера", выберите ваш сервер. Затем нажмите на кнопку "Остановить службу", затем "Запустить службу".



COM-компонента

Не нашли что искали?

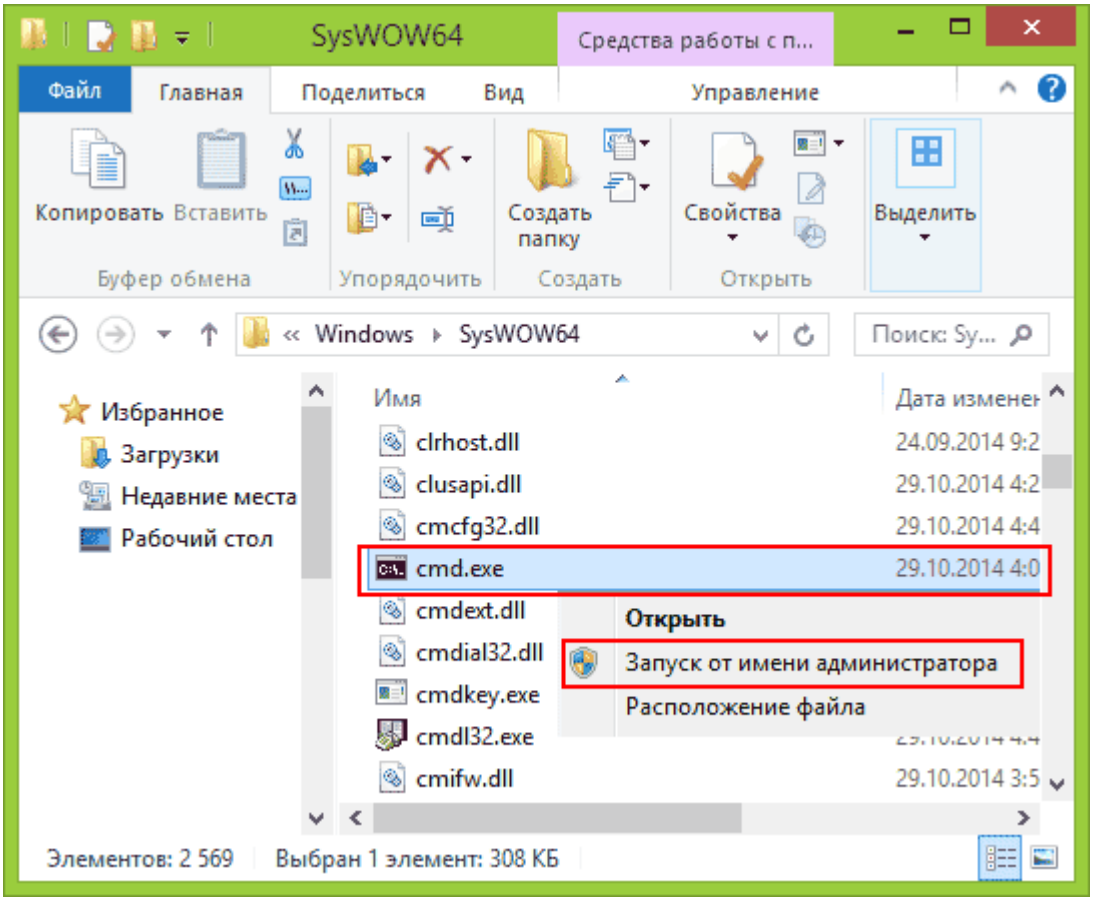
Задать вопрос в техническую поддержку

Процедура регистрации COM-объектов «1С: Предприятия»

Последние изменения: 2024-03-26

Для регистрации COM-объекта нам понадобится запустить несколько файлов из командной строки.

Командная строка — это обычная программа Windows (файл «cmd.exe») и запустить ее можно как любую другую программу. Находится она в папках «Windows/System32» (для 32-разрядных версий Windows) и «Windows/SysWOW64» (для 64-разрядных).



> Запустите файл «cmd.exe» от администратора именно **правой кнопкой мыши**!

1. Перед регистрацией лучше делать отмену регистрации. Запустите из командной строки команды: «1cv8.exe /unregserver» и «regsvr32 comcntr.dll /u».

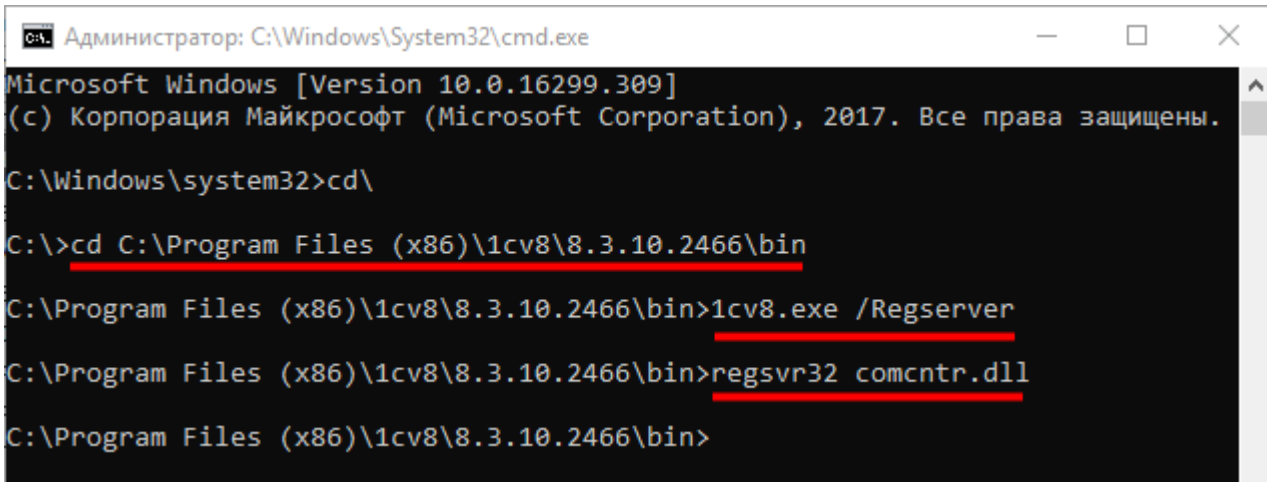
2. Запустите из командной строки команды: «1cv8.exe /regserver» и «regsvr32 comcntr.dll» из папки «1C\bin».

   > Если у вас используется 64-битная платформа 1С, то файлы «1cv8.exe» и «comcntr.dll» хранятся в папке вида «C:\Program Files\1cv82\8.2.19.83\bin». Если у вас 32-битная платформа 1С, то путь к папке будет иметь примерный вид «C:\Program Files (x86)\1cv82\8.2.19.83\bin».

3. Для этого в командной строке наберите: **cd пробел и путь к папке bin** (имя команды пока не указываем).
   Путь должен быть именно к той папке, которую использует платформа.
   В появившейся строке добавьте имя команды, которую нужно запустить.
   Команды запускаем по очереди (правильно напишите команды, включая пробелы и слеш, а лучше скопируйте и вставьте текст).

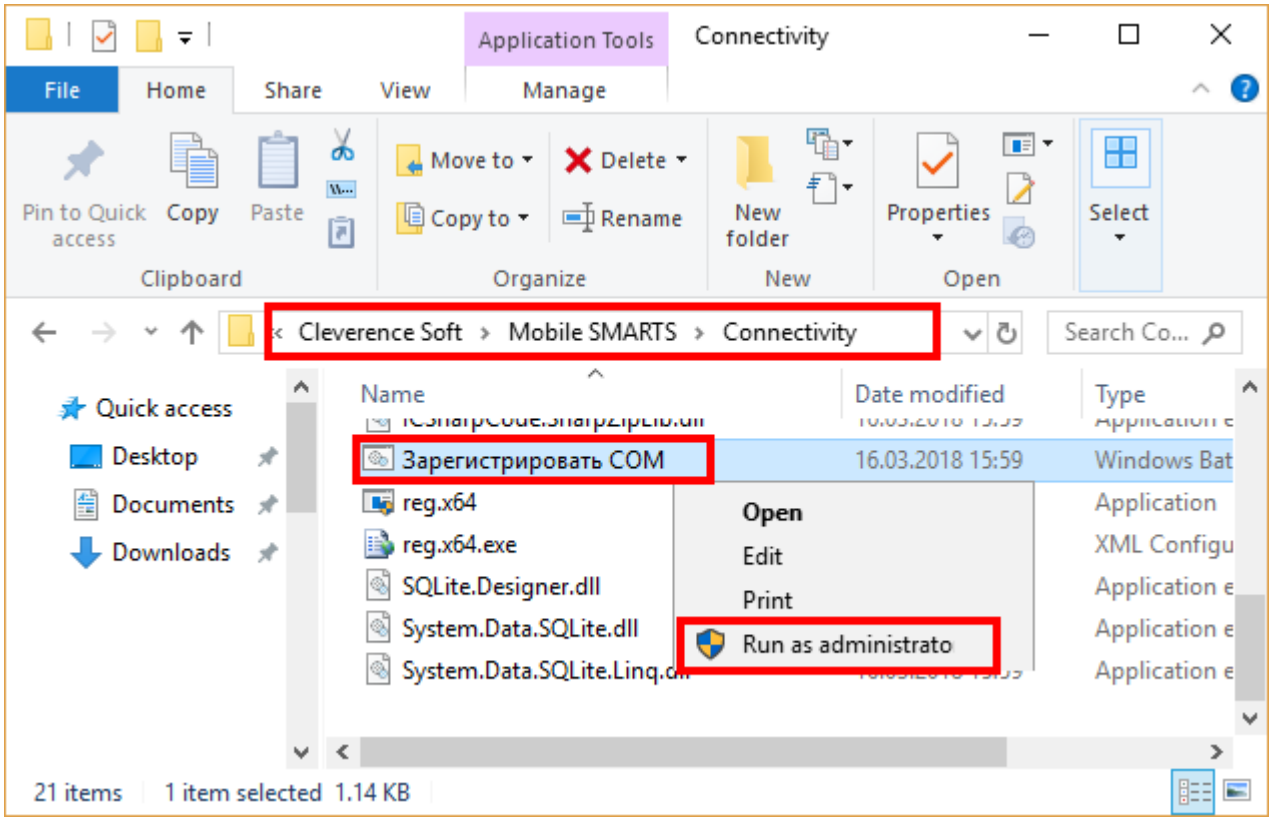
Если у вас платформа x32
Если у вас платформа x64

1cv8.exe /regserver regsvr32 comcntr.dll»
1cv8.exe /regserver %windir%/syswow64/regsvr32.exe

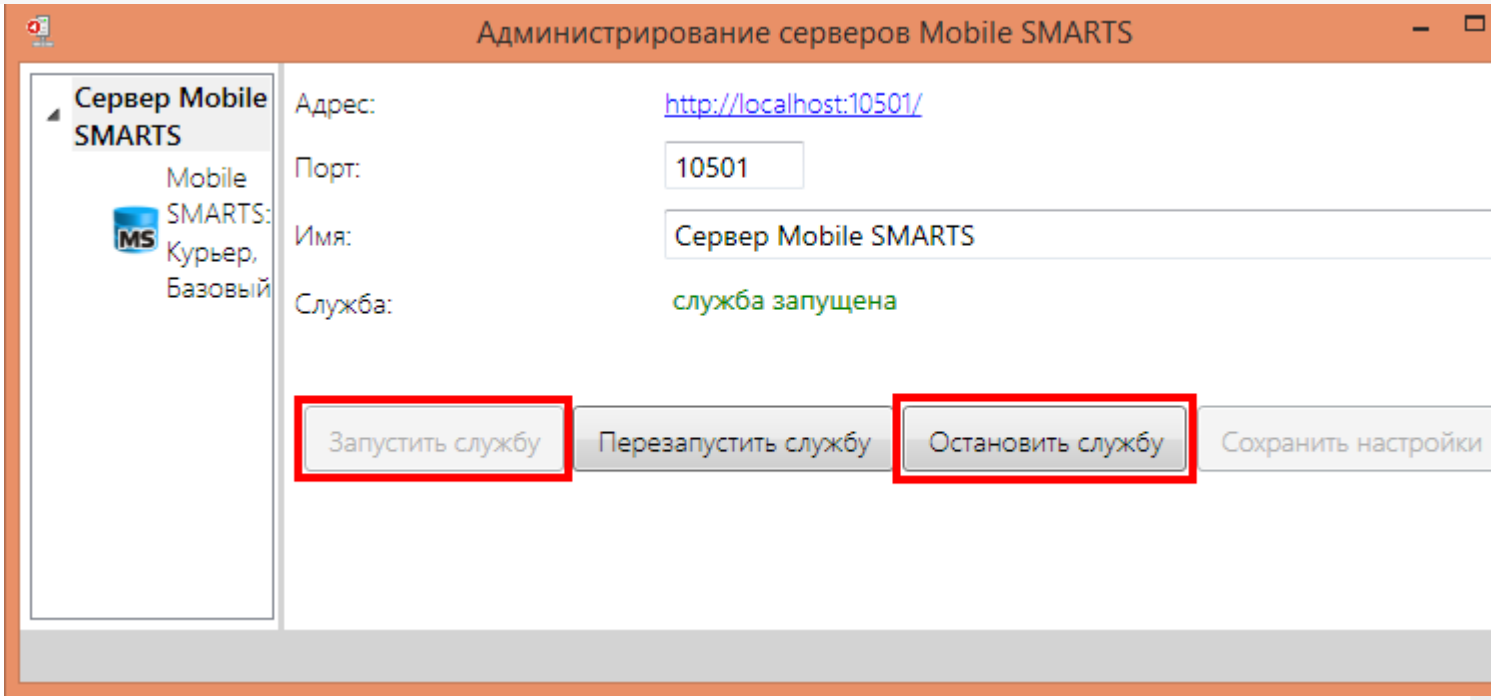


4. Запустите с правами администратора файл «Зарегистрировать COM.bat» из папки «C:\Program Files (x86)\Cleverence Soft\Mobile SMARTS\Connectivity».

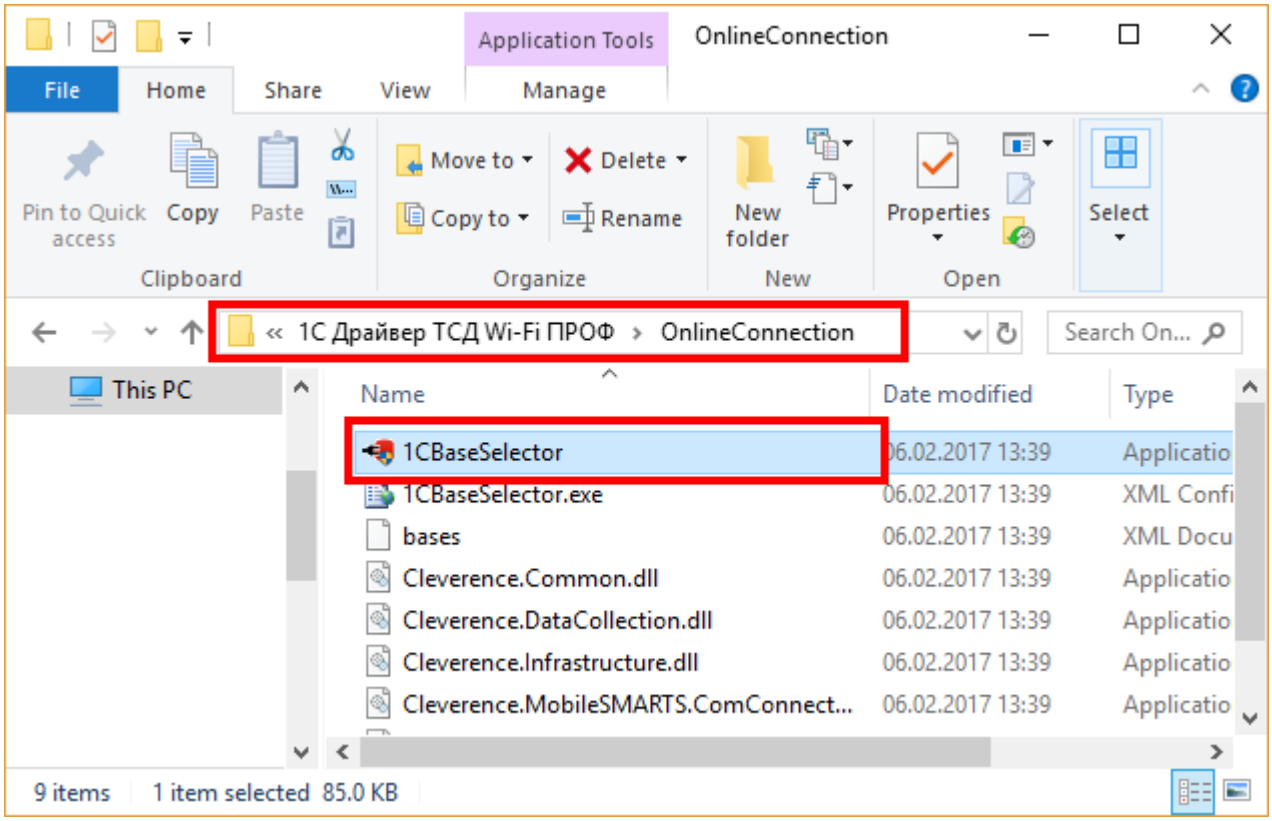
> Запустите файл «Зарегистрировать COM.bat» от администратора именно **правой кнопкой мыши**!



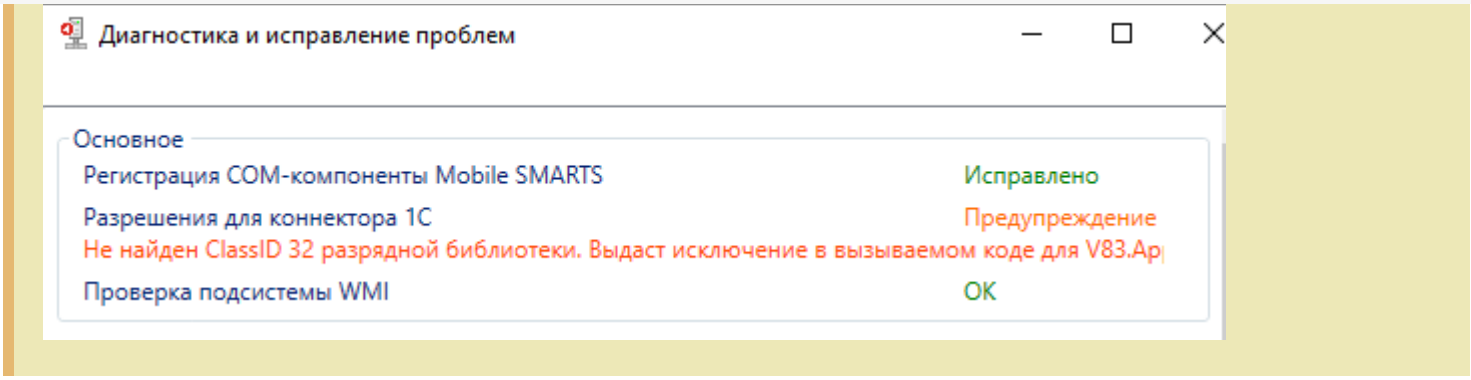
5. Перезапустите службу драйвера. Запустите Mobile SMARTS, нажмите «Настройки локального сервера», выберите ваш сервер. Затем нажмите на кнопку «Остановить сервер данных», затем «Запустить сервер данных».



6. **Данный пункт выполняется только для продукта 1С Драйвер ПРОФ: переподключите промежуточную базу «C:\ProgramData\Cleverence\Database\1С Драйвер ТСД Wi-Fi ПРОФ\OnlineConnection\1CBaseSelector.exe».**



В случае если при диагностике базы, для которой был зарегистрирован COM-объект, появится нижеуказанное сообщение, оно не будет являться ошибкой и никак не повлияет на работу базы. Это сообщение означает, что в вашем конкретном случае была зарегистрирована 64-разрядная библиотека 1С (может быть наоборот).



COM-компонента

Не нашли что искали?

Задать вопрос в техническую поддержку

Подключение к Mobile SMARTS через REST API

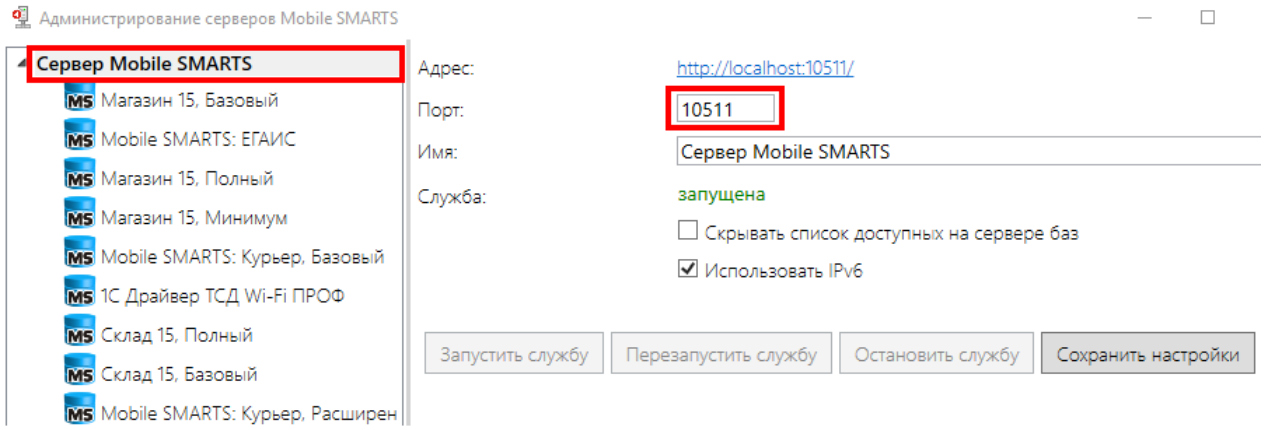
Последние изменения: 2024-03-26

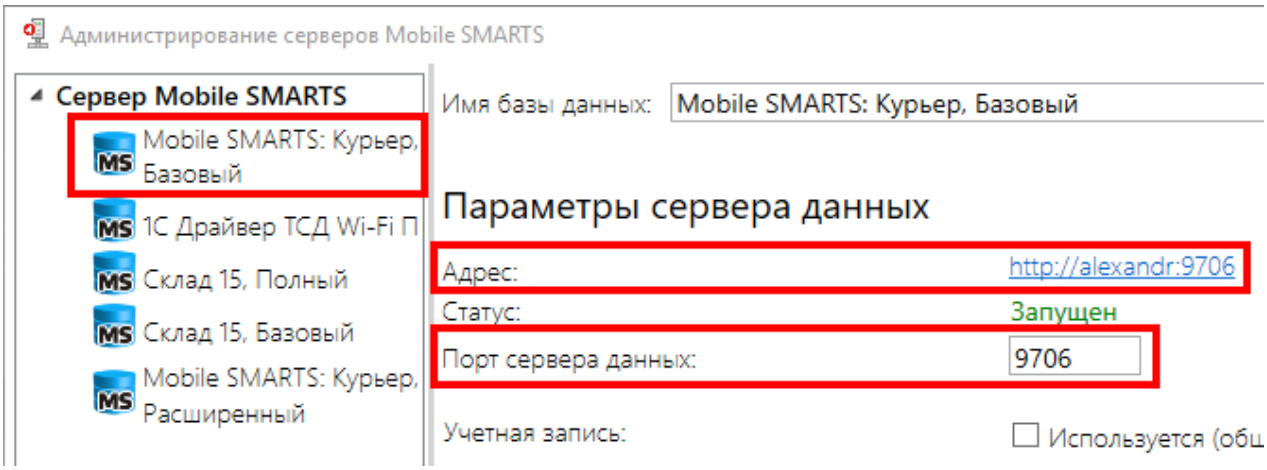
Точкой входа для доступа к API в нашей системе является url вида:

http(s)://{имя сервера}:{порт базы}/api/v1

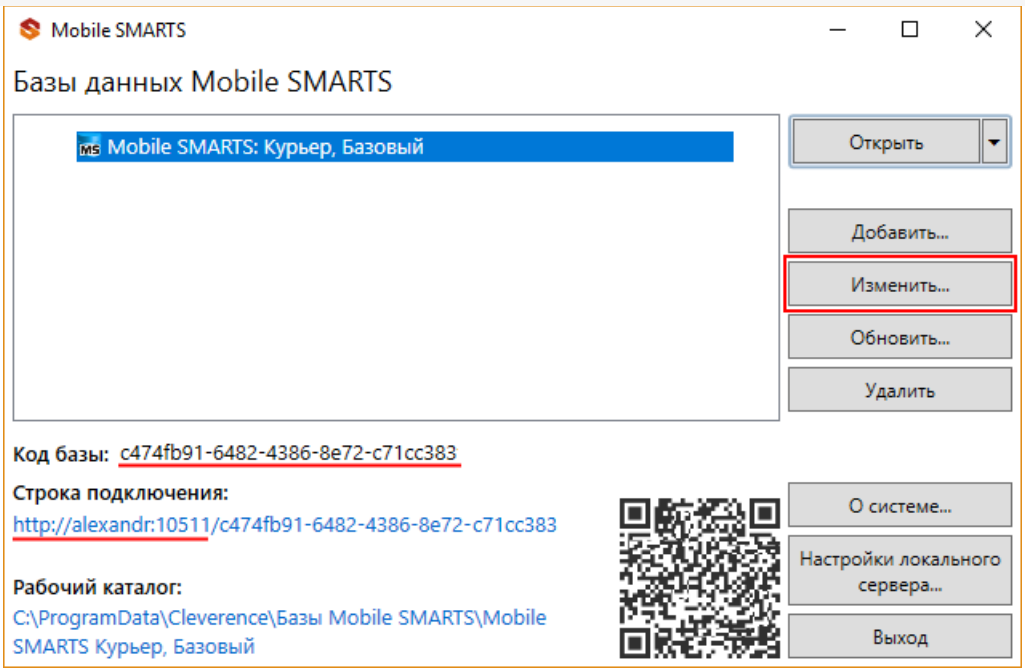
По нему можно получить список методов апи.

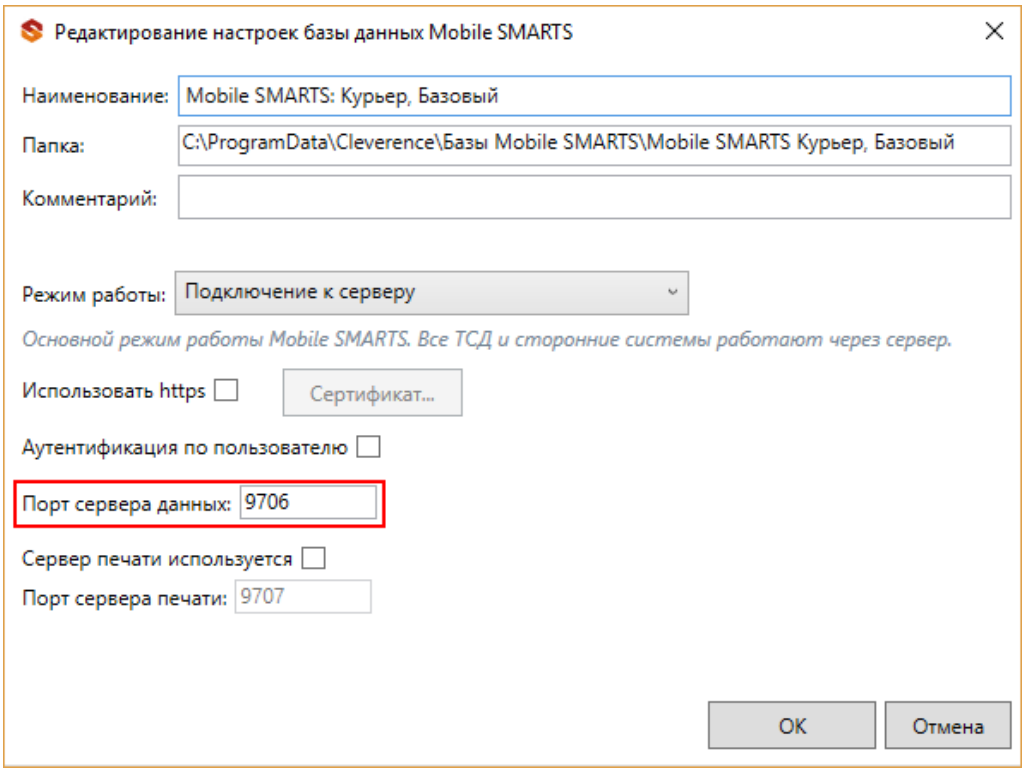
Чтобы узнать порт сервера, порт базы или идентификатор базы, надо воспользоваться приложением для администрирования сервера Mobile SMARTS.





Или менеджером баз данных Mobile SMARTS.





Для приведенных скриншотов url получаются такими:

http(s)://localhost:10511/c474fb91-6482-4386-8e72-c71cc383/api/v1

или

http(s)://localhost:9706/api/v1

интеграция, REST API

Не нашли что искали?

Задать вопрос в техническую поддержку

Проверка запросов через Swagger

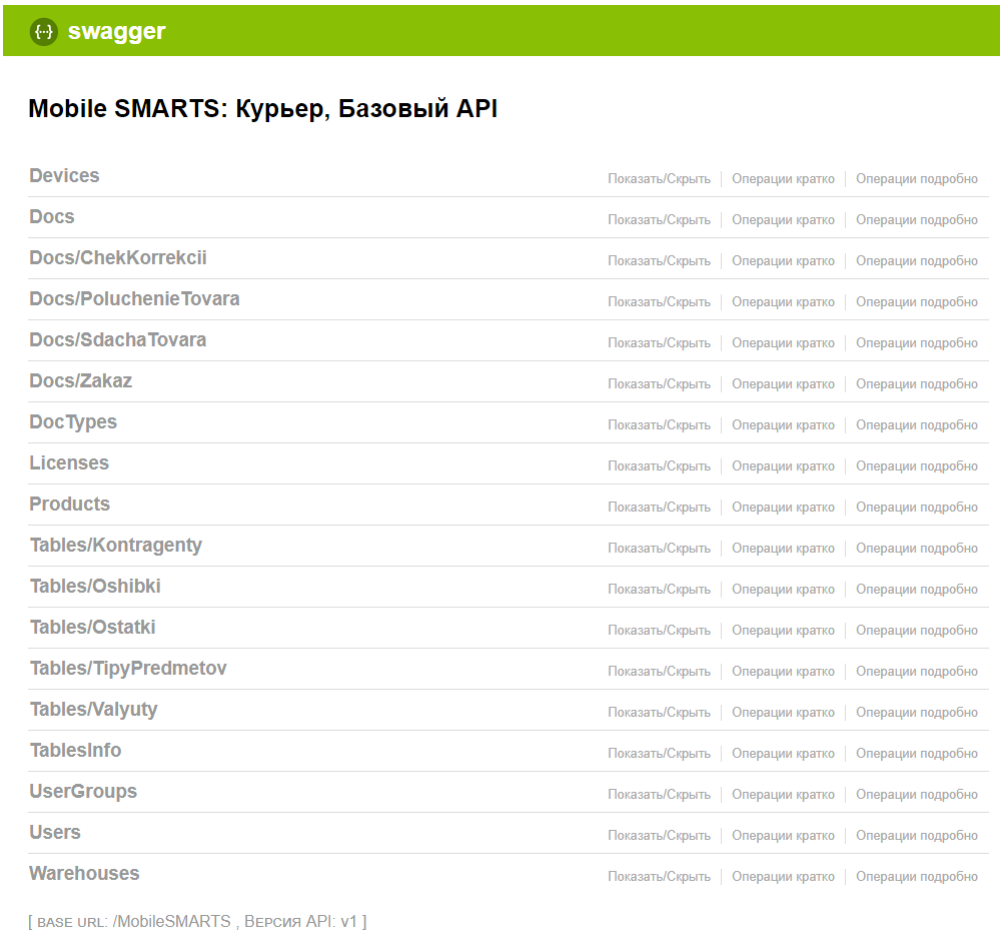
Последние изменения: 2024-03-26

> Примеры Swagger для Mobile SMARTS:
> Swagger для «Mobile SMARTS: Магазин 15»
> Swagger для «Mobile SMARTS: Курьер»

В нашей системе также предусмотрена возможность просматривать структуру API и выполнять простые запросы, используя Swagger. Для доступа к нему необходимо в браузере зайти по адресу:

http://xxx.xxx.xxx.xxx:9000/MobileSMARTS/swagger/ui/index

По этому адресу открывается страница с описанием всех методов.



Каждая группа в этом списке разворачивается и отображает все методы этой группы.

Mobile SMARTS: Курьер, Базовый API

Devices Показать/Скрыть | Операции кратко | Операции подробно

GET	/api/v1/Devices	Список устройств
POST	/api/v1/Devices	Добавить/отредактировать устройство
DELETE	/api/v1/Devices('{deviceId}')	Удалить устройство
GET	/api/v1/Devices('{deviceId}')	Получить устройство по идентификатору
PATCH	/api/v1/Devices('{deviceId}')	Изменить устройство
PUT	/api/v1/Devices('{deviceId}')	Добавить/отредактировать устройство по известному идентификатору

Docs Показать/Скрыть | Операции кратко | Операции подробно

Docs/ChekKorrekcii Показать/Скрыть | Операции кратко | Операции подробно

В каждом методе можно просматривать все входящие и исходящие параметры. Также можно просмотреть описание сущностей, которые отправляются/получаются при использовании этого метода.

Devices Показать/Скрыть | Операции кратко | Операции подробно

GET /api/v1/Devices — Список устройств

Пример ответа (Статус 200)

OK

Описание | Пример


```
{
  "@odata.context": "string",
  "value": [
    {
      "appInstanceId": "string",
      "deviceId": "string",
      "batteryStatus": "string",
      "lastInfoTime": "2018-03-20T06:49:48.982Z",
      "userId": "string",
      "warehouseId": "string",
```


Content Type ответа application/json

Параметры

Параметр	Значение	Описание	Тип параметра	Тип данных
$expand		Expands related entities inline.	query	string
$filter		Filters the results, based on a Boolean condition.	query	string
$select		Selects which properties to include in the response.	query	string
$orderby		Sorts the results.	query	string
$top		Returns only the first n results.	query	integer
$skip		Skips the first n results.	query	integer
$count		Includes a count of the matching results in the response.	query	boolean

Попробовать!

Также можно получить json схемы всех методов апи и по нему генерировать модели в своем приложении.

http://localhost:{порт базы}/swagger/docs/v1 - получение json сваггера.

интеграция, REST API, Swagger

Не нашли что искали?

Задать вопрос в техническую поддержку

Авторизация в системе через REST API

Последние изменения: 2024-03-26

Если в системе включена авторизация, то для начала работы с API необходимо пройти авторизацию в системе.

Реализованы несколько вариантов авторизации:

- BASIC авторизация.
- Авторизация методом GET по адресу /api/session с получением token.
- Авторизация методом POST по адресу /api/session с получением token.

REST API работает только если при работе с ТСД используется сервер Mobile SMARTS. При прямом подключении ТСД к компьютеру через кабель/крэдл или при обмене с учетной системой через папку использовать REST API не получится.

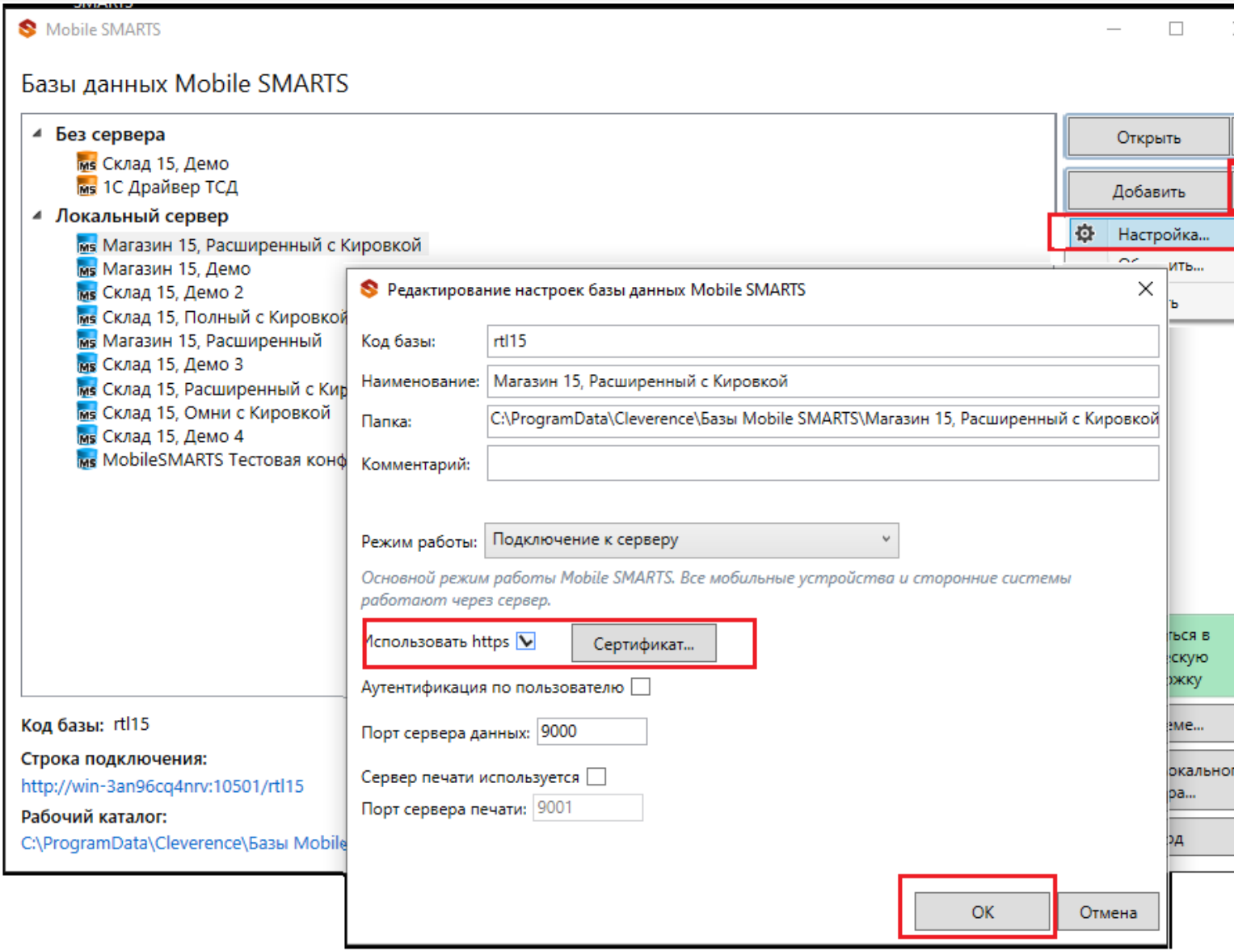
В базе Mobile SMARTS должна быть включена авторизация, тогда для выполнения HTTP-запросов необходимо будет каждый раз авторизовываться, есть 2 варианта авторизации:

- BASIC авторизация — в этом случае при каждом HTTP-запросе нужно будет отправлять логин и пароль;

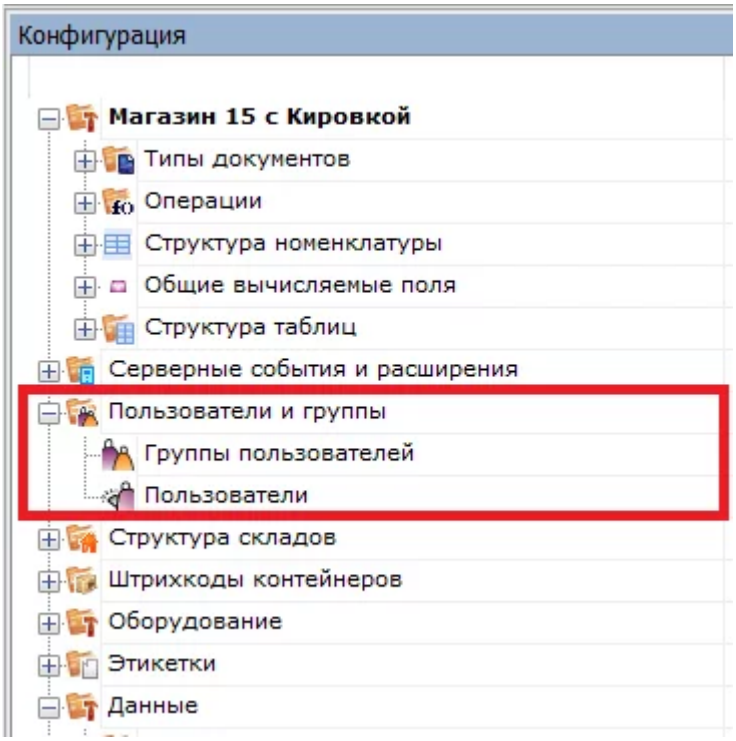
- авторизация с использованием токена — при первом HTTP-запросе передаются логин и пароль, а сервер MS возвращается токен — уникальный идентификатор сессии (access token), который при последующих HTTP-запросах можно будет использовать вместо передачи логина и пароля. Срок действия токена (т.е. сессии) ограничен, поэтому для обновления токена при первом запросе сервер Mobile SMARTS дополнительно возвращает «токен обновления» (refresh token) — он нужен для получения нового токена после истечения срока действия текущего токена.

Обратите внимание. Для защиты передаваемых данных рекомендуется использовать режим с включенной аутентификацией и доступом по https.

Для работы через протокол https необходимо указать это в настройках базы данных, для этого зайдите в менеджер базы, справа кнопки «Добавить» нажмите на выпадающий список и выберите пункт «Настройки», далее в окне «Редактирование настроек» поставьте галочку «Использовать http» и нажмите «ОК».



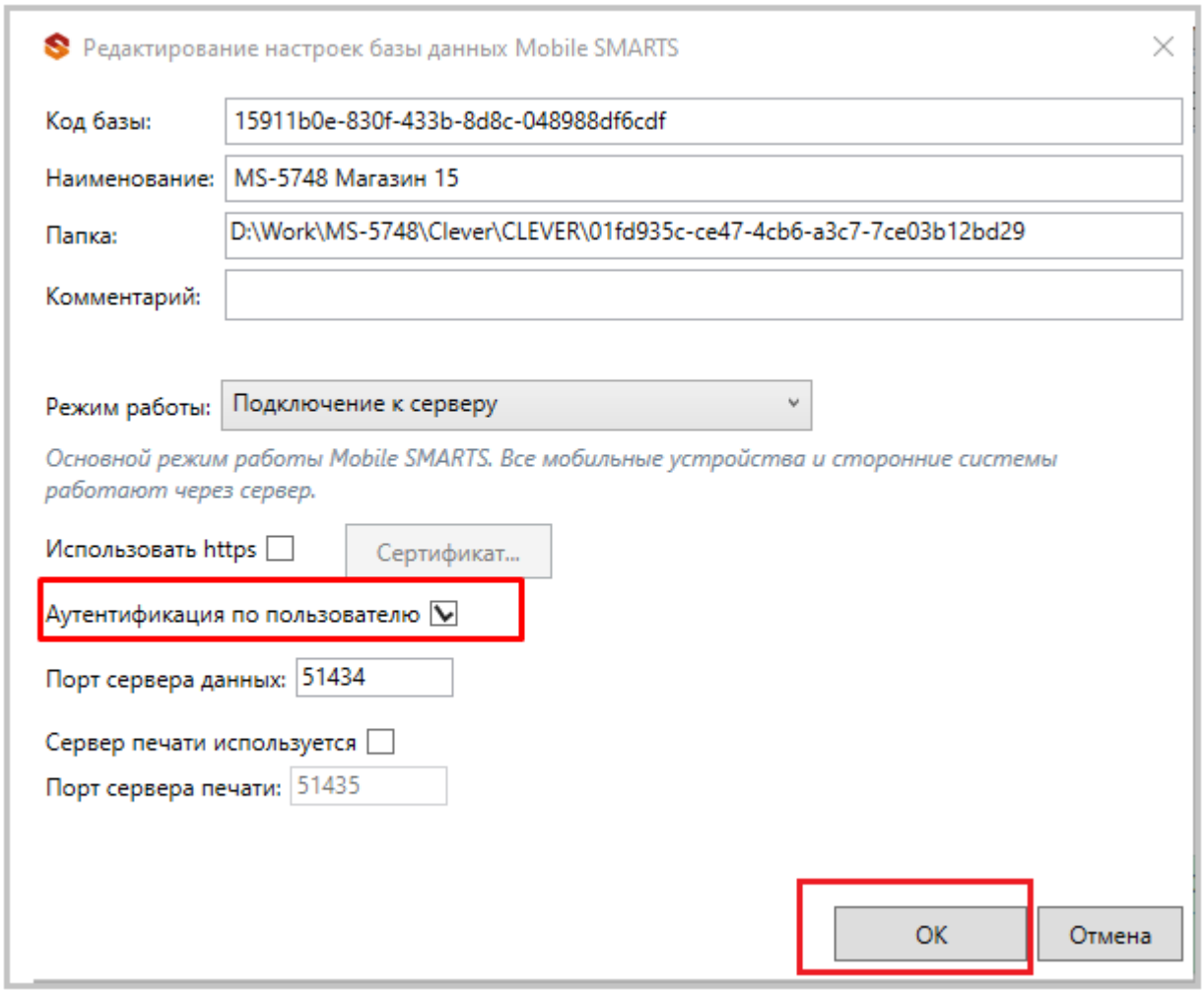
Для работы в режиме с включенной аутентификацией необходимо завести пользователей, задать им логины и пароли. В панели управления Mobile SMARTS узел «Пользователи и группы» содержит данные о пользователях и группах, в которых они состоят.



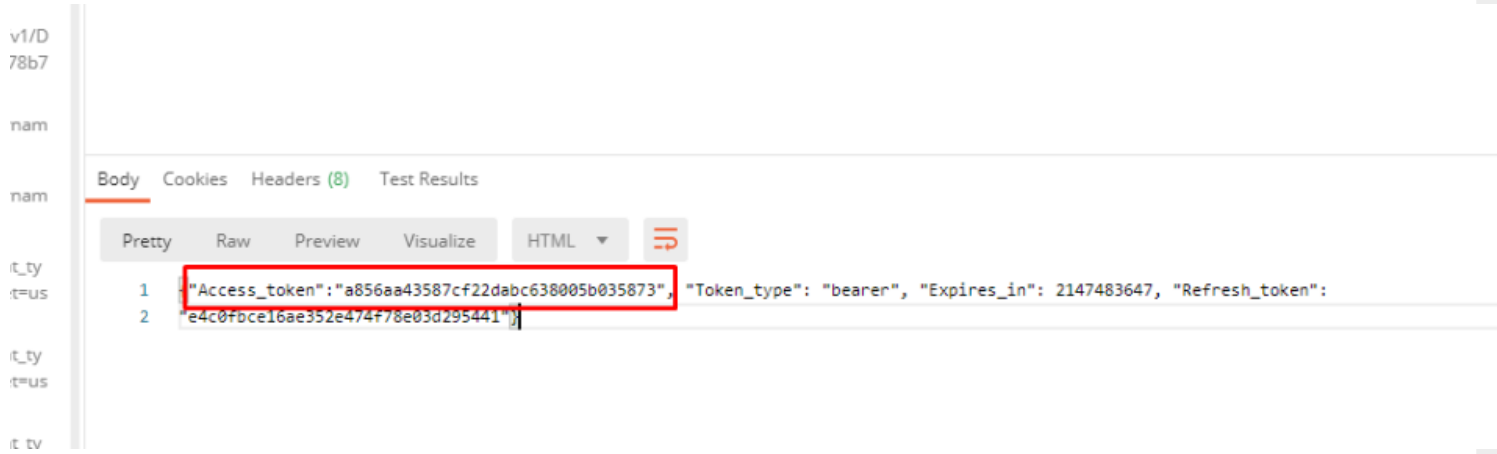
Инструкция по конфигурации настроек пользователей и групп пользователей, как определить роль пользователей и список типов документов, доступных для обработки пользователям такой группы описана в одноименной статье на сайте.

Пример запроса на авторизацию приложения методом POST

1. Для начала в настройках базы Mobile SMARTS включите авторизацию/аутентификацию по пользователю и нажмите на кнопку «ОК»



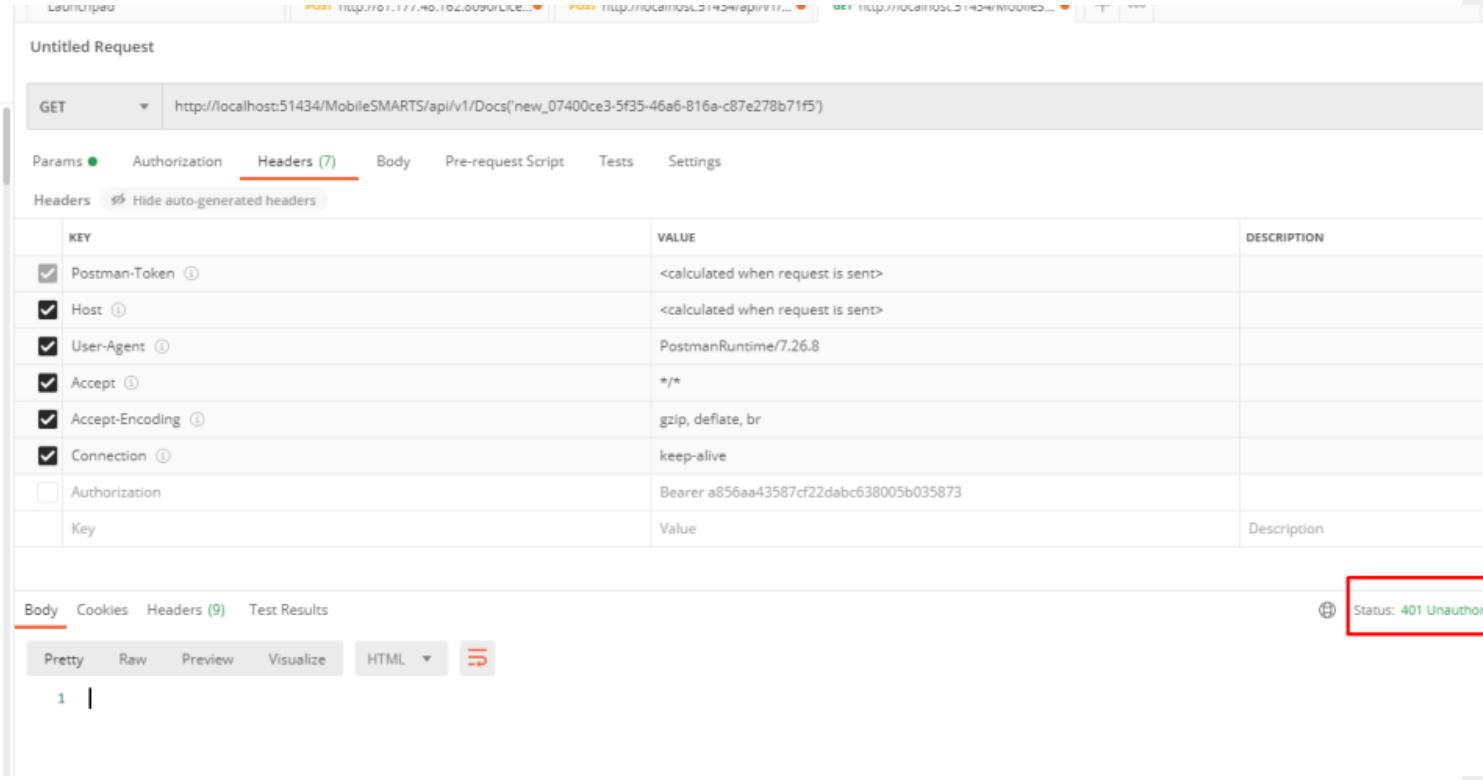
2. Пример запроса на авторизацию
http://localhost:51434/api/v1/session?username={username}&password={password}
в ответ приходит Access_token



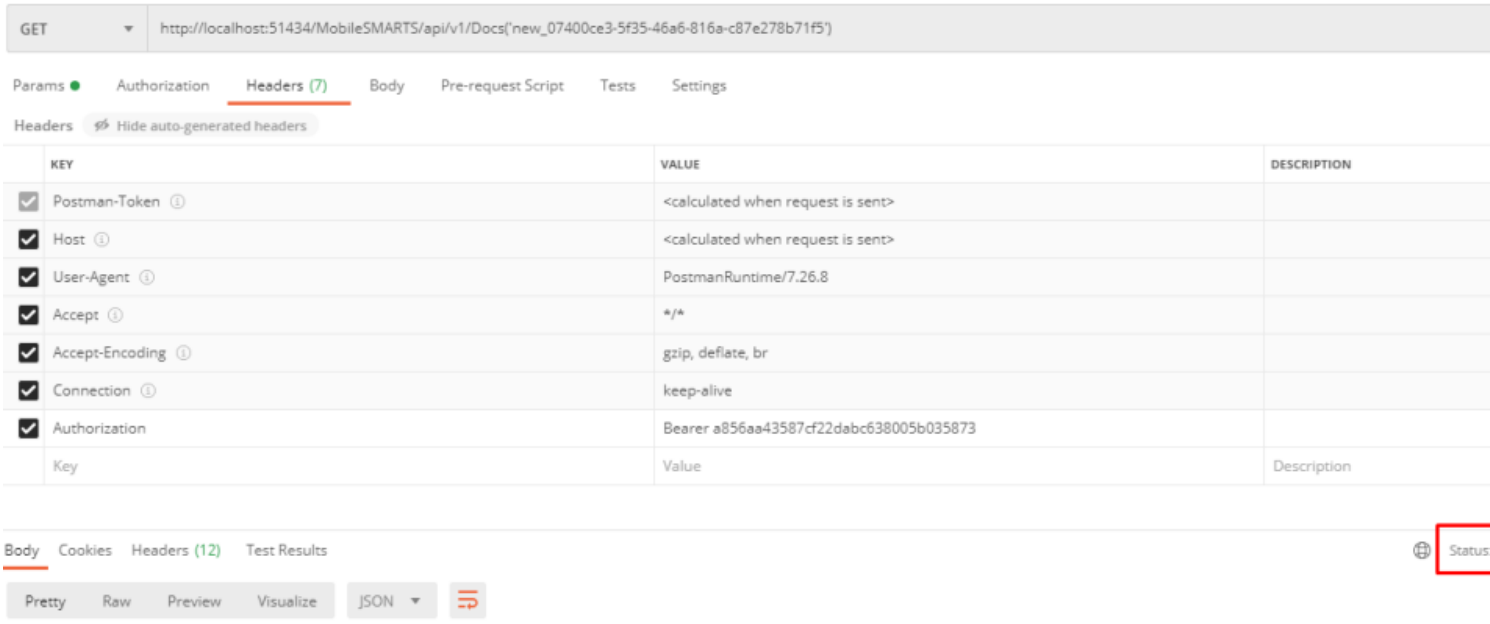
3. В последующих запросах в заголовке указываем

{Authorization: Bearer {Access_token}

Если не указать Authorization, в ответ приходит статус ошибки
401 — Unauthorized



4. При успешной авторизации приходит ответ и статус 202 — OK



BASIC авторизация

Данный вид авторизации чаще всего используется браузером для доступа к функциям API.

При использовании данного метода необходимо в заголовке каждого запроса указывать:

```
Authorization: Basic {login}:{password}
```

Допускается base64 при формировании строки {login}:{password}

Авторизация методом GET

Еще один способ авторизации — отправить GET запрос по адресу /api/v1/session, при этом в url запросе указать параметры login и password:

https://localhost:9000/api/v1/session?username=${Username}&password=${Password}

Ответ сервера:

```
{
Access_token:"123123123",
Token_type:"bearer",
Expires_in:86400,
Refresh_token:"321321321",
}
```

Авторизация методом POST

Авторизация по логину и паролю происходит путем отправки POST запроса на сервер, в результате которого возвращается access_token и refresh token в формате JSON.

Пример запроса:

POST /oauth/token HTTP/1.1
Host: mobilesmarts.ru/api/session
Content-Type: application/x-www-form-urlencoded

grant_type=password&client_id=123&client_secret=user&username=user@domain.ru&password=123456

Ответ сервера:

```
{
Access_token:"123123123",
Token_type:"bearer",
Expires_in:86400,
Refresh_token:"321321321",
}
```

Восстановление после окончания срока действия сессии

Восстановление после окончания срока действия сессии происходит путем отправки refresh_token на сервер, в результате приходит новый access_token и refresh_token

Пример:

POST /oauth/token HTTP/1.1

Host: mobilesmarts.ru/api/session
Content-Type: application/x-www-form-urlencoded

grant_type:refresh_token&client_id=123&client_secret=user&refresh_token=321321321

Ответ:

HTTP/1.1 200 OK
Content-Type: application/json

```
{
Access_token:"99999",
Token_type:"bearer",
Expires_in:86400,
Refresh_token:"789789789",
}
```

Вызов функций с использованием token

Для того чтобы обратиться к функциям (если не используется Basic авторизация), для которых необходима авторизация, необходимо в заголовке Authorization передавать токен:

Authorization: Bearer <token>

Иначе сервер вернет ошибку авторизации 401.

интеграция, REST API

Не нашли что искали?

Задать вопрос в техническую поддержку

Формат запросов в REST API

Последние изменения: 2024-03-26

Формат передачи входящих и исходящих данных — JSON, в кодировке UTF-8. Входящие параметры должны передаваться в теле POST запроса или в виде query-строки (?field=value) для GET запросов. Нужно учитывать, что GET запросы имеют ограничение на длину URL — 2048 символов.

В случае, если указан неверный адрес — возвращается ошибка 404.

В случае возникновения ошибки авторизации, возвращается ошибка 401.

Запросы строятся в формате OData, подробнее можно посмотреть по адресу

http://docs.oasis-open.org/odata/odata/v4.0/odata-v4.0-part2-url-conventions.html

Пагинация

Функции, возвращающие массив значений, поддерживают параметры для порционной загрузки, которые передаются в виде query-строки (?$top=1&$skip=10).

Пример:

https://localhost:9000/api/v1/Docs?$top=5&$skip=10

При выполнении данного запроса мы получим максимум 5 записей начиная с 11 индекса в списке.

Если в запросе на получение списка указать параметр count=true, в результате вернется дополнительно поле count, которое содержит общее число записей в списке на сервере.

Фильтрация

Для фильтрации данных в запросе можно указывать параметр $filter, например:

https://localhost:9000/api/v1/Docs?$filter=lastChangeDate gt 2017-10-06T17:41:10Z and documentTypeName eq 'Заказ'

При выполнении данного запросы мы получим список документов, дата изменения которых более указанной в запросе и тип документа — «Заказ».

Выборка только нужных свойств

Также можно указывать список полей, которые необходимо отобразить в результате, например:

https://localhost:9005/api/v1/Docs?$select=id,tables

Результатом выполнения данного запроса будет список документов, состоящих только из 2-х полей — id и tables.

Выборка записи по идентификатору

Для получения нужной записи необходимо составить запрос с указанием идентификатора этой записи, например:

https://localhost:9005/api/v1/Docs ('165')

Сортировка

Чтобы отсортировать результат необходимо указать параметр $orderby, например:

https://localhost:9005/api/v1/Docs?$orderby=id

В результате получим список документов, отсортированный по идентификатору.

Отображение подтаблиц и строк документов

Для документов реализован вывод записей строк документа и таблиц документа.

Пример:

Если в документе с идентификатором «1» существует таблица «ОплатыВозвраты», то получить строки таблицы можно отправив запрос:

https://localhost:9000/api/v1/Docs ('1')/ОплатыВозвраты

Для получения только строк документа с идентификатором «123» необходимо выполнить запрос

https://localhost:9000/api/v1/Docs ('123')/declaredItems это запрос плановой части документа

http://localhost:9000/api/v1/Docs ('123')?$expand=currentItems запрос фактической части документа

Развертывание сложных свойств

Чтобы отобразить все документы, включая его таблицы со строками, нужно сформировать запрос в таком виде:

https://localhost:9000/api/v1/Docs?$expand=tables ($expand=rows)

Поиск без учета регистра

https://localhost:9000/api/v1/Docs?$filter=Contains (tolower (name), 'фраза')

Запросы DELETE

Для запросов DELETE требуется параметр If-Match, нужно указывать *.

Добавление массива данных

В некоторых методах предусмотрено добавление массива

{ «value»:[{первый элемент}, {второй элемент}]}

интеграция, REST API

Не нашли что искали?

Задать вопрос в техническую поддержку

Примеры запросов в REST API

Последние изменения: 2024-03-26

Получение информации о базе

Для получения информации по текущей базе нужно выполнить запрос:

GET /api/v1/BaseInfo

Ответ:


```
{
"@odata.context": "http://172.19.0.30:9000/MobileSMARTS/api/v1/$metadata#BaseInfo",
"id": "rtl15",
"name": "Магазин 15, Базовый",
"folder": "C:ProgramDataCleverenceБазы Mobile SMARTSМагазин 15, Базовый",
"appId": "F42C7B5F-405C-4076-AE07-9348F189EE71",
"appName": "Магазин 15, Базовый",
"comment": null,
"allConnectionStrings": [
"https://172.19.0.30:10502/rtl15",
"Доступ к swagger:"
"https://172.19.0.30:10502/rtl15/swagger"
],
"appDescription": {
"appId": "F42C7B5F-405C-4076-AE07-9348F189EE71",
"appName": "Магазин 15, Базовый",
"desktopReqPayment": false,
"configId": "8DC2C8BA-77CE-46AD-93CB-6CFFCA7B96D7",
"serverModeSupported": true,
"onlineCallsSupported": false,
"batchModeSupported": true,
"appLink": "http://cleverence.ru/software/mobile-smarts/rtl15/",
"appSupportLink": "https://www.cleverence.ru/support/category:295/",
"aboutLicLink": "http://cleverence.ru/software/mobile-smarts/rtl15/#spec",
"appVersionsLink": "http://cleverence.ru/software/mobile-smarts/rtl15/#spec",
"comment": "Основная поставка Магазин 15",
"appVersion": "1.1.1.155",
"clientVersion": "3.0.0.100",
"panelVersion": "1.0",
"mainServerVersion": "1.0",
"appServerVersion": "3.0.0.2699",
"minPlatformVersion": "1.0",
"minPlatform35Version": "1.0"
},
"appInstanceSettings": {
"mode": "Server",
"hasServerAuth": false,
"serverSettings": {
```

```
"dataService": {
"enabled": true,
"port": 9000,
"useHttps": false,
"deviceAuth": false,
"passwordAuth": false
},
"printService": {
"enabled": false,
"port": 9001,
"useHttps": false,
"deviceAuth": false,
"passwordAuth": false
}
}
}
}
```


Группы пользователей

Работа с массивом
GET /api/v1/UserGroups


```
 {
"@odata.context": "string",
"value": [
{
"id": "string",
"name": "string",
"documentTypeNames": [
"string"
],
": true,
"role": 0,
"serverSideInventory": true,
"autorunDocumentTypeName": "string",
"onStartHandlerName": "string",
"onFinishHandlerName": "string"
}
]
 }
```


Работа с записями по идентификатору

POST /api/v1/UserGroups ('{key}')

PUT /api/v1/UserGroups ('{key}')

PATCH /api/v1/UserGroups ('{key}')

DELETE /api/v1/UserGroups ('{key}')

UserGroup {

id (string, *optional*): Id пользователя ,
name (string, *optional*): Уникальное наименование группы ,
documentTypeNames (Array[string], *optional*, read only),
batchMode (boolean, *optional*): Свойство, позволяющее задать тип обмена данными с сервером. Если true - обмен производится вручную, по запросу пользователем. Иначе, обмен будет производиться периодически, с интервалом, заданным в настройках клиентского приложения ,
role (integer, *optional*): Пользовательская роль = ['0', '1', '2', '3'],
serverSideInventory (boolean, *optional*): Указывает источник номенклатурного справочника. Если true - позиции номенклатуры будут искаться на сервере(необходимо наличие постоянной связи с сервером), иначе - справочник номенклатуры будет загружаться на терминал при операции обмена данными и использоваться локально ,
autorunDocumentTypeName (string, *optional*): Тип документа (виртуальный), который будет открыт автоматически, сразу после запуске программы на ТСД ,
onStartHandlerName (string, *optional*): Имя операции, запускающейся при входе пользователя данной группы ,
onFinishHandlerName (string, *optional*): Имя операции, запускающейся при выходе пользователя данной группы

}

Получение списка пользователей группы

GET /api/v1/UserGroups ('{key}')/users

Пользователи

Работа с массивом

GET /api/v1/Users

Работа с записями по идентификатору

POST /api/v1/Users

PUT /api/v1/Users

PATCH /api/v1/Users

DELETE /api/v1/Users

User {

id (string, *optional*): Id пользователя ,

name (string, *optional*): Имя пользователя ,

password (string, *optional*): (write only) Пароль пользователя ,

description (string, *optional*): Описание пользователя ,

barcode (string, *optional*): (write only) Штрихкод пользователя ,

groupId (string, *optional*): Имя группы в которой состоит пользователь. Подробнее смотрите UserGroup ,

groupName (string, *optional*): Для совместимости со старыми обработками ,

warehouseIds (Array[string], *optional*, read only)

}

Ячейки

GET /api/v1/Cells — Список ячеек

Работа с записями по идентификатору

POST /api/v1/Cells — Добавить/редактировать ячейку

DELETE /api/v1/Cells ('{id}') — Удалить ячейку

GET /api/v1/Cells ('{id}') — Получить ячейку по идентификатору

PUT /api/v1/Cells ('{id}') — Добавь/редактировать ячейку по идентификатору

Обновление справочника со сбросом всех записей

POST /api/v1/Cells/BeginOverwrite — начинает процедуру пакетной выгрузки ячеек на сервер. Все позиции будут накапливаться и не будут доступны до вызова функции EndOverwrite.

POST /api/v1/Cells/EndOverwrite — завершает процедуру пакетной выгрузки номенклатуры. Только после вызова этой функции сервер завершит обработку переданных позиций номенклатуры и они попадут в справочник товаров. Старый справочник товаров при этом будет полностью очищен.

Обновление записей

POST /api/v1/Cells/BeginUpdate — начинает процедуру пакетного обновления ячеек на сервере. Все позиции будут накапливаться и не будут доступны до вызова функции EndUpdate.

POST /api/v1/Cells/EndUpdate — завершает процедуру пакетного обновления ячеек. Только после вызова этой функции сервер завершит обработку переданных позиций номенклатуры и они попадут в справочник товаров. Выгруженная номенклатура будет слита с существующим на сервере справочником.

Сброс обновления/ перезаписывания

POST /api/v1/Cells/ResetUpdate — сбрасывает процедуру пакетного обновления ячеек.

Номенклатура

Получение схемы

Получение списка всех полей номенклатуры:

https://localhost:9000/api/v1/ProductSchema?$expand=allfields

Получение полей номенклатуры:

https://localhost:9000/api/v1/ProductSchema?$expand=fields

Получение списка фиксированных полей:

https://localhost:9000/api/v1/ProductSchema?$expand=defaultFields

Работа с массивом

GET /api/v1/Products — список номенклатуры.

Работа с записями по идентификатору

POST /api/v1/Products — добавить/ редактировать номенклатуру.

DELETE /api/v1/Products ('{id}') — удалить номенклатуру.

GET /api/v1/Products ('{id}') — получить номенклатуру по идентификатору.

PUT /api/v1/Products ('{id}') — добавить/ редактировать номенклатуру по идентификатору.

Обновление справочника со сбросом всех записей

POST /api/v1/Products/BeginOverwrite — начинает процедуру пакетной выгрузки номенклатуры на сервер. Все позиции будут накапливаться и не будут доступны до вызова функции EndOverwrite.

POST /api/v1/Products/EndOverwrite — завершает процедуру пакетной выгрузки номенклатуры. Только после вызова этой функции сервер завершит обработку переданных позиций номенклатуры и они попадут в справочник товаров. Старый справочник товаров при этом будет полностью очищен.

Обновление записей

POST /api/v1/Products/BeginUpdate — начинает процедуру пакетного обновления номенклатуры на сервере. Все позиции будут накапливаться и не будут доступны до вызова функции EndUpdate.

POST /api/v1/Products/EndUpdate — завершает процедуру пакетного обновления номенклатуры. Только после вызова этой функции сервер завершит обработку переданных позиций номенклатуры и они попадут в справочник товаров. Выгруженная номенклатура будет слита с существующим на сервере справочником.

Сброс обновления/ перезаписывания

POST /api/v1/Products/ResetUpdate — сбрасывает процедуру пакетного обновления номенклатуры.

Обновление номенклатуры таблицей значений

POST /api/v1/Products/BeginUploadProducts — начинает выгрузку позиций номенклатуры.

POST /api/v1/Products/AddProductToUpload — добавляет в выгрузку товаров товар с упаковкой.

POST /api/v1/Products/AddProductsToUpload — добавляет в выгрузку товаров товарs с упаковками.

POST /api/v1/Products/EndUploadProducts — завершает выгрузку товаров.

Таблицы

Работа с массивом

GET /api/v1/Tables/BiznesProcessy — получить все записи таблицы.

Работа с записями по идентификатору

POST /api/v1/Tables/BiznesProcessy — редактировать/ добавить запись.

DELETE /api/v1/Tables/BiznesProcessy ('{uid}') — удалить запись из таблицы.

GET /api/v1/Tables/BiznesProcessy ('{uid}') — получить запись по идентификатору.

PATCH /api/v1/Tables/BiznesProcessy ('{uid}') — редактировать запись.

PUT /api/v1/Tables/BiznesProcessy ('{uid}') — редактировать/добавить запись по известному идентификатору.

Обновление справочника со сбросом всех записей

POST /api/v1/Tables/BiznesProcessy/BeginOverwrite — начинает процедуру пакетной выгрузки строк таблицы на сервер. Все позиции будут накапливаться и не будут доступны до вызова функции EndOverwrite.

POST /api/v1/Tables/BiznesProcessy/EndOverwrite — завершает процедуру пакетной выгрузки строк таблицы. Только после вызова этой функции сервер завершит обработку переданных позиций и они попадут в таблицу. Старое содержимое при этом будет полностью очищено.

Обновление записей

POST /api/v1/Tables/BiznesProcessy/BeginUpdate — начинает процедуру пакетного обновления строк таблицы на сервере. Все передаваемые будут накапливаться и не будут доступны до вызова функции EndUpdate.

POST /api/v1/Tables/BiznesProcessy/EndUpdate — завершает процедуру пакетного обновления строк таблицы. Только после вызова этой функции сервер завершит обработку переданных позиций и они попадут таблицу. Выгруженные позиции будут слиты с существующей на сервере таблицей.

Сброс обновления\перезаписывания

POST /api/v1/Tables/BiznesProcessy/ResetUpdate — сбрасывает процедуру пакетного обновления строк таблицы.

Документы

Получение списка типов документов

GET /api/v1/DocTypes — список типов документов.

GET /api/v1/DocTypes ('{uni}') — получить тип документа по идентификатору.

Получение всех полей типа документа

https://localhost:9000/api/v1/DocTypes?$expand=allfields

Работа с массивом

GET /api/v1/Docs — список документов.

POST /api/v1/Docs редактировать/ добавить документ.

Работа с записями по идентификатору

DELETE /api/v1/Docs ('{id}') — удалить документ.

GET /api/v1/Docs ('{id}') — получить документ по идентификатору.

PATCH /api/v1/Docs ('{id}') — редактировать документ.

PUT /api/v1/Docs ('{id}') — редактировать/добавить документ по известному идентификатору.

Обновление записей

При любом редактировании документа, он сразу не сохраняется в систему. Сохранение происходит, если вызвать принудительное сохранение (EndUpdate), либо через 30 сек от последнего изменения.

POST /api/v1/Docs ('{id}')/EndUpdate — принудительно сохраняет документ, когда все строки уже загружены (не дожидаясь сохранения через 30 сек, как указано выше).

Получение строк документа

GET /api/v1/Docs ('{key}')/declaredItems — получить строк документа.

POST /api/v1/Docs ({key})/declaredItems — редактировать/добавить строку документа.

https://localhost:5001/api/v1/Docs ('{id}') ?$expand=currentItems — получить CurentItems строк документа по его ID.

Блокировка документа

POST /api/v1/Docs ('{id}')/Block — блокирует документ для совместной работы.

Тело запроса:

{"timeout»:1000}

, где timeout — время блокировки документа

POST /api/v1/Docs ('{id}')/Unblock — разблокирует документ для совместной работы.

Склады

Работа с массивом

GET /api/v1/Warehouses — получить список складов.

Работа с записями по идентификатору

POST /api/v1/Warehouses — добавление/ редактирование склада.

DELETE /api/v1/Warehouses ('{id}') — удаление склада.

GET /api/v1/Warehouses ('{id}') — получить конкретный склад.

PATCH /api/v1/Warehouses ('{id}') — редактирование существующего склада.

PUT /api/v1/Warehouses ('{id}') — добавление/ редактирование склада по существующему идентификатору.

Warehouse {

storageId (string, *optional*): Идентификатор склада, для хранения (не меняется) ,

id (string, *optional*): Уникальный идентификатор склада ,

name (string, *optional*): Наименование склада ,

cells (Array[Cell], *optional*, read only): Коллекция ячеек склада

}

Cell {

barcode (string, *optional*): Штрихкод ячейки. Может быть шаблонированным. Подробнее про применение шаблонов для ячеек, смотрите Руководство разработчика ,

id (string, *optional*): Id ячейки, искусственный ключ ,

name (string, *optional*): Наименование ячейки ,

description (string, *optional*): Описание ячейки

}

Получение информации о подключенных устройствах

GET /api/v1/Devices — список устройств.

DeviceInfo {

appInstanceId (string, *optional*),

deviceId (string, *optional*): Уникальный идентификатор терминала. Заполняется самим устройством при регистрации его в системе ,

batteryStatus (string, *optional*): Уровень заряда батарей ,

lastInfoTime (string, *optional*): Последнее время получения информации о терминале ,

userId (string, *optional*): Id пользователя, работавший с устройством в момент последнего получения информации о нем ,

warehouseId (string, *optional*): Идентификатор склада, на котором работает пользователь ,

documentId (string, *optional*): Идентификатор документа, с которым работает пользователь ,

cellId (string, *optional*): Идентификатор последней ячейки, с которой работал пользователь ,

deviceName (string, *optional*): Пользовательское имя устройства ,

documentTypeName (string, *optional*),

serverHostedDocument (boolean, *optional*),

deviceIP (string, *optional*),

userGroupId (string, *optional*)

}

Настройки

GET /api/v1/CustomSettings — список настроек

Результат:

```
{
"@odata.context": "http://localhost:9000/MobileSMARTS/api/v1/$metadata#CustomSettings&quot;,
"value": [
{
"name": "testProp",
"value": "testValue"
}
]
 }
```

POST /api/v1/CustomSettings — Добавить/отредактировать настройку

Тело запроса:

```
{
"name": "testProp",
"value": "testValue"
}
```

Также можно добавлять массив настроек:

```
 {
"value":[
 {
"name": "testProp",
"value": "testValue"
 },
 {
"name": "testProp1",
"value": "testValue1"
 }
]
 }
```

GET /api/v1/CustomSettings ('{name}') — получить настройку

Тело ответа:

```
{
"@odata.context":
"http://localhost:9000/MobileSMARTS/api/v1/$metadata#CustomSettings/$entity&quot;,
"name": "testProp",
"value": "testValue"
 }
```

PUT /api/v1/CustomSettings ('{name}') — Добавить/отредактировать настройку по известному идентификатор.

Тело запроса:

```
{
"name": "testProp",
"value": "testValue"
 }
```

DELETE /api/v1/CustomSettings ('{name}') — удалить настройку

Сообщения

Действия с сообщениями:

GET /api/v1/Messages — список сообщений.

POST /api/v1/Messages — добавить/отредактировать сообщение.

GET /api/v1/Messages ('{id}') — получить сообщение по идентификатору.

MessageInfo {

appInstanceId (string, *optional*),
deviceId (string, *optional*): Уникальный идентификатор терминала. Заполняется самим устройством при регистрации его в системе ,
batteryStatus (string, *optional*): Уровень заряда батарей ,
lastInfoTime (string, *optional*): Последнее время получения информации о терминале ,
userId (string, *optional*): Id пользователя, работавший с устройством в момент последнего получения информации о нем ,
warehouseId (string, *optional*): Идентификатор склада, на котором работает пользователь ,
documentId (string, *optional*): Идентификатор документа, с которым работает пользователь ,
cellId (string, *optional*): Идентификатор последней ячейки, с которой работал пользователь ,
deviceName (string, *optional*): Пользовательское имя устройства ,
documentTypeName (string, *optional*),
serverHostedDocument (boolean, *optional*),
deviceIP (string, *optional*),
userGroupId (string, *optional*)

}

REST API

Не нашли что искали?

Задать вопрос в техническую поддержку

Особенности загрузки данных через REST API

Последние изменения: 2024-03-26

Отложенное внесение изменений

Обновление данных с помощью REST API всегда имеет отложенное срабатывание. Это значит, что большинство справочников (номенклатура, таблицы) и документы в базе данных на сервере обновляются не мгновенно, а с задержкой (30 секунд).

Для чего это делается:

- Для предотвращения мгновенной отправки измененного справочника на мобильное устройство, так как требуется время, чтобы успеть обновить его версию на мобильном устройстве.
- Для того, чтобы избежать генерации лишнего промежуточного трафика из-за порционности POST-запросов.
- Относительно справочника номенклатуры это решает проблему лишней нагрузки на сервер по формированию справочника и его индексов, так как данный справочник имеет специфический внутренний формат (используемый для ТСД на Windows CE).

Обновление документа

При загрузке большого числа строк в документе включается режим обновления (BeginUpdate), который завершится автоматически через 30 секунд и документ сохранится на диск. В режиме обновления документ не сохраняется при добавлении каждой строки. Если после загрузки последней строки требуется сразу записать документ на диск, необходимо выполнить запрос EndUpdate. Такая схема предотвращает выполнения события на сервере до завершения загрузки всех строк документа.

Пример:

/api/v1/Docs/ChekKorrekcii ('{id}')/EndUpdate

Обновление таблицы и справочника номенклатуры

При загрузке большого числа строк в какую-либо таблицу системы необходимо включить режим обновления таблицы. Существуют 2 режима обновления — добавление новых записей и полная перезапись таблицы. При перезаписи старые данные заменяются новыми, т. е. таблица очистится и в нее добавятся новые записи.

Для перезаписи необходимо вызвать запросы:

/api/v1/Tables/Kontragenty/BeginOverwrite — включить режим перезаписи для таблицы Kontragenty

{Далее необходимо отправить все новые записи}

/api/v1/Tables/Kontragenty/EndOverwrite — завершить режим перезаписи и заменить предыдущую таблицу Kontragenty

Для обновления записей таблицы необходимо вызвать запросы:

/api/v1/Tables/Kontragenty/BeginUpdate — начать обновление таблицы

{Далее необходимо отправить все новые записи}

/api/v1/Tables/Kontragenty/EndUpdate

Для справочника номенклатуры запросы формируются следующим образом:

/api/v1/Products/BeginOverwrite

/api/v1/Products/EndOverwrite

/api/v1/Products/BeginUpdate

/api/v1/Products/EndUpdate

Пакетная загрузка номенклатуры на сервер REST API

Пакетную загрузку номенклатуры можно реализовать следующим способом:

POST /api/v1/Products/ResetUpdate — сбрасываем процедуру пакетного обновления номенклатуры на сервере

Тело сообщения:

```
{}
```

POST /api/v1/Products/BeginUpdate — начинаем процедуру пакетного обновления номенклатуры на сервере

Тело сообщения:

```
{}
```

POST /api/v1/Products — Добавляем номенклатуру на сервер двумя пакетами

Пакет 1. Тело сообщения:

```
{
 "value": [
  {
   "id": "1f738be6-0433-4a85-866f-3479f7ec1eda",
   "name": "Товар1",
   "barcode": "1001",
   "basePackingId": "шт",
   "packings": [
    {
     "id": "шт"
    }
   ]
  },
  {
   "id": "2f738be6-0433-4a85-866f-3479f7ec1eda ",
   "name": "Товар2",
   "barcode": "1002",
   "basePackingId": "шт",
   "packings": [
    {
     "id": "шт"
    }
   ]
  }
 ]
}
```


Пакет 2. Тело сообщения:

```
{
 "value": [
  {
   "id": "3f738be6-0433-4a85-866f-3479f7ec1eda ",
   "name": "Товар3",
   "barcode": "1003",
   "basePackingId": "шт",
   "packings": [
    {
     "id": "шт"
    }
   ]
  },
  {
   "id": "4f738be6-0433-4a85-866f-3479f7ec1eda ",
   "name": "Товар4",
   "barcode": "1004",
   "basePackingId": "шт",
   "packings": [
    {
     "id": "шт"
    }
   ]
  }
 ]
}
```


Для тестовых сообщений сформированы сообщения в формате json по базовым полям в количестве 2 единиц в одном пакете

POST /api/v1/Products/EndUpdate — завершаем процедуру пакетного обновления номенклатуры на сервере

Тело сообщения:

```
{}
```

интеграция, REST API

Не нашли что искали?

Задать вопрос в техническую поддержку

Расширение API через коннектор

Последние изменения: 2024-03-26

Есть возможность расширить API, создав свой коннектор в серверной части платформы.

Более подробно о создании коннектора можно посмотреть здесь.

Для этого при создании коннектор нужно унаследовать и от интерфейса IApiExtenderPlugin, например:

```
public class ApiExtenderConnector : IConnector, IApiExtenderPlugin
{
public IList<IApiGroup> ApiGroup { get; private set;}
}
```

Свойство ApiGroup возвращает массив классов, унаследованных от IApiGroup, которые будут обрабатывать все запросы расширения из внешней системы.

```
public interface IApiGroup
{
string Id { get; }
object CreateNewEntity();
string GetEntityIdFieldName();
IEnumerable<object> GetList();
object Get(string id);
object Post(object input);
object Put(string id, object input);
void Delete(string id);
}
```

Id — идентификатор, который должен быть уникальным в системе. Он может содержать как буквы, так и цифры. Для обращения к этому расширению в url строку нужно будет вставить этот идентификатор, например:

Создаем класс коннектора:

```
 public class ApiExtenderConnector : IConnector, IApiExtenderPlugin
{
public ApiExtenderConnector() {
ApiGroup = new List<IApiGroup>();
ApiGroup.Add(new ApiTestGroup());
}
public bool IsSelfTimeoutBehavior { get { return false; } }
private string id;
[System.ComponentModel.Description("Идентификатор.")]
public string Id
{
```

```
get { return this.id; }
set { this.id = value; }
}
bool initialized = false;
public bool Initialized
{
get
{
return this.initialized;
}
}
private int timeout = 0;
public int Timeout
{
get { return this.timeout; }
set { this.timeout = value; }
}
private bool enabled = true;
public bool Enabled
{
get
{
return this.enabled;
}
set
{
this.enabled = value;
}
}
private TimeoutBehavaior timeoutBehavaior = TimeoutBehavaior.ThrowException;
public TimeoutBehavaior TimeoutBehavaior
{
get { return this.timeoutBehavaior; }
set { this.timeoutBehavaior = value; }
}
public bool IsSupportDeviceInfoInArgs
{get { return false; }
}
[Cleverence.DataCollection.Xml.XmlSerializable(Cleverence.DataCollection.Xml.XmlSerializationType.
None)]
public IList<IApiGroup> ApiGroup { get; private set;}
public void CheckLicenseLimitations(List<LicenseExternalSystem> supportedSystems)
{
}
public void Initialize()
{
}
public object InvokeMethod(string methodName, object[] args)
{
return null;
}
}
```

Далее необходимо откомпилировать и положить готовую библиотеку в папку сервера. В результате после удачного запуска сервера проверяем ответ сервера по пути

https://localhost:<порт базы>/api/v1/

В ответе должна содержаться группа

{"name»:"Plugins/ApiTestGroup»,"kind»:"EntitySet»,"url»:"Plugins/ApiTestGroup"}

Далее при переходе по пути

https://localhost:<порт базы>/api/v1/Plugins/ApiTestGroup

получаем список наших сущностей (результат выполнения функции GetList ())


```
 {
 "@odata.context":"https://localhost:9005/MobileSMARTS/api/v1/$metadata#Plugins&quot;,
 "value":[
 {"id":"1","name":"1111","valueInt":0,"valueBool":false},
 {"id":"2","name":"2223","valueInt":0,"valueBool":false},
 {"id":"3","name":"3333","valueInt":0,"valueBool":false}]
}
```

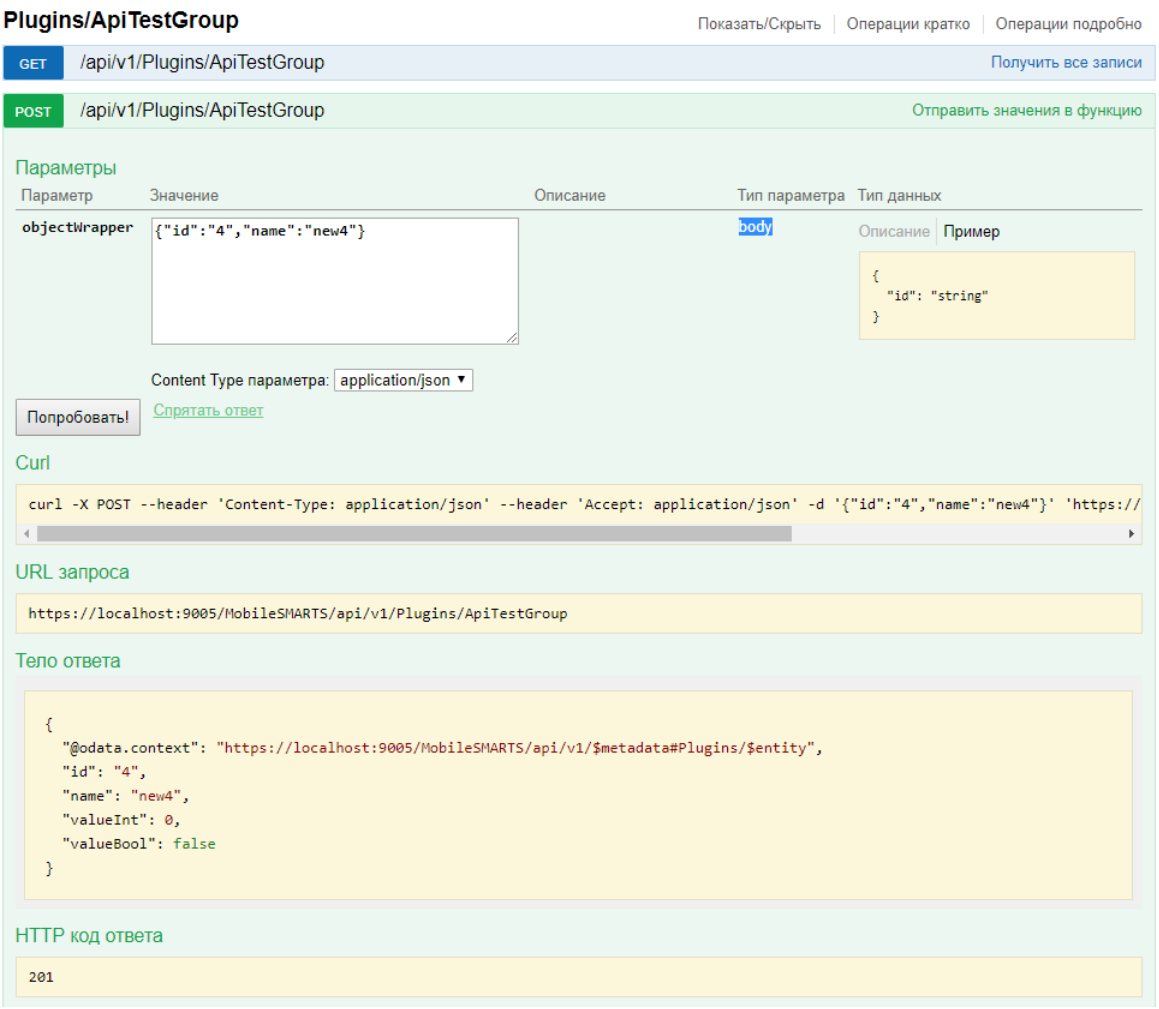

Далее POST запрос можем проверить в swagger. Для этого заходим

https://localhost:<порт базы>/swagger

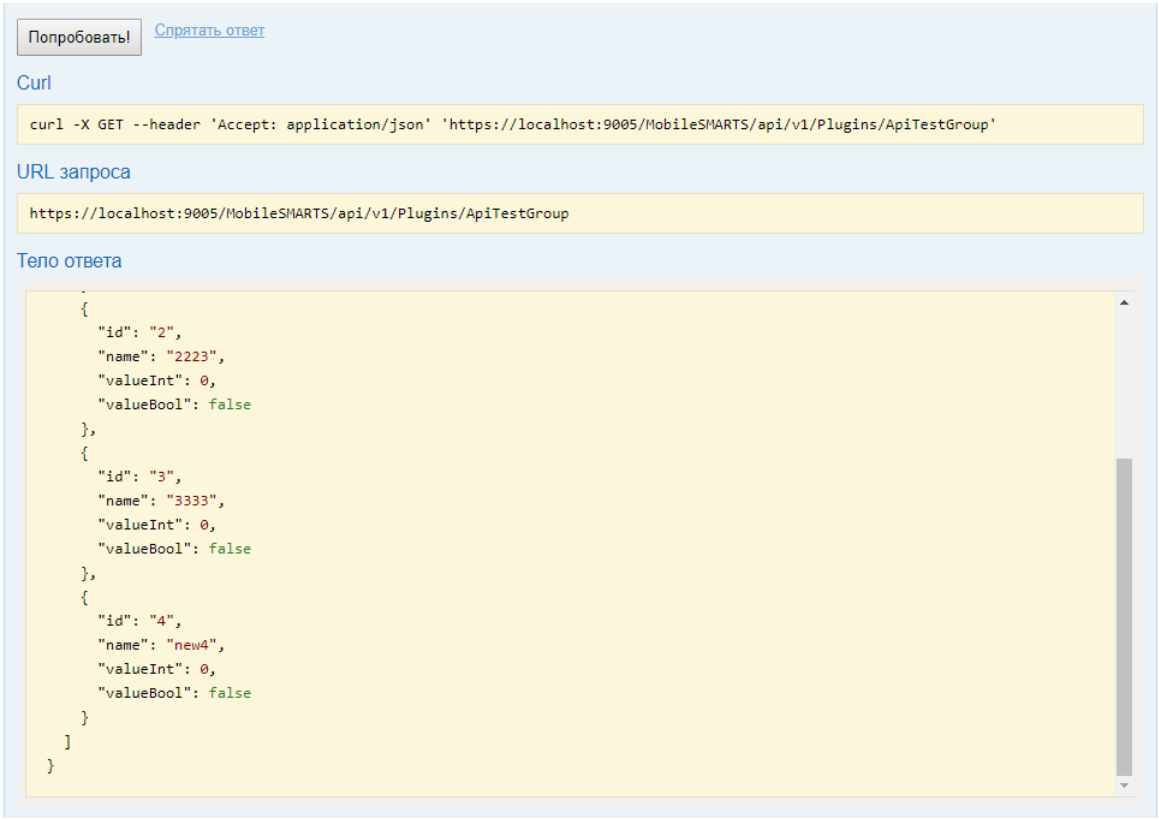
Находим там нашу группу Plugins/ApiTestGroup, выбираем POST, вставляем в параметр body:

{"id»:"4»,"name»:"new4"}

Нажимаем кнопку «Попробовать!»



Далее заходим в блок GET и видим что наша новая сущность сохранилась:



Для удаления заходим в блок DELETE, вводим наш новый идентификатор 4 и выполняем.

DELETE /api/v1/Plugins/ApiTestGroup('{id}') Запрос на удаление значений

Параметры

Параметр	Значение	Описание	Тип параметра	Тип данных
id	4	key: id	path	string
If-Match	*	If-Match header	header	string

Что может прийти в ответ

HTTP код	Причина	Структура ответа	Заголовки
204	NoContent		

Попробовать!

Проверяем в блоке GET и видим, что новая сущность исчезла.

Попробовать! Спрятать ответ

Curl

```
curl -X GET --header 'Accept: application/json' 'https://localhost:9005/MobileSMARTS/api/v1/Plugins/ApiTestGroup'
```

URL запроса

```
https://localhost:9005/MobileSMARTS/api/v1/Plugins/ApiTestGroup
```

Тело ответа


```
    {
      "id": "2",
      "name": "2223",
      "valueInt": 0,
      "valueBool": false
    },
    {
      "id": "3",
      "name": "3333",
      "valueInt": 0,
      "valueBool": false
    }
  ]
}
```


Не нашли что искали?

Задать вопрос в техническую поддержку

Блокировка документа через REST API

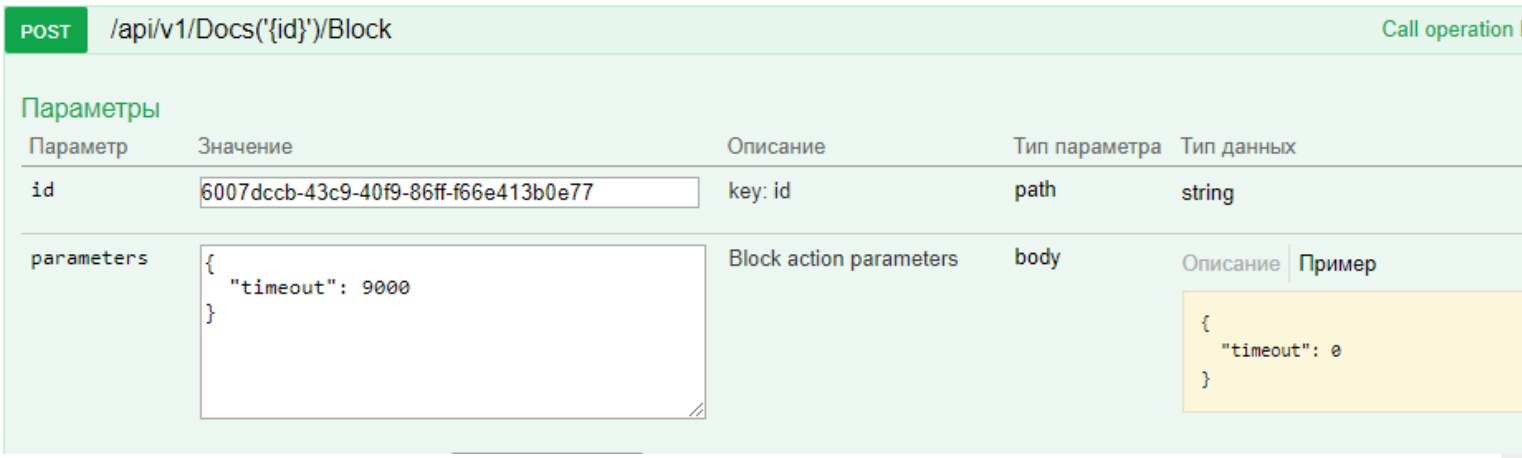
Последние изменения: 2024-03-26

Для редактирования коллективного документа через API нужно его заблокировать.

Блокировка методом block

Для этого нужно использовать метод block с идентификатором документа

http://localhost:12471/MobileSMARTS/api/v1/Docs ('6007dccb-43c9-40f9-86ff-f66e413b0e77')/Block

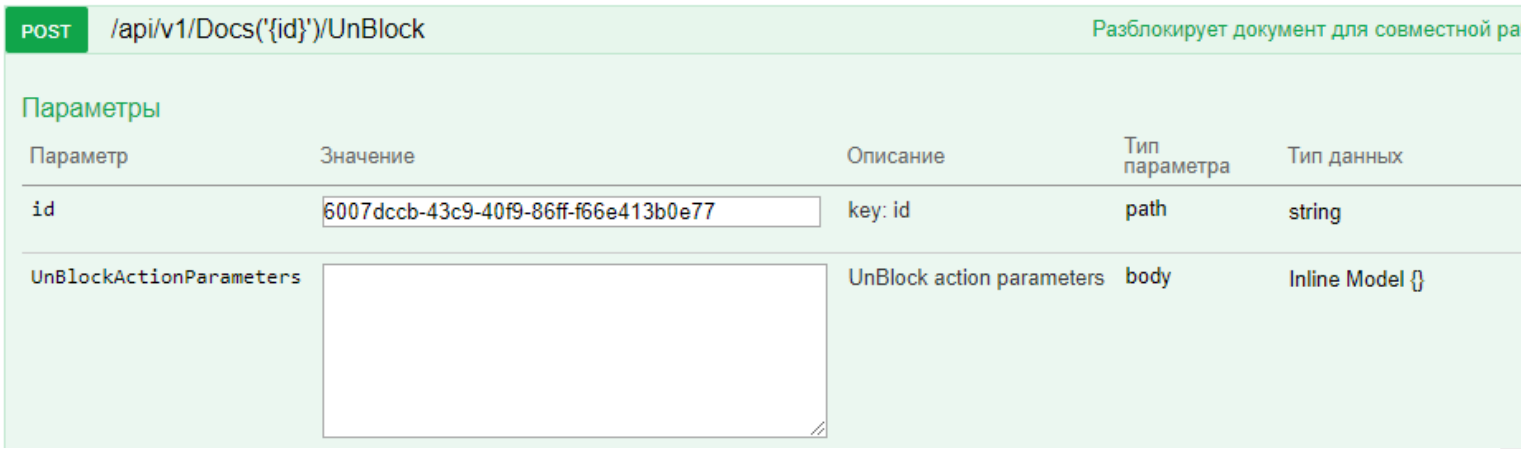


В параметрах нужно обязательно указывать таймаут (через какое время документ автоматически разблокируется, если его не разблокировали вручную). Диапазон от 500 до 10000 мс.

Разблокировка методом unblock

Для ручной разблокировки нужно использовать метод unblock с указанием идентификатора документа. После ручной разблокировки — автоматическая разблокировка не произойдет.

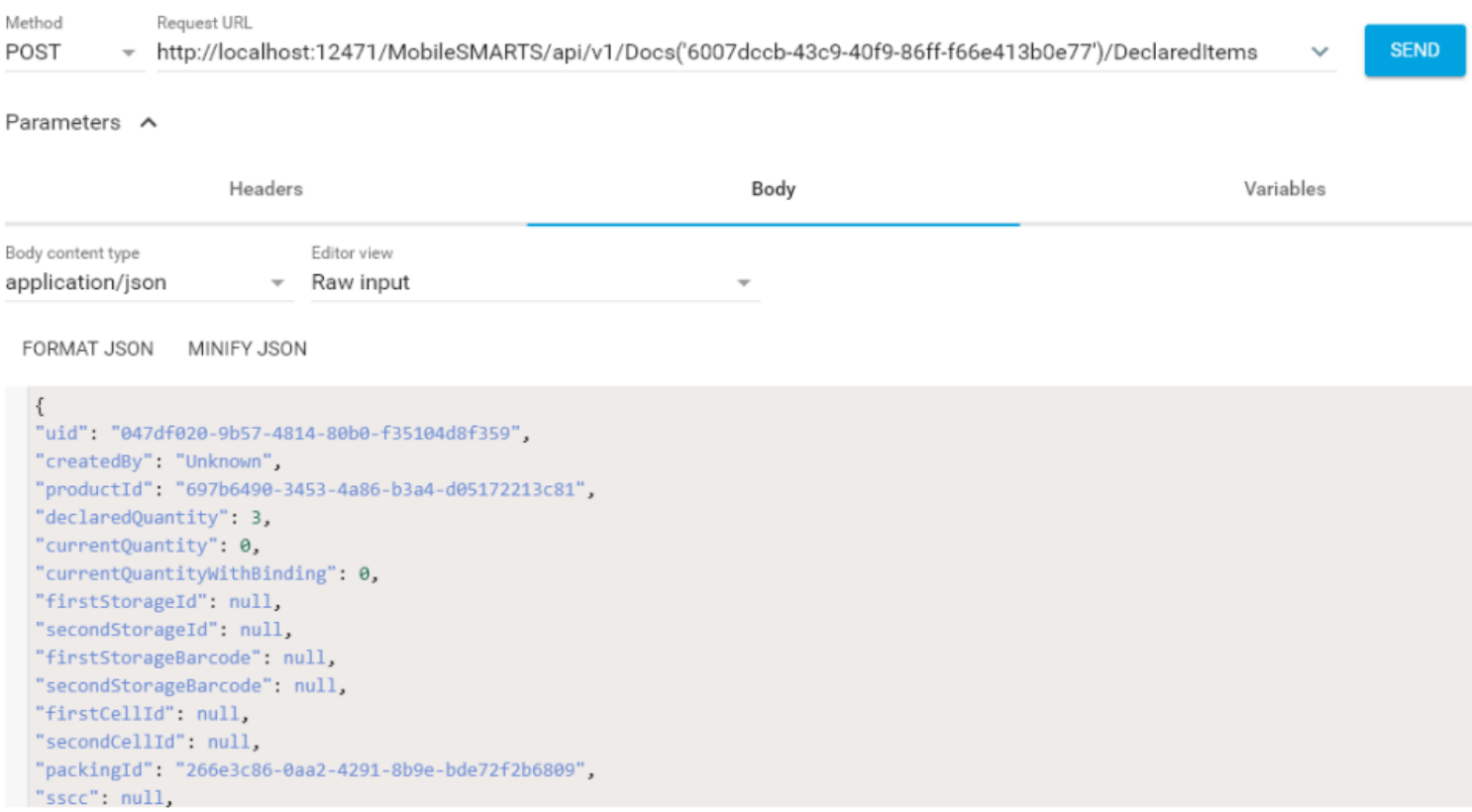
http://localhost:12471/MobileSMARTS/api/v1/Docs ('6007dccb-43c9-40f9-86ff-f66e413b0e77')/Unblock



Добавления новой строки методом post

Для добавления новой отдельной строки документа можно использовать метод post в DeclaredItems.

Пример

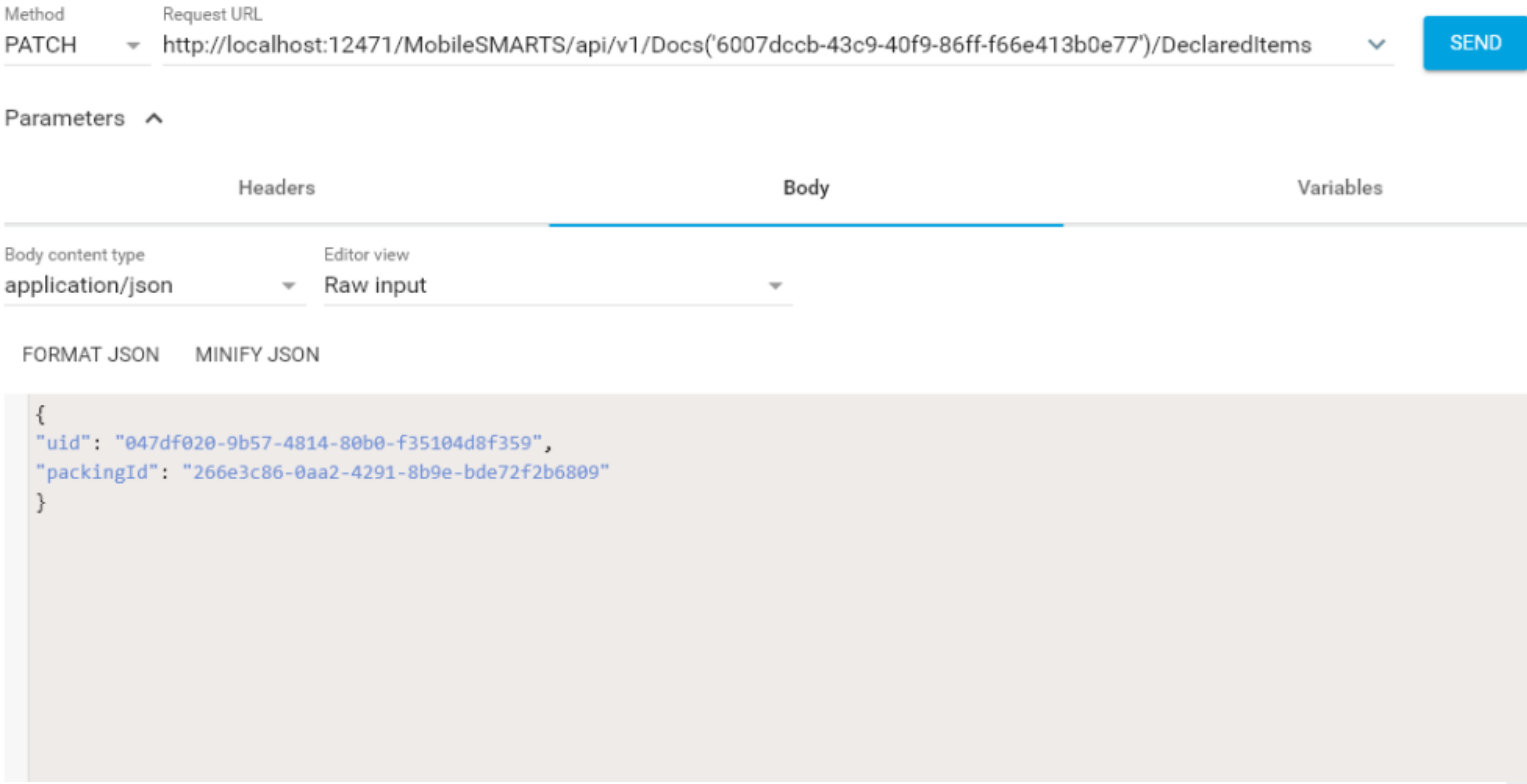


Редактирование методом patch

Для редактирования существующей строки можно пользоваться методом patch.

Пример

http://localhost:12471/MobileSMARTS/api/v1/Docs ('6007dccb-43c9-40f9-86ff-f66e413b0e77')/DeclaredItems



При патче можно указывать только те поля, которые необходимо обновить в нужной строке. В теле обязательно нужно указывать uid нужной строки!

Удаление строки методом Delete

Для удаления нужной строки у документа можно использовать метод Delete.

Пример

DELETE http://localhost:12471/MobileSMARTS/api/v1/Docs ('6007dccb-43c9-40f9-86ff-f66e413b0e77')/DeclaredItems ('047df020-9b57-4814-80b0-f35104d8f359'),

тут в урле необходимо указать как идентификатор документа, так и uid строки из declaredItems.

Не нашли что искали?

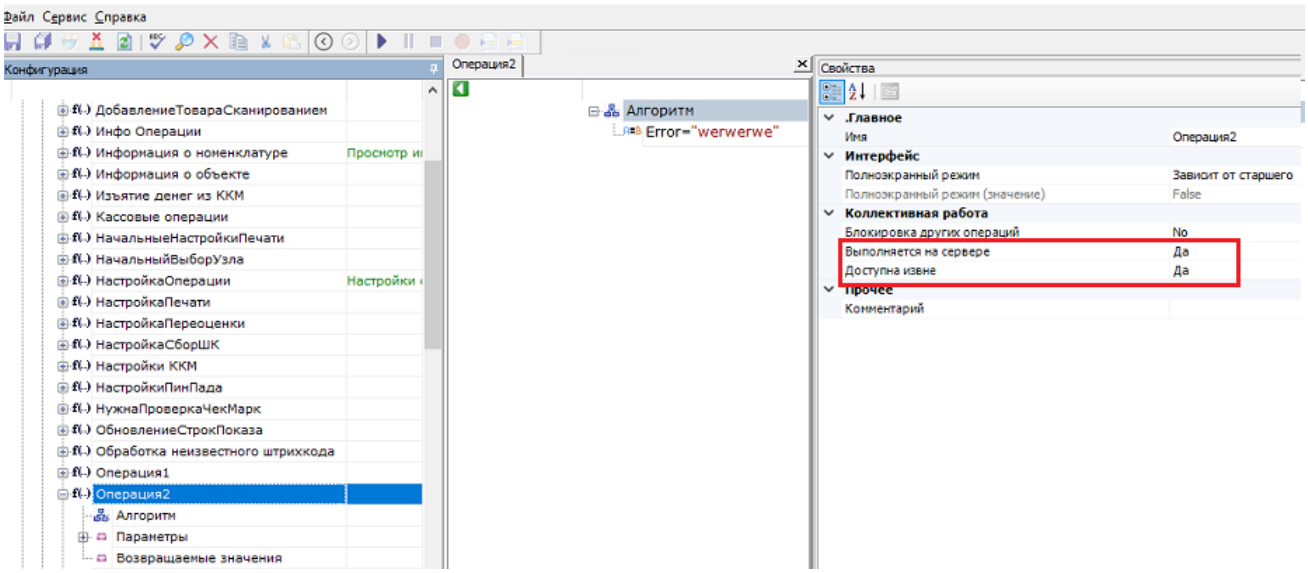
Задать вопрос в техническую поддержку

Вызов серверных операций через REST API

Последние изменения: 2024-03-26

Начиная с версии 3.2.46.20453 у платформы Mobile SMARTS появилась возможность вызывать серверные операции с помощью REST API. Для этого необходимо:

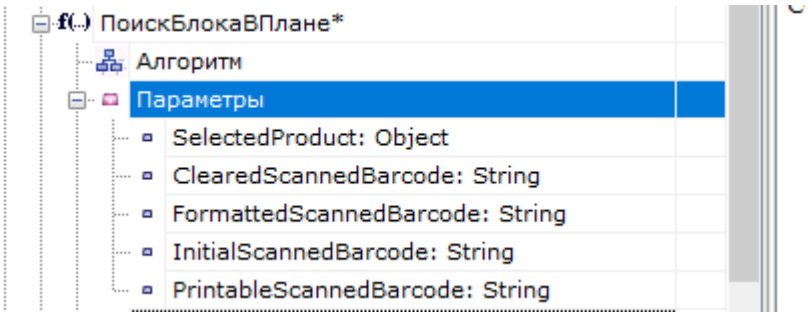
1. В панели управления Mobile SMARTS для требуемой операции установить значения «Да» у параметров «Выполняется на сервере» и «Доступна извне».



2. Выполнить запрос http://localhost:9000/MobileSMARTS/api/v1/Operations('{OperationName}'),

   где {OperationName} - имя требуемой операции.

3. Входные параметры, необходимые для выполнения операции указываются тут:



4. Если в операцию необходимо передать параметры, то они перечисляются в теле запроса в формате json:

```
{
"a": "параметр1",
"b": "параметр2"
}
```

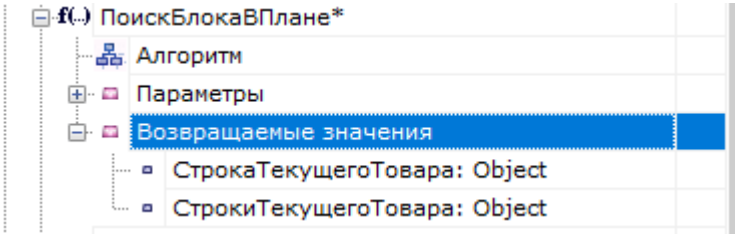
5. Если в параметрах необходимо передать сложный тип, то при описании параметра в json необходимо добавить поле "@odata.type", в котором указывается тип передаваемого параметра. Например:

```
{
"field0": "11",
"field1": {
"@odata.type": "#Cleverence.Warehouse.Document",
"id": "7c342252-de63-42fe-9742-b47b0a40a7ee"
}
}
```

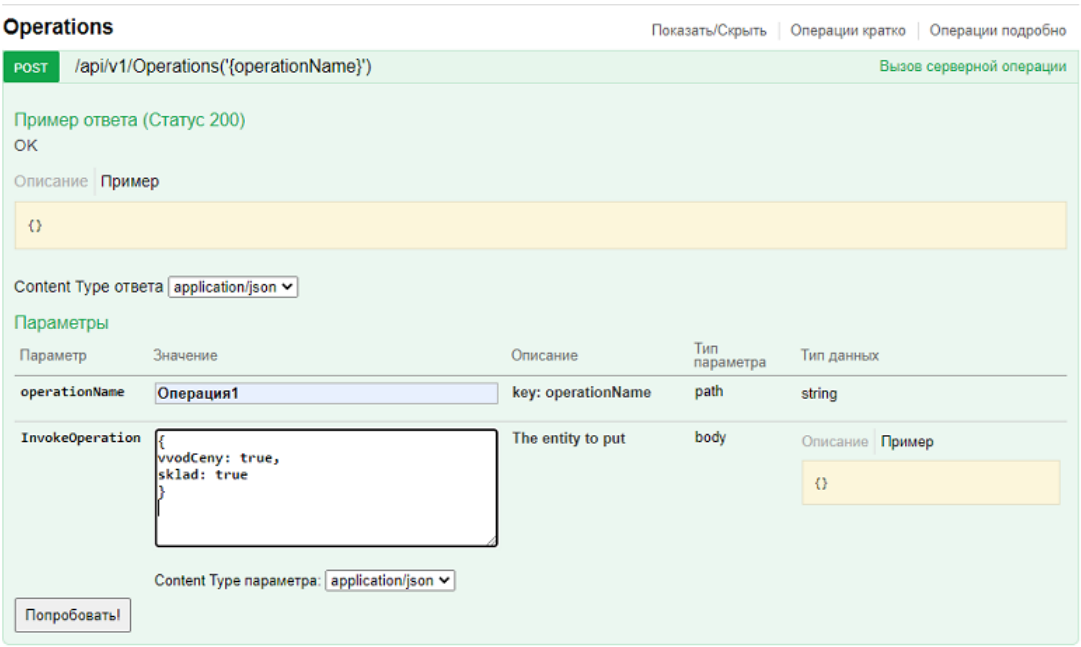
6. **В ответе сервер вернет результат выполнения операции в виде json, в котором будут перечислены выходные параметры после выполнения операции. Например:**

```
{
vvodCeny: true,
sklad: true
}
```

7. **Выходные параметры указываются при настройке операции тут:**



8. **Выполнение серверной операции можно протестировать в Swagger:**



Не нашли что искали?

Задать вопрос в техническую поддержку

Интеграция с «1С:Предприятием» в отраслевых продуктах на платформе Mobile SMARTS

Последние изменения: 2024-03-26

Программные продукты на платформе Mobile SMARTS должны быть интегрированы с какой-либо учетной системой, чтобы переносить в нее все данные по товарам с мобильных устройств (ТСД), что необходимо для осуществления дальнейшего учета и контроля товаров. В качестве учетной могут выступать различные программы (Excel/ TXT, SAP, Axapta), но в данной статье мы рассмотрим интеграцию программных продуктов от «Клеверенс» с учетной системой «1С: Предприятие».

Доступ из 1С к мобильному устройству

Требуется для осуществления таких операций, как выгрузка документа-задания на ТСД, выгрузка справочников номенклатуры товаров и др. Используется во время работы в офлайн-режиме.

Реализуется при помощи:

- либо специальных интеграционных обработок для 1С;
- либо через реализованную в продуктах «Клеверенса» поддержку БПО (библиотека стандартных подсистем и торгового оборудования).

Доступ с мобильного устройства к 1С

Требуется для получения актуальной информации о товаре (номенклатура, остатки и др.) и новых документов на обработку в режиме реального времени.

Используется во время работы в онлайн-режиме, когда требуется доступ к базе 1С (что такое онлайн-режим можно прочитать здесь (для «Магазина 15») и здесь (для «Склада 15»).

Реализуется при помощи:

- либо коннектора к внешней системе (1С) через COM и специальных интеграционных обработок для 1С, которые используются коннектором;
- либо коннектора к внешней системе (1С) через веб-сервисы и специального расширения «Клеверенс» для конфигураций 1С.

Различия в интеграции

В чем различия интеграций с 1С у разных отраслевых продуктов на Mobile SMARTS («Магазин 15», «Склад 15» и т. п.):

- Отличается список поддерживаемых готовых конфигураций «1С: Предприятия», потому что реализация правильного обмена данных из коробки требует отдельных усилий и отличается у разных продуктов.
- Информационный обмен документами между мобильными устройствами сбора данных и 1С происходит с использованием различных бизнес-процессов (подробнее об этом можно почитать в статьях «Бизнес-процессы» и «Бизнес-процессы в 1С»).
- Разная структура дополнительных таблиц Mobile SMARTS, которые предназначены для заполнения выгружаемых на ТСД документов данными, необходимыми для работы.

Более подробно узнать о интеграциях продуктов от «Клеверенса» с «1С: Предприятие» можно в соответствующих разделах на сайте.

«Mobile SMARTS: Магазин 15»

Узнать о поддерживаемых конфигурациях 1С, бизнес-процессах, обработках 1С и другую полезную информацию можно в разделе «Интеграция с «1С: Предприятие»».

Узнать о том, как работать с обработкой 1С, какие есть настройки для работы и обмена справочниками и документами можно в разделе «Магазин 15» для «1С:Предприятие».

«Mobile SMARTS: Склад 15»

Узнать о поддерживаемых конфигурациях 1С, бизнес-процессах и другую полезную информацию можно в разделе «Интеграция с «1С: Предприятие»».

Узнать о том, как работать с обработкой 1С, какие есть настройки для работы и обмена справочниками и документами можно в разделе «Склад 15» для «1С:Предприятие».

Драйвера для ТСД (ПРОФ драйвер)

Почитать о интеграции данного продукта с 1С можно в следующих статьях:

- Подключение драйвера в 1С (для онлайна и офлайна)
- Промежуточная конфигурация 1С (для онлайна)
- Настройка промежуточной базы 1С (для онлайна)

«Mobile SMARTS: Курьер»

Узнать, какие есть настройки для работы и обмена справочниками и документами можно в разделе «Интеграция с 1С».

Не нашли что искали?

Задать вопрос в техническую поддержку

Интеграция ЕГАИС с произвольной конфигурацией «1С: Предприятия»

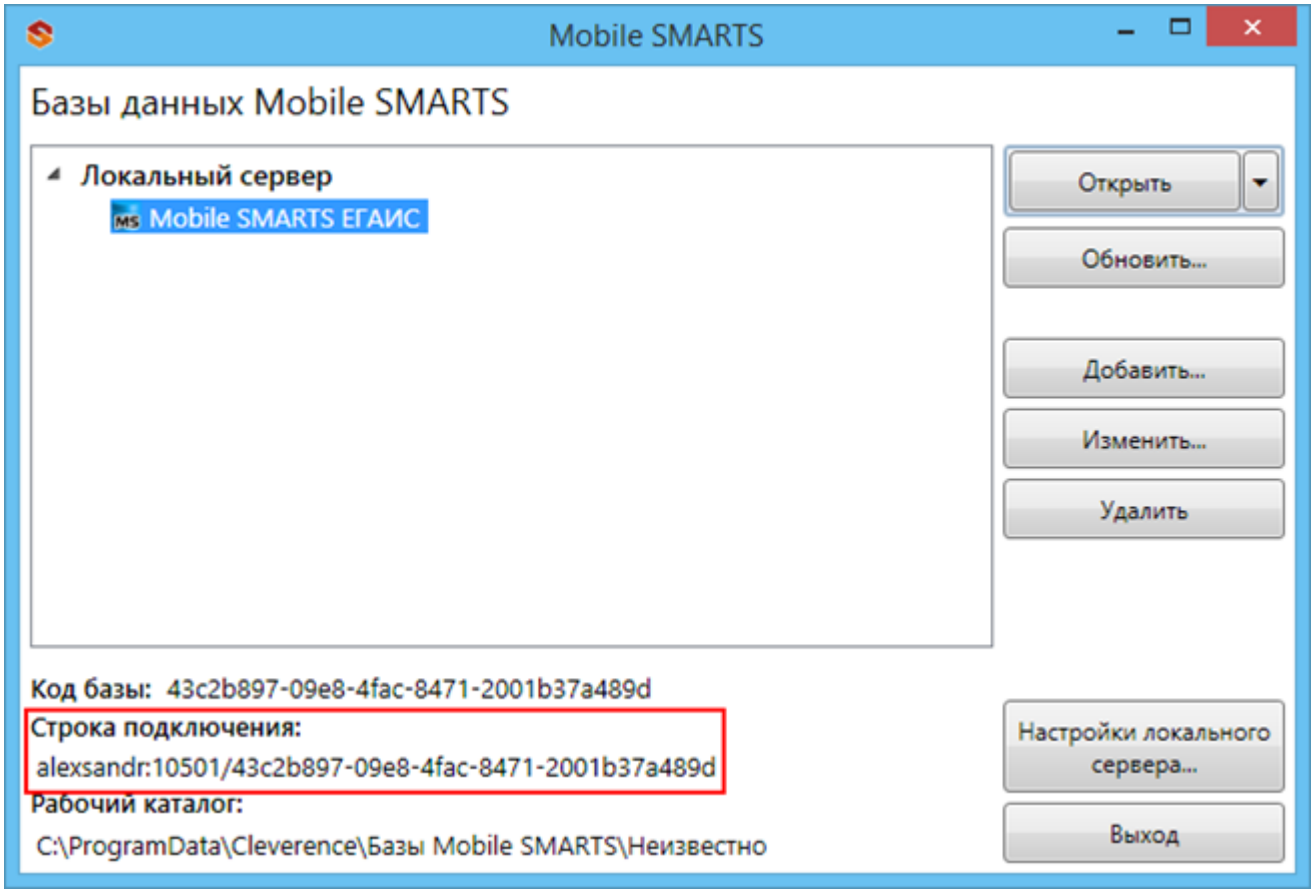
Последние изменения: 2024-03-26

- Скачать инструкцию по интеграции базы MSv8 (для 1С:Предприятия 8);
- Скачать инструкцию по интеграции базы MSv7 (для 1С:Предприятия 7.7);
- Скачать инструкцию по интеграции «Mobile SMARTS: ЕГАИС» с произвольной системой через COM

Первоначальная настройка и подключение

1. Устанавливаем платформу и конфигурацию Mobile SMARTS ЕГАИС;
2. На рабочем столе появится иконка для запуска Mobile SMARTS;

3. Зайдя в менеджер баз данных Mobile SMARTS, мы можем проверить адрес подключения и рабочий каталог, по которому расположена база «ЕГАИС»;



4. В установку входит внешняя компонента Cl.TerminalConnector, которая (если установка прошла корректно), уже будет зарегистрирована в системе;
5. Если эта компонента встала успешно, то мы получили возможность создавать com объект драйвера, и можем обращаться напрямую к нашей базе «ЕГАИС».

Вызываем компоненту и подключаем необходимую базу:

Пример кода 1С:Предприятие 8

```
//мДрайверТСД = новый COMОбъект("Cleverence.Warehouse.StorageConnector"); // Для
серверной версии
 ПрогИД = "AddIn.Cl.TerminalConnector";
 ПодключитьВнешнююКомпоненту(ПрогИД);
 мДрайверТСД = Новый (ПрогИД);
 СтрокаПодключения = "dt-501-2:10501/6d7e3a2f-3967-4734-bb9c-d1ba605aaef8";
 МассивПодключения = Новый Массив;
 МассивПодключения.Добавить(СтрокаПодключения);
 Если Не мДрайверТСД.Подключить(МассивПодключения, "") Тогда
  Сообщить("Не удалось подключиться к базе данных MS");
 КонецЕсли;
 // СтрокаПодключения - строка подключения из настройки базы MS
```

Пример кода 1С:Предприятие 7.7

```
 Компонента = "AddIn.Cl.TerminalConnector";
 Если ПодключитьВнешнююКомпоненту(Компонента) = 0 Тогда
ОписаниеРезультата = "ошибка загрузки внешней компоненты """ + Компонента + """";
 Иначе
  мДрайверТСД = СоздатьОбъект(Компонента);
  мДрайверТСД.УстановитьВерсию1С("v7");
  СтрокаПодключения = "dt-501-2:10501/6d7e3a2f-3967-4734-bb9c-d1ba605aaef8";
  Сз = СоздатьОбъект("СписокЗначений");
  Сз.ДобавитьЗначение(СтрокаПодключения);
  Если мДрайверТСД.Подключить(Сз,"") = 0 Тогда
     Описание = "";
   мДрайверТСД.ПолучитьОшибку(Описание);
   Сообщить("Ошибка при подключении: " + Описание);
   мДрайверТСД.ОсвободитьРесурсы();
   Возврат; КонецЕсли;
 КонецЕсли;      Описание = "";
```

Пример кода для произвольной учетной системы через компоненту COM

```
connection = new COM("Cleverence.Warehouse.StorageConnector");
// СтрокаПодключения - строка подключения из настройки базы MS
connection.SelectCurrentApp(СтрокаПодключения);
```

Обмен данными

Разделяем обмен данными на четыре этапа внутри

- **Этап 1 – выгрузка номенклатуры;**
- **Этап 2 – выгрузка документов;**
- **Этап 3 – выгрузка данных форм «А»;**
- **Этап 4 – загрузка документов.**

Этап первый – выгрузка номенклатуры

Перед выгрузкой номенклатуры мы должны её инициализировать:

Пример кода 1С:Предприятие 8

```
Если Не мОбъектТСД.НачатьВыгрузкуТоваров(мДанные) Тогда
   мОбъектТСД.ПолучитьОшибку(мОбъектТСД.ОписаниеОшибки);
   ВывестиПредупреждение( "Ошибка при инициализации выгрузки: " + м
   ОбъектТСД.ОписаниеОшибки );
   Сообщить("Ошибка при инициализации выгрузки: " + мОбъектТСД.ОписаниеОшибки,
   СтатусСообщения.Важное );
   мОбъектТСД.ОсвободитьРесурсы();
   Возврат;
КонецЕсли;
```

Пример кода 1С:Предприятие 7.7

```
Если мОбъектТСД.НачатьВыгрузкуТоваров(мДанные) = 0 Тогда
   мОбъектТСД.ПолучитьОшибку(мОбъектТСД.ОписаниеОшибки);
   ВывестиПредупреждение( "Ошибка при инициализации выгрузки: " +
мОбъектТСД.ОписаниеОшибки );
   Сообщить("Ошибка при инициализации выгрузки: " + мОбъектТСД.ОписаниеОшибки);
   мОбъектТСД.ОсвободитьРесурсы();
   Возврат;
КонецЕсли;
```

Если по какой-либо причине начать выгрузку товаров не удалось, то система выдаст сообщение об ошибке.

Далее мы определяем, что есть определенная структура полей, которая неизменна. Сначала формируем массив строк в 1С для выгрузки, затем циклом последовательно его обходим – по каждой позиции.

мДанные – строка выгрузки. Пример кода 1С:Предприятие 8

```
// +++ Перед началом выгрузки необходимо передать в Mobile SMARTS структуру полей
выгрузки
// там, где тип указан явно – int или Boolean, они и есть. В других случаях тип поля всегда
строка
мДанные = Новый Массив(25);
мДанные.Установить(  0, "Product.Ид" );
мДанные.Установить(  1, "Product.Marking" );
…
…
мДанные.Установить( 24, "Product.АлкоКод" );
 Если Не мДрайверТСД.НачатьВыгрузкуТоваров(мДанные) Тогда // Происходит проверка на
соответствие структуры полей
  Ошибка = мДрайверТСД.ПолучитьОшибку("Описание ошибки");
  Сообщить("Ошибка при инициализации выгрузки: " + Ошибка, СтатусСообщения.Важное );
  мДрайверТСД.ОсвободитьРесурсы();
  Возврат;
КонецЕсли;
```

```
КонецЕсли;
СправочникНоменклатуры = Справочники.Номенклатура.Выбрать();
 Пока СправочникНоменклатуры.Следующий() Цикл
Если НЕ СправочникНоменклатуры.ЭтоГруппа Тогда
мДанные.Установить(  0, XMLСтрока(СправочникНоменклатуры.Ссылка));//"Product.Ид" );
мДанные.Установить(  1, СправочникНоменклатуры.Артикул);//"Product.Marking" );
мДанные.Установить(  2, СправочникНоменклатуры.Код);//"Product.Barcode" );
мДанные.Установить(  3, "0000000000000");//"Packing.Barcode" );
мДанные.Установить(  4, СправочникНоменклатуры.Наименование);//"Product.Name" );
мДанные.Установить(  5,
XMLСтрока(СправочникНоменклатуры.ЕдиницаДляОтчетов.Ссылка));//"Product.BasePackingId" );
мДанные.Установить(  6,
XMLСтрока(СправочникНоменклатуры.ЕдиницаДляОтчетов.Ссылка));//"Packing.Ид" );
мДанные.Установить(  7,
СправочникНоменклатуры.ЕдиницаДляОтчетов.Наименование);//"Packing.Name" );
мДанные.Установить(  8, 1);//"Packing.UnitsQuantity" );
мДанные.Установить(  9, "");//"Packing.descr" );
мДанные.Установить( 10, "");//"Packing.serial" );
мДанные.Установить( 11, 0);//"Packing.price" );
мДанные.Установить( 12, 1);//"Packing.qty" );
мДанные.Установить( 13, ложь);//"Product.withserial" );
мДанные.Установить( 14, Истина);//"Product.Алко" ); продукцией (bool)
мДанные.Установить( 15, "лицензия на продажу спирта"); //"Product.АлкоВидЛиц" );
мДанные.Установить( 16, Истина);//"Product.АлкоМарк" );
мДанные.Установить( 17, "420");//"Product.АлкоКодВ" );
мДанные.Установить( 18, "СправочникНоменклатуры.Наименование");//"Product.АлкоНаимВ" );
мДанные.Установить( 19, 1);//"Product.АлкоОбъем" );
мДанные.Установить( 20, 25);//"Product.АлкоКрепость" );
мДанные.Установить( 21, "РуОпт");//"Product.Производитель" );
мДанные.Установить( 22, "7774444454");//"Product.ПроизвИНН" );
мДанные.Установить( 23, "7707707700");//"Product.ПроизвКПП" );
мДанные.Установить( 24, "22550");//"Product.АлкоКод" );
Если Не мДрайверТСД.ДобавитьВВыгрузкуТоваров(мДанные) Тогда


    Ошибка = мДрайверТСД.ПолучитьОшибку("Описание ошибки");


    Сообщить( "Ошибка при выгрузке данных: " +  ". " + Ошибка, СтатусСообщения.Важное );


    Прервано = Истина;
Прервать;
КонецЕсли;
КонецЕсли;
КонецЦикла;
// ++ После того, как весь товар передан на СТД, необходимо поверить, что в процессе
передачи не возникло исключений
Если Не мДрайверТСД.ЗавершитьВыгрузкуТоваров() Тогда
 НомерОшибки = мДрайверТСД.ПолучитьОшибку("Номер ошибки");
 Сообщить("Ошибка при завершении выгрузки: " + НомерОшибки);
 Прервано = Истина;
```

```
КонецЕсли;
```

мДанные – строка выгрузки. Пример кода 1С:Предприятие 7.7

```
// +++ Перед началом выгрузки необходимо передать в Mobile SMARTS структуру полей выгрузки
// там, где тип указан явно – int или Boolean, они и есть. В других случаях тип поля всегда строка
мДанные = СоздатьОбъект("СписокЗначений");
мДанные.ДобавитьЗначение("Product.Ид" );
мДанные.ДобавитьЗначение("Product.Marking" );
…
…
мДанные.ДобавитьЗначение(24, " Packing.АлкоКод" );
 Если Не мДрайверТСД.НачатьВыгрузкуТоваров(мДанные) Тогда // Происходит проверка на соответствие структуры полей
  Ошибка = мДрайверТСД.ПолучитьОшибку("Описание ошибки");
  Сообщить("Ошибка при инициализации выгрузки: " + Ошибка, СтатусСообщения.Важное );
  мДрайверТСД.ОсвободитьРесурсы();
  Возврат;
 КонецЕсли;
Спр = СоздатьОбЪект("Справочник.Номенклатура");
 Спр.ВыбратьЭлементы();
 Пока Спр.ПолучитьЭлемент() = 1 Цикл
 мДанные.УдалитьВсе();
 мДанные.ДобавитьЗначение(Спр.Артикул);//"Product.Ид" );  // uuid элемента справочника
 мДанные.ДобавитьЗначение(Спр.Артикул);//"Product.Marking" ); // Артикул
 мДанные.ДобавитьЗначение(Спр.Код);//"Product.Barcode" ); // Код товара
 мДанные.ДобавитьЗначение("0000000000000");//"Packing.Barcode" ); // ШК
 мДанные.ДобавитьЗначение(Спр.Наименование);//"Product.Name" ); // Наименование
 мДанные.ДобавитьЗначение(Спр.Артикул);//"Product.BasePackingId" ); //Является базовой единицей
 мДанные.ДобавитьЗначение(Спр.Артикул);//"Packing.Ид" ); // Единица измерения
 мДанные.ДобавитьЗначение(Спр.Наименование);//"Packing.Name" ); // Наименование единицы измерения
 мДанные.ДобавитьЗначение(1);//"Packing.UnitsQuantity" ); // коэффициент  (кратность) единицы измерения
 мДанные.ДобавитьЗначение("");//"Packing.descr" ); // характеристика единицы измерения
 мДанные.ДобавитьЗначение("");//"Packing.serial" ); // серийный номер единицы измерения
 мДанные.ДобавитьЗначение(0);//"Packing.price" ); // стоимость
 мДанные.ДобавитьЗначение(1);//"Packing.qty" ); //Количество
 мДанные.ДобавитьЗначение(0);//"Product.withserial" ); // Используются серии
 мДанные.ДобавитьЗначение(1);//" Packing.Алко" ); // Является алкоголе содержащей продукцией (bool)
 мДанные.ДобавитьЗначение("лицензия на продажу спирта"); //" Packing.АлкоВидЛиц" ); // Вид лицензии
 мДанные.ДобавитьЗначение(1);//" Packing.АлкоМарк" ); // Является маркируемой продукцией
 мДанные.ДобавитьЗначение("420");//" Packing.АлкоКодВ" );  // Код вида алкогольной продукции
мДанные.ДобавитьЗначение("СправочникНоменклатуры.Наименование");//"Product.АлкоНаимВ" );  // Наименование вида алкогольной продукции
 мДанные.ДобавитьЗначение(1);//" Packing.АлкоОбъем" );  // Объем
```

```
мДанные.ДобавитьЗначение(1);//" Packing.АлкоОбъем" );  // Объем
мДанные.ДобавитьЗначение(25);//" Packing.АлкоКрепость" ); // Крепость
мДанные.ДобавитьЗначение("РуОпт");//" Packing.Производитель" ); // производитель
мДанные.ДобавитьЗначение("7774444454");//" Packing.ПроизвИНН" ); // ИНН Производителя
мДанные.ДобавитьЗначение("7707707700");//" Packing.ПроизвКПП" ); // КПП Производителя
мДанные.ДобавитьЗначение("22550");//" Packing.АлкоКод" );  // Код алкогольной продукции
Если мДрайверТСД.ДобавитьВВыгрузкуТоваров(мДанные) = 0 Тогда


    Ошибка = мДрайверТСД.ПолучитьОшибку("Описание ошибки");


    Сообщить( "Ошибка при выгрузке данных: " +  ". " + Ошибка, СтатусСообщения.Важное );


    Прервано = 1;
Прервать;
КонецЕсли;
КонецЕсли;
КонецЦикла;
// ++ После того, как весь товар передан на СТД, необходимо поверить, что в процессе
передачи не возникло исключений
Если Не мДрайверТСД.ЗавершитьВыгрузкуТоваров() Тогда
 НомерОшибки = мДрайверТСД.ПолучитьОшибку("Номер ошибки");
 Сообщить("Ошибка при завершении выгрузки: " + НомерОшибки);
 Прервано = Истина;
КонецЕсли;
```

Понятно, что в тестовой выгрузке номенклатуры ряд полей заполнены произвольно. (алкоКод, АлкоОбъем, и т.д.). Эти поля заполняются в зависимости от используемой конфигурации.

Для выгрузки номенклатуры используется три функции компоненты подключения:

Начать выгрузку товаров

BeginUploadProducts(bool anyway, bool overwriteExisting, bool generateFullTextSearch)

Параметр	Тип	Описание
anyway	bool	Начать новую выгрузку, даже если какая-то другая выгрузка уже была открыта. В случае true – позволяется начинать сразу несколько выгрузок одновременно.

overwriteExistingbool		Флаг, определяющий полностью перезаписывать весь справочник номенклатуры или слить к выгруженным ранее.
generateFullTextSearchbool		Флаг, определяющий следует ли генерировать индексы для поиска товаров по части имени. Если такой функционал не требуется, то лучше использовать false так как такой индекс значительно увеличивает объем данных, загружаемых на ТСД.

Выгрузка товаров

UploadProducts(ProductCollection products)

Параметр	Тип	Описание
products	ProductCollection	Коллекция товаров для выгрузки.

Функция принимает для выгрузки коллекцию товаров (Product).

Структура данных Mobile SMARTS требует, чтобы каждый товар содержал хотя бы одну упаковку (Packing).

Пример кода для произвольной учетной системы через компоненту COM

```
// создание коллекции
productsColl = new COM("Cleverence.Warehouse.ProductCollection");
// создание товара
product = new COM("Cleverence.Warehouse.Product");
product.SetField(“Id”, уникИдТовара);
product.SetField(“Name”, "Товар1");
//создание упаковки
packing = new COM("Cleverence.Warehouse.Product");
packing.SetField(“Id”, идУпаковки); //уникальный в пределах товара
packing.SetField(“Name”, "шт");
product.Packings.Add(packing); //добавление упаковки в товар
//обязательно – установка кода базовой упаковки для товара
product.SetField(“BasePackingId”, packing.Id);
productsColl.Add(product); //добавление товара в коллекцию

//выгрузка
connection.UploadProducts(productsColl);
```

Список полей для заполнения в товаре:

Поле	Тип	Основное/дополнительное	Описание
Id	string	основное	Уникальный идентификатор товара.
Name	string	основное	Наименование товара.
BasePackingId	string	основное	Уникальный идентификатор базовой упаковки товара.
Marking	string	основное	Артикул.
withserial	int	дополнительное	Ведется ли серийный учет по товару (1 – ведется, 0 – нет).
Alco	bool	дополнительное	Является алкоголе содержащей продукцией.

АлкоКод			
string			
дополнительное			
Код алкогольной продукции.			

АлкоВидЛиц			
string			
дополнительное			
Вид лицензии (просто информационное поле).			

АлкоМарк			
bool			
дополнительное			
Является маркируемой продукцией.			

АлкоКодВ			
string			
дополнительное			
Код вида алкогольной продукции.			

АлкоНаимВ			
string			
дополнительное			
Наименование вида алкогольной продукции.			

АлкоОбъем			
double			
дополнительное			
Объем.			

АлкоКрепость			
double			
дополнительное			
Крепость.			

Производитель			
string			
дополнительное			
Производитель.			

ПроизвИНН			
string			
дополнительное			
ИНН Производителя.			

ПроизвКПП			
string			
дополнительное			
КПП Производителя.			

Список полей для заполнения в упаковке:

Поле	Тип	Основное/дополнительное	Описание

Id	string	основное	Уникальный идентификатор товара.
Name	string	основное	Наименование упаковки.
UnitsQuantity	double	основное	Коэффициент (кратность) единицы измерения.
Barcode	string	основное	Штрихкод товара (обычно EAN13) в текущей упаковке.
price	double	дополнительное	Стоимость.
qty	double	дополнительное	Количество.
serial	string	дополнительное	Серия, привязанная к упаковке (для серийного учета).
descr	string	дополнительное	Характеристика товара, привязанная к упаковке.

Заполнив таким образом данные, мы можем отправить их в терминал (по адресу и порту сервера, который получили при первоначальной настройке).

Пример кода 1С:Предприятие 8

```
// Выгрузка
Если Не мОбъектТСД.ДобавитьВВыгрузкуТоваров(мДанные) Тогда
мОбъектТСД.ПолучитьОшибку(мОбъектТСД.ОписаниеОшибки);
Сообщить( "Ошибка при выгрузке данных: " + ". " + мОбъектТСД.ОписаниеОшибки,
СтатусСообщения.Важное );
Прервано = Истина;
Прервать;
КонецЕсли;
```

Пример кода 1С:Предприятие 7.7

```
// Выгрузка
 Если мОбъектТСД.ДобавитьВВыгрузкуТоваров(мДанные) = 0 Тогда
   мОбъектТСД.ПолучитьОшибку(мОбъектТСД.ОписаниеОшибки);
 Сообщить( "Ошибка при выгрузке данных: " + ". " + мОбъектТСД.ОписаниеОшибки,
       СтатусСообщения.Важное );
   Прервано = 1;
      Прервать;
    КонецЕсли;
```

После того, как все данные выгружены, необходимо завершить выгрузку товаров:

Пример кода 1С:Предприятие 8

```
Если Не мОбъектТСД.ЗавершитьВыгрузкуТоваров() Тогда
НомерОшибки = мОбъектТСД.ПолучитьОшибку(мОбъектТСД.ОписаниеОшибки);
Сообщить("Ошибка при завершении выгрузки: " + мОбъектТСД.ОписаниеОшибки);
Прервано = Истина;
КонецЕсли;
```

Пример кода 1С:Предприятие 7.7

```
 Если мОбъектТСД.ЗавершитьВыгрузкуТоваров() = 0 Тогда
НомерОшибки = мОбъектТСД.ПолучитьОшибку(мОбъектТСД.ОписаниеОшибки);
Сообщить("Ошибка при завершении выгрузки: " + мОбъектТСД.ОписаниеОшибки);
Прервано = 1;
 КонецЕсли;
```

Заполнение/получение основных полей можно производить как напрямую, так и через функции SetField/GetField. Для дополнительных полей можно использовать только GetField/SetField.

ИМЕНА ПОЛЕЙ ЧУСТВИТЕЛЬНЫ К РЕГИСТРУ!

Пример кода для произвольной учетной системы через компоненту COM

```
product.Id = “00001”;
//или
product.SetField(“Id”, “00001”);
//доп поле
product.SetField(“Алко”, true);
```

Завершение выгрузки товаров

EndUploadProducts()

Завершает процедуру выгрузки товаров. После её вызова сервер будет считать выгрузку завершенной и начнет у себя обновление справочников, генерацию индексов.

До вызова этой функции товары не считаются выгруженными и недоступны.

Полный цикл Пример кода для произвольной учетной системы через компоненту COM

```
connection = new COM("Cleverence.Warehouse.StorageConnector");
// СтрокаПодключения - строка подключения из настройки базы MS
connection.SelectCurrentApp(СтрокаПодключения);
//начало выгрузки
//вне зависимости от прошлых выгрузок, с полным переписыванием на сервере, без индекса по
//именам товаров
Connection.BeginUploadProducts(true, true, false);
productsColl = new COM("Cleverence.Warehouse.ProductCollection");
for(int i = 0; i< колво_товаров_в_системе; i++) //цикл по товарам в системе
{
 //выгружаем блоками по 500 товаров
 if(productsColl.Count == 500)
 {
  //выгрузка блока из 500 товаров
  connection.UploadProducts(productsColl);
  productsColl = new COM("Cleverence.Warehouse.ProductCollection");
 }
 // создание товара
 product = new COM("Cleverence.Warehouse.Product");
 product.SetField(“Id”, “ид” + i);
 product.SetField(“Name”, "Товар " + i);
 //создание упаковки
 packing = new COM("Cleverence.Warehouse.Product");
 packing.SetField(“Id”, идУпаковки); //уникальный в пределах товара
 packing.SetField(“Name”, "шт");
 ...
 //заполнение остальных полей упаковки
 ...
 product.Packings.Add(packing); //добавление упаковки в товар
 //обязательно – установка кода базовой упаковки для товара
 product.SetField(“BasePackingId”, packing.Id);
 productsColl.Add(product); //добавление товара в коллекцию
}
//выгрузка оставшихся товаров
if (productsColl.Count > 0)
 connection.UploadProducts(productsColl);
//завершение выгрузки
connection.EndUploadProducts();
```

Этап второй. Выгрузка документов

Аналогично работе с номенклатурой, инициализируем внешнюю компоненту Клеверенс. После чего, определяем, какой документ, и куда мы будем выгружать. На сервере Mobile SMARTS по умолчанию доступные следующие документы:

- Собрать штрихкоды;
- Сбор начальных остатков;
- Приход на склад;
- Подбор заказа;
- Инвентаризация.

Применительно к алкогольной тематике, мы можем проставить следующее соответствие:

- сбор начальных остатков = Акт постановки на баланс;
- приход на склад = ТТН ЕГАИС в статусе «от поставщика»;
- подбор заказа = ТТН ЕГАИС в статусе «покупателю».

Сбор начальных остатков.

Описание полей:

На сервере Mobile SMARTS:ЕГАИС по умолчанию есть следующие типы документов, доступные для выгрузки:

- Приход на склад;
- Подбор заказа;
- Инвентаризация.

Применительно к алкогольной тематике, мы можем проставить следующее соответствие:

- приход на склад = ТТН ЕГАИС в статусе «от поставщика»;
- подбор заказа = ТТН ЕГАИС в статусе «покупателю».

Функция выгрузки

UploadDocument(Document document)

Параметр	Тип	Описание
document	Document	Документ Mobile SMARTS для выгрузки.

Функция принимает для выгрузки объект документа (Document).

Пример кода для произвольной учетной системы через компоненту COM

```
// Аналогично работе с номенклатурой, инициализируем подключение
connection = new COM("Cleverence.Warehouse.StorageConnector");
// СтрокаПодключения - строка подключения из настройки базы MS
connection.SelectCurrentApp(СтрокаПодключения);
// создание документа
document = new COM("Cleverence.Warehouse.Document");
//заполнение документа
//...
//...
// выгрузка в Mobile SMARTS
connection.UploadDocument(document);
```

Каждый документ Mobile SMARTS имеет шапочную часть документа, и две табличных части (DeclaredItems - строки заявки, CurrentItems – фактические строки с ТСД).

При выгрузке документа нам необходимо заполнить нужные поля шапки и строки заявки.

Заполнение шапки документа:

Реквизиты «шапки» документа

Реквизит	Тип	Основное/доп.	Описание
Id	string	основное	Идентификатор документа. Обязательно заполняется при выгрузке документа и должен иметь уникальное значение.
Name	string	основное	Отображаемое имя документа. Обязательно заполняется при выгрузке документа.

Appointment

string

основное

Назначение документа - код пользователя или имя группы, которым данный документ назначается на исполнение. Если значение пустое, то документ попадает к первому свободному пользователю, которому разрешен тип документа.

UserID

string

основное

Идентификатор пользователя, который выполнил документ. Должен иметь уникальное значение для каждого пользователя.

UserName

string

основное

Имя пользователя.

DocumentTypeName

string

основное

Имя типа документа. Обязательно заполняется при выгрузке документа и должно иметь уникальное значение.

Modified

boolean

основное

Признак того, что документ был изменен. Заполняется при изменении документа одним из пользователей.

InProgress

boolean

основное

Признак того, что документ захвачен пользователем на обработку. Заполняется, когда документ был захвачен пользователем.

Finished

boolean

основное

Признак того, что обработка документа пользователем была завершена, и его можно забирать назад в учетную систему. Заполняется, когда работа с документом была завершена на терминале.

WareHouseID

string

основное

Идентификатор склада, к которому привязан документ. Должен иметь уникальное значение для каждого склада.

Barcode

string

основное

Штрихкод документа. Для выбора документа по штрихкоду на терминале или на сервере должен быть обязательно заполнен.

Priority

Int32

основное

Приоритет документа. Более приоритетные документы раньше отдаются на терминал для обработки.

DistributeByBarCode

boolean

основное

Признак выдачи документа по штрихкоду. Документы с таким признаком не поступают на мобильный терминал автоматически, а могут быть выбраны с сервера только по штрихкоду. Свойство может успешно применяться только при наличии постоянной связи с сервером.

ServerHosted

boolean

основное

Признак того, что документ должен выполняться "на сервере". Такой документ могут одновременно открыть на редактирование несколько пользователей. Все изменения в документе будут происходить одновременно для всех работающих с ним пользователей. Работа в таком режиме требует наличия постоянной связи с сервером.

КонтрольКолва	Int32	дополнительное	Контроль количества по заявке. 1 – контролируется, в документ нельзя принимать товары, которых нет в заявке, а также нельзя превышать количество товара, который в заявке есть 0 – без контроля.
НомерЕгаис	string	дополнительное	Номер ТТН ЕГАИС.

Пример кода для произвольной учетной системы через компоненту COM

```
// создание документа
document = new COM("Cleverence.Warehouse.Document");
document.Id = "5BFF9B72-BA92-409D-BA01-CA909C3CB7C5";
document.DocumentTypeName = "Приход на склад";
document.Name = "Приход на склад №000001";
//для пользователя «оператор», заведенного в стандартной поставке
document.Appointment = "оператор";
//для выбора с сервера произвольным пользователем
//document.Appointment = "";
//document.DistibuteByBarcode = true;
//для склада, заведенного в стандартной поставке
document.WarehouseId = "1";
//дополнительные поля, также как и в товаре, заполняются через SetField
document.SetField("КонтрольКолва", 1);
document.SetField("НомерЕгаис","4576456456456");
```

Заполнение строк заявки

Каждая строка документа – это объект DocumentItem.

Пример кода для произвольной учетной системы через компоненту COM

```
// создание строки документа
documentItem = new COM("Cleverence.Warehouse.DocumentItem");
```

Реквизиты «строк» документа

Реквизит	Тип	Основное/доп.	Описание
CreatedBy	CreateBy	основное	Проверка упаковки.
ProductID	string	основное	Идентификатор товара, для которого описана данная позиция.
DeclaredQuantity	Double	основное	Заданное количество в позиции. Свойство заполняется при выгрузке документа из учетной системы, чтобы ограничить объем товара данного типа, с котором оперирует пользователь в документе.
CurrentQuantity	Double	основное	Текущее количество товара в позиции. Показывает какое реальное количество товара задействовано пользователем.

FirstCellID

string

основное

FirstStorageBarcode

string

основное

Штрихкод первого места хранения к которой привязан товар. В качестве места может выступать ячейка или палета.

PackingID

string

основное

Идентификатор упаковки для заданного товара. Товар задается свойством ProductId [ИдТовара].

SSCC

string

основное

Уникальный номер единицы хранения. Служит для идентификации конкретных экземпляров товара. Заполняется из штрихкода товара по шаблону при выполнении операции.

ExpiredDate

DateTime

основное

Срок годности товара. Инициализируется при занесении конкретного товара в позицию документа пользователем либо из штрихкода по шаблону, либо ручным выбором.

SecondStorageBarcode

string

основное

Штрихкод второго места хранения к которой привязан товар. В качестве места может выступать ячейка или палета.

BindedLine

DocumentItem

основное

Задает связь строки из CurrentItems [СтрокиФакт] со строкой в DeclaredItems [СтрокиПлан].

Code

string

дополнительное

Barcode

string

дополнительное

Штрихкод.

Serial

string

дополнительное

Серия товара (если используется учет по сериям).

Sn

string

дополнительное

Серийный номер.

Price

Decimal

дополнительное

Цена единицы товара в строке.

ЦенаСклад

string

дополнительное

Алко

boolean

дополнительное

Признак того, что товар является алкогольной или спиртосодержащей продукцией.

ПроверкаЧМ

boolean

дополнительное

Нужно проверять Check Mark или нет.

АлкоСН

string

дополнительное

Серийный номер бутылки.

АлкоКод

string

дополнительное

Строка с кодом алкогольной продукции в ЕГАИС.

АлкоНаим

string

дополнительное

Наименование товара.

АлкоКодВ

string

дополнительное

Строка с кодом вида алкогольной продукции.

АлкоОбъем

Decimal

дополнительное

Ёмкость тары в литрах. (в базе может быть в декалитрах, например, в Рознице).

АлкоКрепость

Decimal

дополнительное

Процентное содержание спирта.

Производитель

string

дополнительное

Строка с наименованием производителя (уже есть реализация в мобильной печати). Для алкоголя должно выгружаться вне зависимости от того, включена ли мобильная печать и стоит ли там галочка.

ПроизводительИНН	string	дополнительное	ИНН производителя, если уже известен для данного товара.
ПроизводительКПП	string	дополнительное	КПП производителя, если уже известен для данного товара.
АлкоПДФ	string	дополнительное	Строка с PDF 417.

Этап третий. Выгрузка данных форм «А»

Некоторые операции, например, приход на склад позволяют проводить автоматическую проверку и подбор номеров форм «А», если конечно данные об этих формах выгрузить заранее в Mobile SMARTS.

Формы А выгружаются в виде дополнительной таблицы.

Формат таблицы форм А

Реквизиты строки дополнительной табличной части документа «ФормыА»

Реквизит	Тип	Основное/доп.	Описание
КодФормы	string	дополнительное	Код формы в системе ЕГАИС.

КодНоменклатуры	string	дополнительное	Уникальный идентификатор номенклатуры из справочника номенклатуры, если есть.
АлкоКод	string	дополнительное	Строка с кодом алкогольной продукции в ЕГАИС.
Начало	int32	дополнительное	Начало диапазона серийных номеров.
Конец	int32	дополнительное	Конец диапазона серийных номеров.

Основной метод записи в поля документ – это команда «SetField» («УстановитьПоле» - русск)

Соответственно, метод получения данных из полей – «GetField» («ПолучитьПоле» - русск)

ИМЕНА ПОЛЕЙ ЧУСТВИТЕЛЬНЫ К РЕГИСТРУ!

Давайте попробуем создать документ «Сбор начальных остатков»

Перед началом выгрузки документа, мы должны определить его тип (в какой документ Mobile SMARTS мы будем выгружать документ 1С).

Пример кода 1С:Предприятие 8

```
ДокументТСД = Новый COMОбъект("Cleverence.Warehouse.Document");
ДокументТСД.Ид  - uuid Документа  =  XmlСтрокой(СсылкаНаНашДокумент);
ДокументТСД.ИмяТипаДокумента    = «Сбор начальных остатков»;
```

Пример кода 1С:Предприятие 7.7

```
ДокументТСД = СоздатьОбъект("Cleverence.Warehouse.Document");
ДокументТСД.Ид  - uuid Документа  =  ПолучитьUID(ССылкаНаНашДокумент);
ДокументТСД.ИмяТипаДокумента    = «Сбор начальных остатков»;
Функция ПолучитьUID(ССылкаНаНашДокумент)
 Стр=ЗначениеВСтрокуВнутр(Объект);
 СЗ=СоздатьОбъект("СписокЗначений");
 СЗ.ИзСтрокиСРазделителями(Сред(Стр,2,СтрДлина(Стр)-2));



     Возврат формат(СЗ.ПолучитьЗначение(4),"Ч(0)5")+Прав(СтрЗаменить(Формат("", "С9"), " ",
"0") + СокрЛП(СЗ.ПолучитьЗначение(СЗ.РазмерСписка())), 9);



    КонецФункции
// Здесь мы добавили функцию формирования УИД для документа
```

Таким образом мы создали новый документ. После его создания, необходимо явным образом объявить создание строк документа.

Строки документа создаются следующим образом:

Пример кода 1С:Предприятие 8

```
СтрокаДокументаТСД = Новый COMОбъект("Cleverence.Warehouse.DocumentItem");
//
// тут происходит заполнение полей объекта DocumentItem
СтрокаДокументаТСД.КоличествоПлан = 100
// Например, вышеприведённый код выгрузит 1 строку, где поле DeclaredQuantity будет равно
100
ДокументТСД.СтрокиПлан.Добавить(СтрокаДокументаТСД);
// После заполнения строк документа, нам необходимо зафиксировать сам объект:
Ответ = мДрайверТСД.ВыгрузитьДокумент(ДокументТСД);



    Если Не Ответ Тогда
 НомерОшибки = мОбъектТСД.ПолучитьОшибку(мОбъектТСД.ОписаниеОшибки);
 Сообщить("Ошибка: " + мОбъектТСД.ОписаниеОшибки, СтатусСообщения.Важное);
Иначе
 Сообщить("Документ """ + Строка(Документ1С) + """ выгружен на ТСД.",
СтатусСообщения.Обычное);
КонецЕсли;
```

Пример кода 1С:Предприятие 7.7

```
СтрокаДокументаТСД = СоздатьОбъект ("Cleverence.Warehouse.DocumentItem");
//
// тут происходит заполнение полей объекта DocumentItem
СтрокаДокументаТСД.КоличествоПлан = 100;
// Например, вышеприведённый код выгрузит 1 строку, где поле DeclaredQuantity будет равно
100
ДокументТСД.СтрокиПлан.Добавить(СтрокаДокументаТСД);
// После заполнения строк документа, нам необходимо зафиксировать сам объект:
Ответ = мДрайверТСД.ВыгрузитьДокумент(ДокументТСД);



    Если Ответ = 0 Тогда
 НомерОшибки = мОбъектТСД.ПолучитьОшибку(мОбъектТСД.ОписаниеОшибки);
 Сообщить("Ошибка: " + мОбъектТСД.ОписаниеОшибки);
Иначе
 Сообщить("Документ """ + Строка(Документ1С) + """ выгружен на ТСД.");
КонецЕсли;
```

В наших конфигурациях обычно используется следующий алгоритм выгрузки строк документа: (метод «установитьполе» равен в нашем случае команде «SetField»).

Пример кода 1С:Предприятие 8

```
Для каждого строкаРекв из СтрокаФормата.Реквизиты Цикл // Тут у нас определена сущность
«строка формата», которая как раз и определяет, какие реквизиты документа 1С мы передаём
в реквизиты MS
Если Не строкаРекв.Выгружать Тогда
 Продолжить; // не выгружаем ненужные реквизиты
КонецЕсли;



    значение = РекДок[строкаРекв.Реквизит];
Если не ТипЗначенияДоступенВСМАРТС(значение) Тогда
 значение = Строка(значение);
 значение = ?(значение = "<>","",значение);
КонецЕсли;
 Если ЗначениеЗаполнено(значение) Тогда
 ДокументТСД.УстановитьПоле(строкаРекв.ПолеДокументаТСД, значение);
 // Метод УстановитьПоле(«НаименованиеПоляДокумента», «ПередаваемоеЗначение»)
 КонецЕсли;



    КонецЦикла;
```

Пример кода 1С:Предприятие 7.7

```
Для каждого строкаРекв из СтрокаФормата.Реквизиты Цикл // Тут у нас определена сущность «строка формата», которая как раз и определяет, какие реквизиты документа 1С мы передаём в реквизиты MS
Если строкаРекв.Выгружать = 0 Тогда
 Продолжить; // не выгружаем ненужные реквизиты
КонецЕсли;



    значение = РекДок[строкаРекв.Реквизит];
Если ТипЗначенияДоступенВСМАРТС(значение) = 0 Тогда
 значение = Строка(значение);
 значение = ?(значение = "<>","",значение);
КонецЕсли;
 Если ПолучитьПустоеЗначение(значение) = 0 Тогда
 ДокументТСД.УстановитьПоле(строкаРекв.ПолеДокументаТСД, значение);
 // Метод УстановитьПоле(«НаименованиеПоляДокумента», «ПередаваемоеЗначение»)
 КонецЕсли;


    КонецЦикла;
```

После выгрузки шапки документа (заголовок, организация, код ЕГАИС организации, и т.д.), мы можем приступить к выгрузке непосредственно строк документа. Необходимо уточнить, что структура базы данных Mobile SMARTS построена следующим образом – существуют «плановые» строки, которые выгружаются непосредственно из 1С, а после считывания штрих-кода с товара, в документ Mobile SMARTS добавляются строки «факта», которые затем мы с вами и будем обрабатывать.

Выгружаем плановые строки:

Пример кода 1С:Предприятие 8

```
 Для каждого строкаДок из МассивСтрок Цикл


    Товар = Неопределено;
СтрокаДокументаТСД = Новый COMОбъект("Cleverence.Warehouse.DocumentItem");
 КонецЦикла;
```

Пример кода 1С:Предприятие 7.7

```
 Для каждого строкаДок из МассивСтрок Цикл


    Товар = СтрокаТовара.Номенклатура.Код;
СтрокаДокументаТСД = СоздатьОбъект("Cleverence.Warehouse.DocumentItem");
 КонецЦикла;
```

Если есть возможность выгрузить диапазоны справок А, то используется следующий метод:

Пример кода 1С:Предприятие 8

```
ТаблицаСправокА = ДокументТСД.Таблицы.ДобавитьЭлемент ();
ТаблицаСправокА.Name = "ФормыА";
мДрайверТСД.ВыгрузитьДопТаблицу(ТаблицаСправокА);
Для Каждого СтрокаСправок из Документ1С.Товары Цикл
НоваяСтрокаТаблицы = ТаблицаСправокА.Строки.ДобавитьЭлемент ();
НоваяСтрокаТаблицы.SetField("КодФормы",СтрокаСправок.СправкаА.Код);
НоваяСтрокаТаблицы.SetField("Начало",СтрокаСправок.СправкаА.ДиапазоныНомеров[0].Начальн
НоваяСтрокаТаблицы.SetField("Конец",СтрокаСправок.СправкаА.ДиапазоныНомеров[0].Конечны
НоваяСтрокаТаблицы.SetField("АлкоКод",СтрокаСправок.АлкогольнаяПродукция.Код);
НоваяСтрокаТаблицы.SetField("КодНоменклатуры",ОпределитьНоменклатуруНаСервере(Строка
КонецЦикла;
```

Пример кода 1С:Предприятие 7.7


```
ТаблицаСправокА = ДокументТСД.Таблицы.ДобавитьЭлемент ();
ТаблицаСправокА.Name = "ФормыА";
мДрайверТСД.ВыгрузитьДопТаблицу(ТаблицаСправокА);
Для Каждого СтрокаСправок из Документ1С.Товары Цикл
НоваяСтрокаТаблицы = ТаблицаСправокА.Строки.ДобавитьЭлемент ();
НоваяСтрокаТаблицы.SetField("КодФормы",СтрокаСправок.СправкаА.Код);
НоваяСтрокаТаблицы.SetField("Начало",СтрокаСправок.СправкаА.ДиапазоныНомеров[0].Начальн
НоваяСтрокаТаблицы.SetField("Конец",СтрокаСправок.СправкаА.ДиапазоныНомеров[0].Конечны
НоваяСтрокаТаблицы.SetField("АлкоКод",СтрокаСправок.АлкогольнаяПродукция.Код);
НоваяСтрокаТаблицы.SetField("КодНоменклатуры",ОпределитьНоменклатуруНаСервере(Строка
КонецЦикла;
```


Поле	Тип	Основное/дополнительное	Описание
КодФормы	String	дополнительное	Код формы в системе ЕГАИС.
КодНоменклатуры	String	дополнительное	Уникальный идентификатор номенклатуры из справочника номенклатуры, если есть.
АлкоКод	String	дополнительное	Строка с кодом алкогольной продукции в ЕГАИС.

Начало		
Int32	дополнительное	Начало диапазона серийных номеров.
Конец		
Int32	дополнительное	Конец диапазона серийных номеров.

После того, как мы закончили выгружать документ, используется следующая конструкция:

Пример кода 1С:Предприятие 8

```
Ответ = мОбъектТСД.ВыгрузитьДокумент(ДокументТСД);  Если Не Ответ Тогда
НомерОшибки = мОбъектТСД.ПолучитьОшибку(мОбъектТСД.ОписаниеОшибки);
        Сообщить("Ошибка: " + мОбъектТСД.ОписаниеОшибки, СтатусСообщения.Важное);
Иначе
        Сообщить("Документ """ + Строка(Документ1С) + """ выгружен на ТСД.",
        СтатусСообщения.Обычное);
КонецЕсли;
```

Пример кода 1С:Предприятие 7.7

```
Ответ = мОбъектТСД.ВыгрузитьДокумент(ДокументТСД);  Если Ответ = 0 Тогда
НомерОшибки = мОбъектТСД.ПолучитьОшибку(мОбъектТСД.ОписаниеОшибки);
        Сообщить("Ошибка: " + мОбъектТСД.ОписаниеОшибки);
Иначе
        Сообщить("Документ """ + Строка(Документ1С) + """ выгружен на ТСД);
КонецЕсли;
```

Таким образом мы передали информацию о том, что документ выгружен, либо, если произошла ошибка, получим её описание.

Начало работы с дополнительной таблицей

Для доступа к доп. таблице необходимо запросить специальный объект для работы с ней.

GetTableAccessor(string tableName)

Параметр	Тип	Описание
tableName	String	Имя таблицы.

Пример кода для произвольной учетной системы через компоненту COM

```
var formTable = connection.GetTableAccessor("ФормыА");
```

Далее работа с таблицей ведется через этот объект. Выгрузка производится аналогично выгрузке

номенклатуры.

Начать выгрузку таблицы

BeginUpload(bool overwriteAllTable)

Параметр	Тип	Описание
overwriteAllTable	bool	Флаг, определяющий полностью перезаписывать всю таблицу или слить с выгруженной ранее.

Пример кода для произвольной учетной системы через компоненту COM

```
var formTable = connection.GetTableAccessor("ФормыА");
formTable.BeginUpload(true);
```

Выгрузка строк таблицы

Upload(RowCollection rows)

Параметр	Тип	Описание
rows	RowCollection	Коллекция строк таблицы для выгрузки.

Функция принимает для выгрузки коллекцию строк таблицы (Row).

Пример кода для произвольной учетной системы через компоненту COM

```
// создание коллекции строк таблицы
var rowCol = new COM("Cleverence.Warehouse.RowCollection");
// создание строки таблицы
var row = new COM("Cleverence.Warehouse.Row");
row.SetField("КодФормы", код_формы);
row.SetField("КодНоменклатуры", код_номенклатуры);
row.SetField("АлкоКод", "0345345345435");
row.SetField("Начало", 10234955554);
row.SetField("Конец",  10234955800);
//добавление строки в коллекцию
rowCol.Add(row);
//...
//заполнение строк
//...
//выгрузка коллекции строк
formTable.Upload(rowCol);
```

Закончить выгрузку таблицы

EndUpload()

```
formTable.EndUpload();
```

Завершает процедуру выгрузки строк таблицы. После её вызова сервер будет считать выгрузку завершенной и начнет у себя обновление таблицы.

До вызова этой функции новые строки не считаются выгруженными и недоступны.

Полный цикл Пример кода для произвольной учетной системы через компоненту COM

```
connection = new COM("Cleverence.Warehouse.StorageConnector");
// СтрокаПодключения - строка подключения из настройки базы MS
connection.SelectCurrentApp(СтрокаПодключения);
var formTable = connection.GetTableAccessor("ФормыА");
//начало выгрузки
//полная выгрузка, с переписыванием всей таблицы на сервере
formTable.BeginUpload(true);
// создание коллекции строк таблицы
var rowCol = new COM("Cleverence.Warehouse.RowCollection");
for(int i = 0; i< колво_форм_А; i++) //цикл по формам А в системе
{
 //выгружаем блоками по 500 строк
 if(rowCol.Count == 500)
 {
  //выгрузка блока из 500 товаров
  formTable.Upload(rowCol);
  rowCol = new COM("Cleverence.Warehouse.RowCollection");
 }
 // создание строки таблицы
 var row = new COM("Cleverence.Warehouse.Row");
 row.SetField("КодФормы", код_формы);
 row.SetField("КодНоменклатуры", код_номенклатуры);
 row.SetField("АлкоКод", алко_код);
 row.SetField("Начало", начало_интервала_серийных_номеров);
 row.SetField("Конец",  конец_интервала_серийных_номеров);
 //добавление строки в коллекцию
 rowCol.Add(row);
}
//выгрузка оставшихся товаров
if (rowCol.Count > 0)
 formTable.Upload(rowCol);
//завершение выгрузки
formTable.EndUpload();
```

Этап четвертый. Загрузка документов

Загрузка документов аналогична выгрузке, исключение составляет лишь несколько моментов:

1. **Для чтения полей читаем строки факта.**

```
Для Каждого СтрокаТаблицыДокумента из ДокументТСД["СтрокиФакт"]
```

2. **Поля читаем командой «GetField», либо «ПолучитьПоле».**
3. **После того, как мы загрузили документ, обработали его определенным образом, необходимо выполнить команду очистки списка документов.**

```
мОбъектТСД.Драйвер.УдалитьДокументы(«Список документов»);
```

Пример Б – загрузка с ТСД документа «Ввод начальных остатков» (1С:Предприятие 8)

```
// Демонстрационный пример для формирования документа "сбор начальных остатков"
```

```
// Демонстрационный пример для формирования документа "Сбор начальных остатков"
// Создаём таблицу значений, и загружаем в неё данные, полученные с терминала
// Тут мы можем перебрать все документы, которые были выгружены на терминал.
// Внимание - у документов есть несколько статусов, доступных через свойства
// ДокументыТерминалов.Изменен  - boolean; // ДокументыТерминалов.Завершен - boolean;
// Если для вас небходимо собирать только завершённые документы, пользуйтесь значениями этих полей.
// После чего можем сохранить документы // Единственное, что необходимо типизировать таблицу значений перед тем, как загружать в неё данные
 ТаблицаДанныхСТерминала = Новый ТаблицаЗначений;
 ТаблицаДанныхСТерминала.Колонки.Добавить("CreatedBy");
 ТаблицаДанныхСТерминала.Колонки.Добавить("ProductID");
 ТаблицаДанныхСТерминала.Колонки.Добавить("DeclaredQuantity");
 ТаблицаДанныхСТерминала.Колонки.Добавить("CurrentQuantity");
 ТаблицаДанныхСТерминала.Колонки.Добавить("FirstCellID");
 ТаблицаДанныхСТерминала.Колонки.Добавить("FirstStorageBarcode");
 ТаблицаДанныхСТерминала.Колонки.Добавить("PackingID");
 ТаблицаДанныхСТерминала.Колонки.Добавить("SSCC");
 ТаблицаДанныхСТерминала.Колонки.Добавить("Index");
 ТаблицаДанныхСТерминала.Колонки.Добавить("RegisteredDate");
 ТаблицаДанныхСТерминала.Колонки.Добавить("RegistrationDate");
 ТаблицаДанныхСТерминала.Колонки.Добавить("ExpiredDate");
 ТаблицаДанныхСТерминала.Колонки.Добавить("SecondCellID");
 ТаблицаДанныхСТерминала.Колонки.Добавить("SecondStorageBarCode");
 ТаблицаДанныхСТерминала.Колонки.Добавить("BindedLine");
 ТаблицаДанныхСТерминала.Колонки.Добавить("code");
 ТаблицаДанныхСТерминала.Колонки.Добавить("barcode");
 ТаблицаДанныхСТерминала.Колонки.Добавить("serial");
 ТаблицаДанныхСТерминала.Колонки.Добавить("desc");
 ТаблицаДанныхСТерминала.Колонки.Добавить("sn");
 ТаблицаДанныхСТерминала.Колонки.Добавить("price");
 ТаблицаДанныхСТерминала.Колонки.Добавить("Ячейка");
 ТаблицаДанныхСТерминала.Колонки.Добавить("Заблокировано");
 ТаблицаДанныхСТерминала.Колонки.Добавить("ЦенаСклад");
 ТаблицаДанныхСТерминала.Колонки.Добавить("Алко");
 ТаблицаДанныхСТерминала.Колонки.Добавить("ПроверкаЧМ");
 ТаблицаДанныхСТерминала.Колонки.Добавить("АлкоСН");
 ТаблицаДанныхСТерминала.Колонки.Добавить("АлкоКод");
 ТаблицаДанныхСТерминала.Колонки.Добавить("АлкоНаим");
 ТаблицаДанныхСТерминала.Колонки.Добавить("АлкоКодВ");
 ТаблицаДанныхСТерминала.Колонки.Добавить("АлкоОбъем");
 ТаблицаДанныхСТерминала.Колонки.Добавить("АлкоКрепость");
 ТаблицаДанныхСТерминала.Колонки.Добавить("Производитель");
 ТаблицаДанныхСТерминала.Колонки.Добавить("ПроизвИНН");
 ТаблицаДанныхСТерминала.Колонки.Добавить("ПроизвКПП");
 ТаблицаДанныхСТерминала.Колонки.Добавить("АлкоПДФ");
 Для инд = 0 По ДокументыТерминалов.Количество - 1 Цикл
ДокументТерминала = ДокументыТерминалов.Элемент(инд);
  // Все документы сбора начальных остатков мы можем собрать в одну ТЗ
Если ДокументыТерминалов.Элемент(инд).ИмяТипаДокумента = "Сбор начальных остатков" Тогда
   СтрокиДокументаПлан = ДокументТерминала.СтрокиПлан;
```

```
  КоличествоСтрокПлан = СтрокиДокументаПлан.Количество;
  СтрокиДокументаФакт = ДокументТерминала.СтрокиФакт;
  КоличествоСтрокФакт = СтрокиДокументаФакт.Количество;
  Для СтрокаФакт = 0 по СтрокиДокументаФакт.Количество - 1 Цикл
  ТекущаяСтрокаДокумента = СтрокиДокументаФакт.Элемент(СтрокаФакт);
   СтрокаТЗ = ТаблицаДанныхСТерминала.Добавить();
   Для Каждого Колонка из ТаблицаДанныхСТерминала.Колонки Цикл
СтрокаТЗ[Колонка.Имя] = ТекущаяСтрокаДокумента.ПолучитьПоле(Колонка.Имя);
   КонецЦикла;
  КонецЦикла;
 КонецЕсли;



    КонецЦикла;
// Всё, таблица готова для дальнейшей обработки в 1С
```

Пример Б – загрузка с ТСД документа «Ввод начальных остатков» (1С:Предприятие 7.7)

```
// Демонстрационный пример для формирования документа "сбор начальных остатков"
// Создаём таблицу значений, и загружаем в неё данные, полученные с терминала
// Тут мы можем перебрать все документы, которые были выгружены на терминал.
// Внимание - у документов есть несколько статусов, доступных через свойства
// ДокументыТерминалов.Изменен  - boolean; // ДокументыТерминалов.Завершен - boolean;
// Если для вас небходимо собирать только завершённые документы, пользуйтесь значениями
этих полей.
// После чего можем сохранить документы // Единственное, что необходимо типизировать
таблицу значений перед тем, как загружать в неё данные
 ТаблицаДанныхСТерминала = СоздатьОбъект(“ТаблицаЗначений”);
 ТаблицаДанныхСТерминала.НоваяКолонка("CreatedBy");
 ТаблицаДанныхСТерминала.НоваяКолонка("ProductID");
 ТаблицаДанныхСТерминала.НоваяКолонка("DeclaredQuantity");
 ТаблицаДанныхСТерминала.НоваяКолонка("CurrentQuantity");
 ТаблицаДанныхСТерминала.НоваяКолонка("FirstCellID");
 ТаблицаДанныхСТерминала.НоваяКолонка("FirstStorageBarcode");
 ТаблицаДанныхСТерминала.НоваяКолонка("PackingID");
 ТаблицаДанныхСТерминала.НоваяКолонка("SSCC");
 ТаблицаДанныхСТерминала.НоваяКолонка("Index");
 ТаблицаДанныхСТерминала.НоваяКолонка("RegisteredDate");
 ТаблицаДанныхСТерминала.НоваяКолонка("RegistrationDate");
 ТаблицаДанныхСТерминала.НоваяКолонка("ExpiredDate");
 ТаблицаДанныхСТерминала.НоваяКолонка("SecondCellID");
 ТаблицаДанныхСТерминала.НоваяКолонка("SecondStorageBarCode");
 ТаблицаДанныхСТерминала.НоваяКолонка("BindedLine");
 ТаблицаДанныхСТерминала.НоваяКолонка("code");
 ТаблицаДанныхСТерминала.НоваяКолонка("barcode");
 ТаблицаДанныхСТерминала.НоваяКолонка("serial");
 ТаблицаДанныхСТерминала.НоваяКолонка("desc");
 ТаблицаДанныхСТерминала.НоваяКолонка("sn");
 ТаблицаДанныхСТерминала.НоваяКолонка("price");
 ТаблицаДанныхСТерминала.НоваяКолонка("Ячейка");
 ТаблицаДанныхСТерминала.НоваяКолонка("Заблокировано");
 ТаблицаДанныхСТерминала.НоваяКолонка("ЦенаСклад");
```

```
ТаблицаДанныхСТерминала.НоваяКолонка("Алко");
ТаблицаДанныхСТерминала.НоваяКолонка("ПроверкаЧМ");
ТаблицаДанныхСТерминала.НоваяКолонка("АлкоСН");
ТаблицаДанныхСТерминала.НоваяКолонка("АлкоКод");
ТаблицаДанныхСТерминала.НоваяКолонка("АлкоНаим");
ТаблицаДанныхСТерминала.НоваяКолонка("АлкоКодВ");
ТаблицаДанныхСТерминала.НоваяКолонка("АлкоОбъем");
ТаблицаДанныхСТерминала.НоваяКолонка("АлкоКрепость");
ТаблицаДанныхСТерминала.НоваяКолонка("Производитель");
ТаблицаДанныхСТерминала.НоваяКолонка("ПроизвИНН");
ТаблицаДанныхСТерминала.НоваяКолонка("ПроизвКПП");
ТаблицаДанныхСТерминала.НоваяКолонка("АлкоПДФ");
Для инд = 0 По ДокументыТерминалов.Количество - 1 Цикл
ДокументТерминала = ДокументыТерминалов.Элемент(инд);
 // Все документы сбора начальных остатков мы можем собрать в одну ТЗ
Если ДокументыТерминалов.Элемент(инд).ИмяТипаДокумента = "Сбор начальных остатков"
Тогда
  СтрокиДокументаПлан = ДокументТерминала.СтрокиПлан;
  КоличествоСтрокПлан = СтрокиДокументаПлан.Количество;
  СтрокиДокументаФакт = ДокументТерминала.СтрокиФакт;
  КоличествоСтрокФакт = СтрокиДокументаФакт.Количество;
  Для СтрокаФакт = 0 по СтрокиДокументаФакт.Количество - 1 Цикл
  ТекущаяСтрокаДокумента = СтрокиДокументаФакт.Элемент(СтрокаФакт);
   СтрокаТЗ = ТаблицаДанныхСТерминала.Добавить();
   Для Каждого Колонка из ТаблицаДанныхСТерминала.Колонки Цикл
СтрокаТЗ[Колонка.Имя] = ТекущаяСтрокаДокумента.ПолучитьПоле(Колонка.Имя);
   КонецЦикла;
  КонецЦикла;
 КонецЕсли;



    КонецЦикла;
// Всё, таблица готова для дальнейшей обработки в 1С
```

Получение документов из Mobile SMARTS

Document **GetDocument(**string **id)**

Параметр	Тип	Описание
id	string	Идентификатор документа для получения. Если такой документ не найден будет возвращен null.

DocumentCollection **GetDocuments(**string **docType,** bool **checkForFinish)**

Параметр	Тип	Описание
docType	string	Имя типа документов для возврата. Если передана пустая строка, то будут возвращены документы всех типов.
checkForFinish	bool	Флаг, указывающий возвращать только завершенные документы или все подряд.

В отличие от выгрузки, в загруженном документе извлекать данные следует не из строк плана, а из фактических строк CurrentItems.

Пример кода для произвольной учетной системы через компоненту COM

```
connection = new COM("Cleverence.Warehouse.StorageConnector");
connection.SelectCurrentApp(СтрокаПодключения);
//запрашиваем завершенные документы всех типов
var docs = connection.GetDocuments("", true);
for(int i = 0; i< docs.Count; i++) //цикл по полученным документам
{
 var doc = docs.Item(i);
// var doc = docs[i];
 for(int j = 0; j< doc.CurrentItems.Count; j++) //цикл по строкам факт документа
 {
  var dItem = doc.CurrentItems.Item(j);
  //запрос полей из строки
  string prodId = dItem.GetField("ProductId"); //код товара
  string packId = dItem.GetField("PackingId"); //код упаковки
  double qty =  dItem.GetField("CurrentQuantity"); //факт кол-во в строке
  string alcoCode = dItem.GetField("АлкоКод"); //код упаковки
  //...
  //получение остальных полей, согласно таблице доступных полей
  //...
 }
}
```

Поля строк документа для получения из фактических строк

Поле	Тип	Основное/доп.	Описание
ProductID	String	основное	Идентификатор товара, для которого описана данная позиция.
PackingID	String	основное	Идентификатор упаковки для заданного товара. Товар задается свойством ProductId [ИдТовара].
CurrentQuantity	Double	основное	Фактическое количество товара в заданном виде упаковки.
descr	String	дополнительное	Характеристика товара (если ведется учет с характеристиками).
serial	String	дополнительное	Серия товара (если используется учет по сериям).
price	Decimal	дополнительное	Цена единицы товара в строке.
Алко	Boolean	дополнительное	Признак того, что товар является алкогольной или спиртосодержащей продукцией.
АлкоКод	String	дополнительное	Код алкогольной продукции в ЕГАИС, полученный из отсканированных марок.
АлкоПДФ	String	дополнительное	Строка с PDF 417.
АлкоСН	string	дополнительное	Серийный номер бутылки, если сканировался.
Проверка ЧМ	Boolean	дополнительное	Признак того, что данная конкретная

ПроизводительИНН	Производитель прошел проверку в CheckMark на легальность.
String	
дополнительное	
ПроизводительКПП	Наименование производителя продукции в ЕГАИС, если уже известно для данного товара, либо если было получено из CheckMark.
String	
дополнительное	
АлкоНаим	ИНН производителя, если уже известен для данного товара, либо если был получен из CheckMark.
String	
дополнительное	
АлкоКодВ	КПП производителя, если уже известен для данного товара, либо если был получен из CheckMark.
String	
дополнительное	
АлкоОбъем	Наименование товара из ЕГАИС, если уже известно для данного товара, либо если было получено из CheckMark.
Decimal	
дополнительное	
АлкоКрепость	Код вида алкогольной продукции из ЕГАИС, если уже известен для данного товара, либо если был получен из CheckMark.
Decimal	
дополнительное	
АлкоЕстьВФормах	Ёмкость тары в литрах из ЕГАИС, если уже известна для данного товара, либо если была получена из CheckMark.
Boolean	
дополнительное	
КодФормыА	Процентное содержание спирта из ЕГАИС, если уже известно для данного товара, либо если было получен из CheckMark.
String	
дополнительное	
ДатаРозлива	Признак того, что данная марка была найдена в выгруженных формах А.
DateTime	
дополнительное	
	Код формы А, где была найдена марка.
	Дата розлива позиции, если вводилась, иначе null.

Удаление документа

После успешной загрузки его необходимо удалить из Mobile SMARTS, иначе при следующем запрсое он опять будет возвращен.

RemoveDocument(string documentId)

Параметр	Тип	Описание
id	string	Идентификатор документа для удаления.

Пример кода для произвольной учетной системы через компоненту COM

```
connection = new COM("Cleverence.Warehouse.StorageConnector");
connection.SelectCurrentApp(СтрокаПодключения);
//удаление конкретного документа по его идентификатору
connection.RemoveDocument(doc.Id);
```

ЕГАИС, интеграция, инструкции, 1С

Не нашли что искали?

Задать вопрос в техническую поддержку

Промежуточная конфигурация «1С: Предприятия»

Последние изменения: 2024-03-26

> Промежуточная конфигурация 1С есть **только в драйвере ПРОФ!** В других драйверах она не требуется! Промежуточная конфигурация — специальная конфигурация, в которой собраны процедуры и функции (не данные!!) для запроса данных из рабочей базы 1С при онлайн работе мобильного устройства в базой 1С в драйвере ПРОФ.

Зачем нужна промежуточная конфигурация:

- В стандартных конфигурациях 1С нет поддержки ТСД при онлайн работе (поиск товаров, выборка остатков и цен, вызов печати), а в некоторых вообще нет поддержки торгового оборудования. Нужные процедуры, функции и запросы отсутствуют, и ТСД попросту нечего позвать из 1С. Драйвер ПРОФ от Клеверенс продается как полностью рабочий коробочный продукт, а не только программа на ТСД. Чтобы расширить функционал стандартной конфигурации без снятия её с поддержки, в составе драйвера ПРОФ Клеверенс поставляет бесплатную промежуточную конфигурацию, которая гоняет данные между рабочей базой 1С и ТСД. Это примерно то же самое, что и БПО (Библиотека подключаемого оборудования), только для Wi-Fi ТСД. Все функции для онлайн работы с данными полностью реализованы в этой отдельной (промежуточной) конфигурации. Без этой конфигурации не было бы возможности за 5 мин подключить и протестировать работу ТСД в онлайн обмене с рабочей базой 1С. А такие вещи, как модуль автоматической загрузки/выгрузки документов, было бы довольно долго разрабатывать самостоятельно.
- Доработка стандартной конфигурации, добавление в неё функций обмена с ТСД, снимает её с поддержки. Промежуточная конфигурация от Клеверенс содержит только логику и настройки обмена. Она может обновляться и дорабатываться независимо от конфигурации рабочей базы, в которой ведется основной учет. Рабочая база не снимается с поддержки и может обновляться без особых проблем (проблем будет не больше, чем было бы вообще без каких-то там ТСД). Это может быть очень важно для тех клиентов, у которых нет своего штата 1С-ников.
- Чтобы использовать её как источник готового кода 1С. Функционал промежуточной конфигурации, при желании, всегда можно полностью перенести в конфигурацию своей рабочей базы. Промежуточная конфигурация полностью открыта. Единственное следует помнить, что при таком переносе обе конфигурации снимаются с поддержки (и основная конфигурация 1С, и промежуточная конфигурация драйвера ПРОФ).

Схема взаимодействия при работе на ТСД

Промежуточная конфигурация не хранит никаких складских или товарных данных. Все данные всегда берутся ею из рабочей базы через OLE подключение.

- ТСД запрашивает у сервера терминалов Mobile SMARTS нужные ему данные;
- Сервер терминалов Mobile SMARTS отправляет онлайн вызов (запрос) в промежуточную конфигурацию 1С для получения (сохранения) данных;
- Промежуточная конфигурация запрашивает эти данные в рабочей базе

1С;

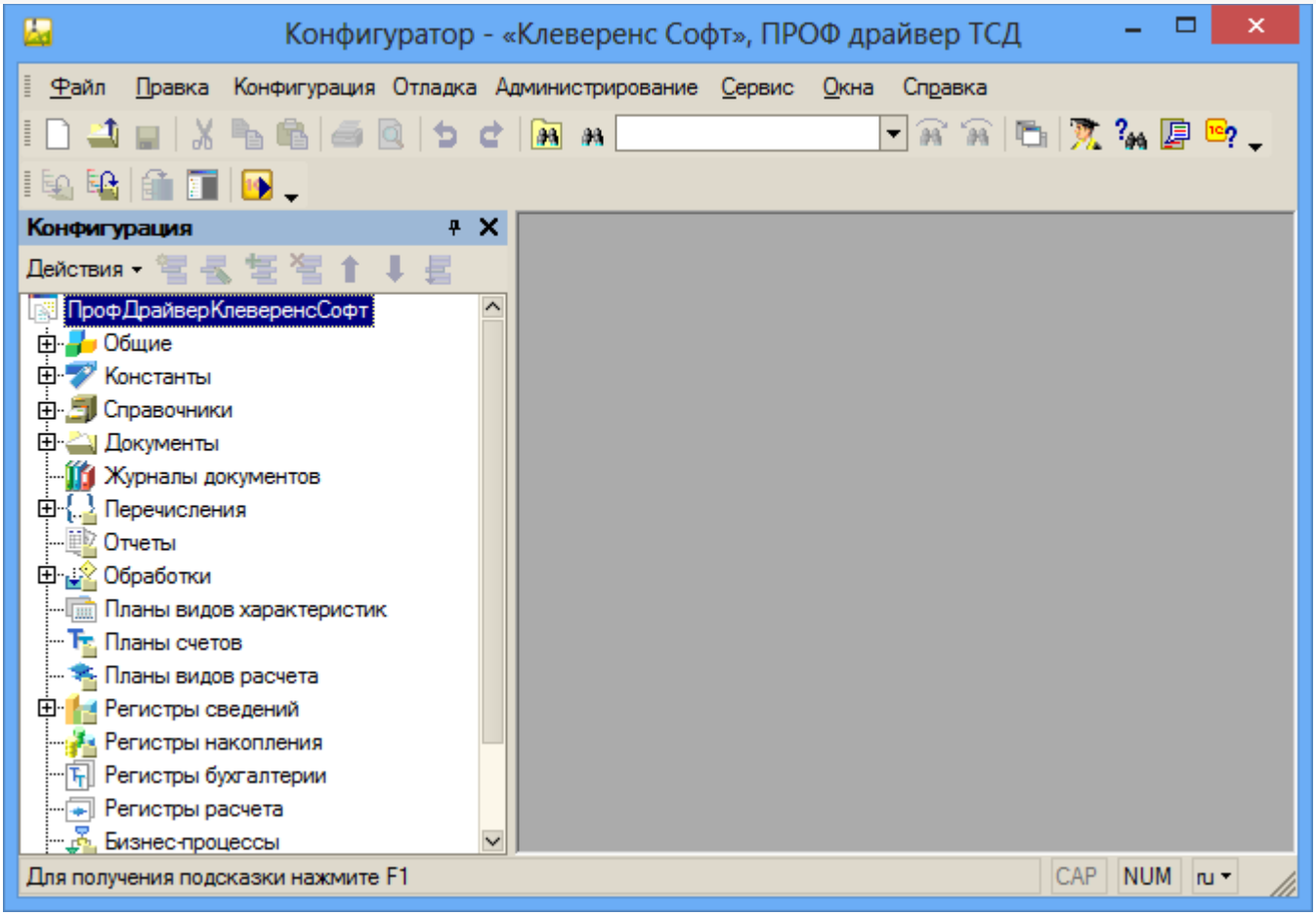
- Нужные данные выгружаются на сервер терминалов Mobile SMARTS;
- Сервер терминалов Mobile SMARTS отправляет данные на ТСД.

Установка промежуточной конфигурации

При установке драйвера регистрация промежуточной конфигурации происходит в 1С автоматически. После загрузки промежуточной базы автоматически откроется окно настройки онлайн соединения, и другого расширенного функционала ПРОФ драйвера (подробнее см. документацию к драйверу раздел «Настройка промежуточной базы 1С для онлайн подключения»). Предусмотрена возможность сохранять сделанные настройки промежуточной конфигурации в отдельный файл для последующего восстановления на случай, если необходимо будет перенести конфигурацию или она была нечаянно удалена или что-то перестало работать (подробнее см. Сохранение и восстановление настроек промежуточной конфигурации драйвера ПРОФ).

Конфигурирование

Все изменения в промежуточной конфигурации выполняются так же, как и в основной конфигурации 1С с помощью конфигуратора.



промежуточная конфигурация 1С, драйвер ПРОФ

Не нашли что искали?

Задать вопрос в техническую поддержку

Выгрузка справочника номенклатуры из «1С:Предприятие» в «Курьер 15»

Последние изменения: 2024-03-26

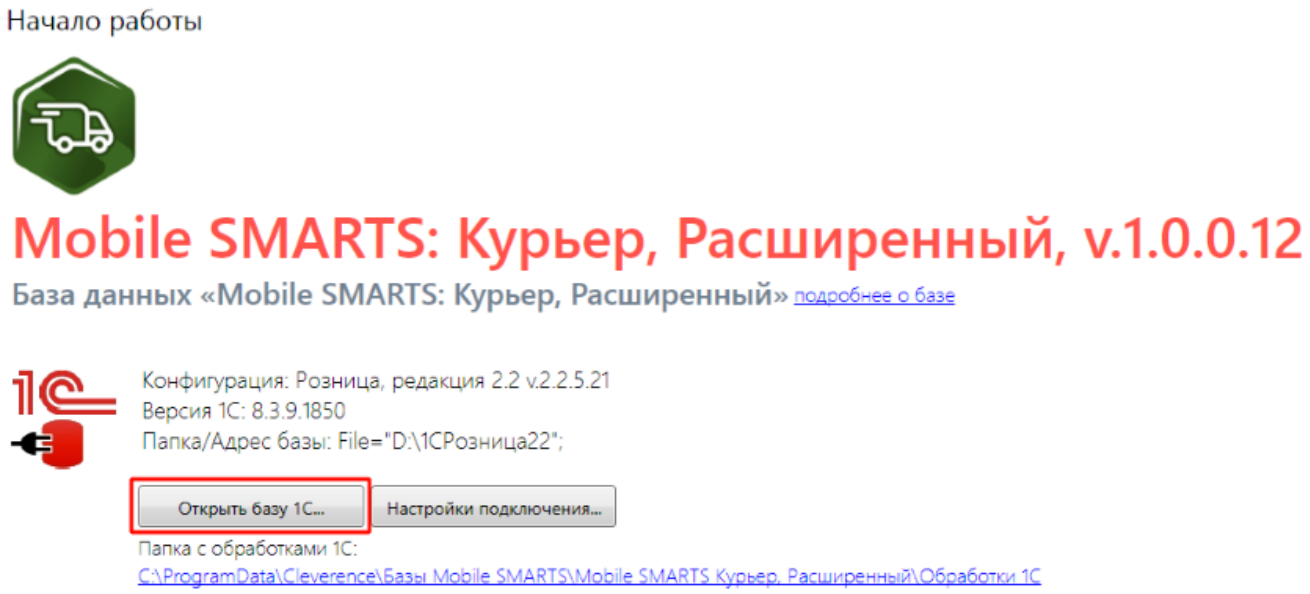
При выгрузке заказов из «1С: Предприятия» на мобильное устройство происходит и частичная выгрузка номенклатуры, только по товарным позициям выгружаемого заказа, но эта номенклатура не может быть использована для создания других заказов непосредственно на мобильном устройстве.

> **Отдельно справочник номенклатуры можно не выгружать на мобильное устройство. Номенклатура может быть выгружена вместе с документом.**

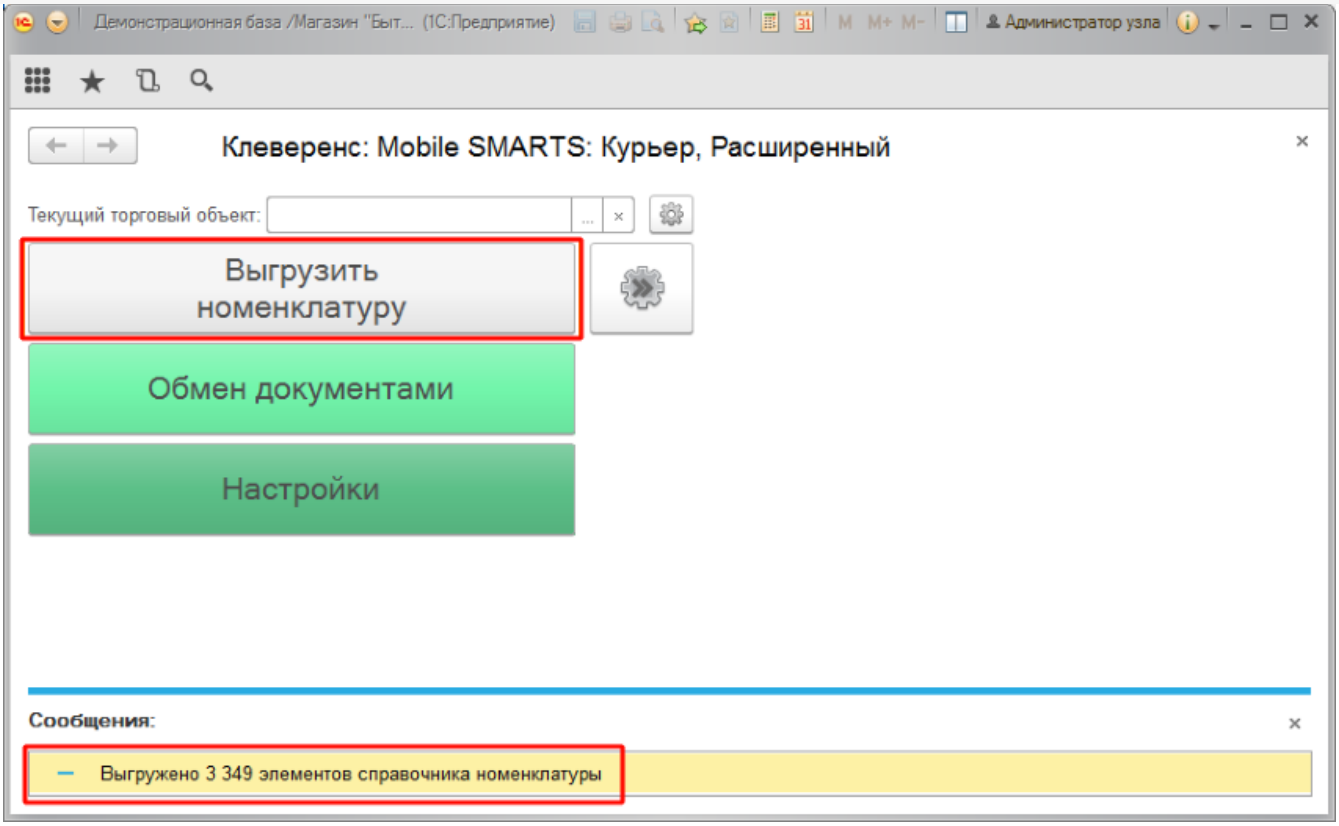
Выгрузка полного справочника номенклатуры позволяет курьеру создавать новые заказы на своем мобильном устройстве.

Для выгрузки справочника номенклатуры используйте обработку 1С идущую в комплекте поставки «Курьера 15».

Запуск обработки осуществляется через кнопку «Открыть базу 1С».



Используйте кнопку «Выгрузить номенклатуру».



По результатам работы в нижней части окна обработки будет выведен отчет по произведенной выгрузке.

Курьер, интеграция, 1С

Не нашли что искали?

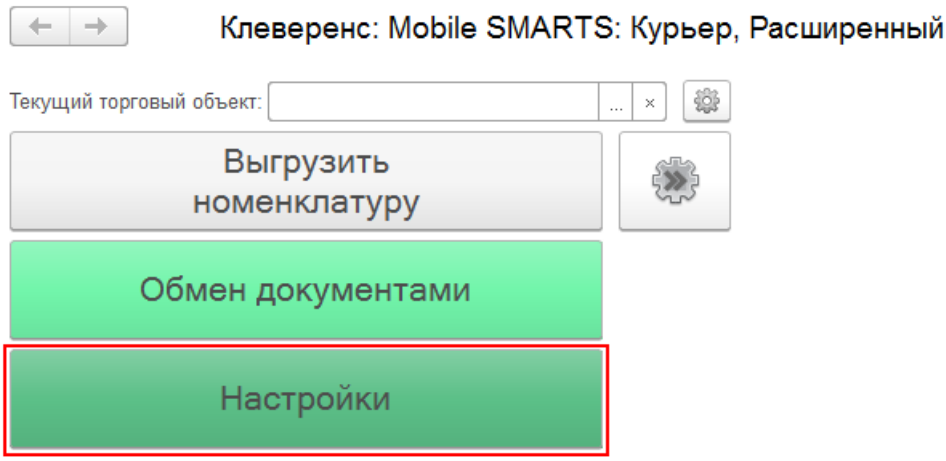
Задать вопрос в техническую поддержку

Подключение и настройка «Курьера 15» в «1С:Предприятие»

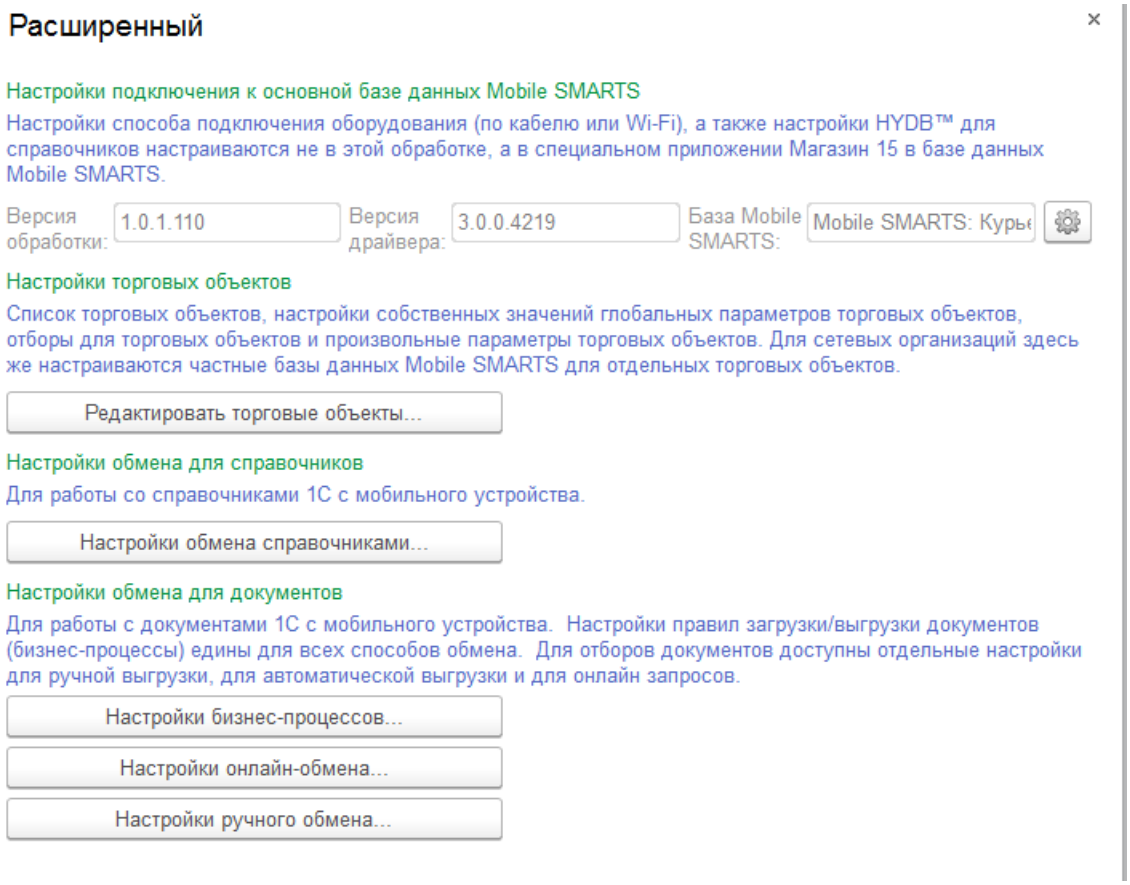
Последние изменения: 2024-03-26

Настройка интеграции с «1С: Предприятие» осуществляется с помощью обработки 1С, идущей в комплекте поставки «Курьера 15».

Для выполнения необходимых настроек необходимо запустить обработку, далее использовать кнопку «Настройки», расположенную в главном окне обработки 1С.



В открывшейся оснастке есть возможность произвести все необходимые настройки.



Настройки подключения к основной базе данных Mobile SMARTS подробно описаны в статье Подключение к базе данных Mobile SMARTS в обработке 1С.

Настройки торговых объектов производятся по аналогии с другими продуктами «Клеверенс» и подробно описаны в статье «Настройка торговых объектов в «Магазине 15»».

Настройки обмена справочников производятся по аналогии с другими продуктами «Клеверенс» и подробно описаны в статье «Настройки обмена справочников в «Магазине 15»».

Настройки бизнес-процессов производятся по аналогии с другими продуктами «Клеверенс» и подробно описаны в статье «Настройки бизнес-процессов».

Настройки онлайн-обмена производятся по аналогии с другими продуктами «Клеверенс» и подробно описаны в статье «Настройка онлайн обмена документами в «Магазине 15»».

Настройки ручного обмена производятся по аналогии с другими продуктами «Клеверенс» и подробно описаны в статье «Настройка ручного обмена документами в «Магазине 15»».

Курьер, интеграция, 1С

Не нашли что искали?

Задать вопрос в техническую поддержку

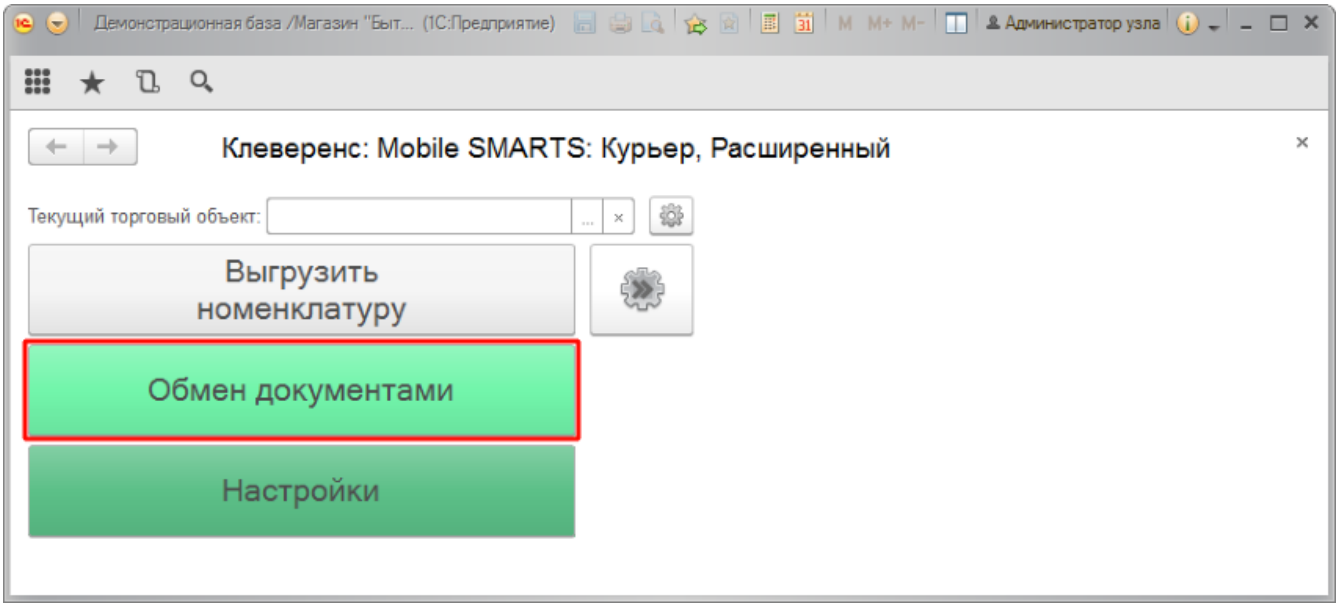
Выгрузка документов из «1С:Предприятие» на мобильное устройство в «Курьер 15»

Последние изменения: 2024-03-26

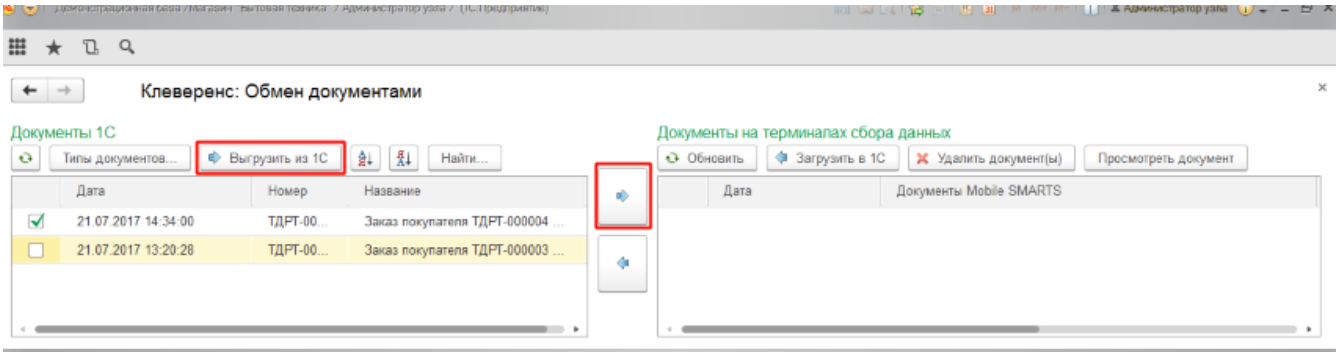
Для выгрузки документов (заказов) из «1С: Предприятия» на мобильное устройство используйте обработку 1С, идущую в комплекте поставки «Курьер».

Запуск обработки осуществляется через кнопку «Открыть базу 1С» в главном окне приложения «Курьер».

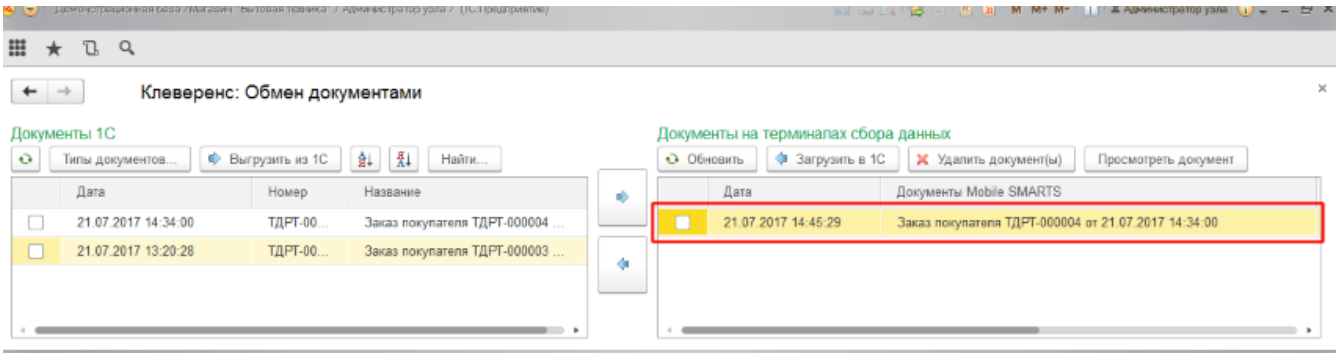
Для осуществления выгрузки документов (заказов) из 1С на мобильное устройство используйте кнопку «Обмен документами» в обработке 1С.



Для выгрузки выбранных документов на мобильное устройство используйте кнопки выгрузки. Отфильтровать выгружаемые документы по типу возможно с помощью кнопки «Типы документов...».



Выгруженные документы появятся в правой панели окна выгрузки.



Курьер, интеграция, 1С

Не нашли что искали?

Задать вопрос в техническую поддержку

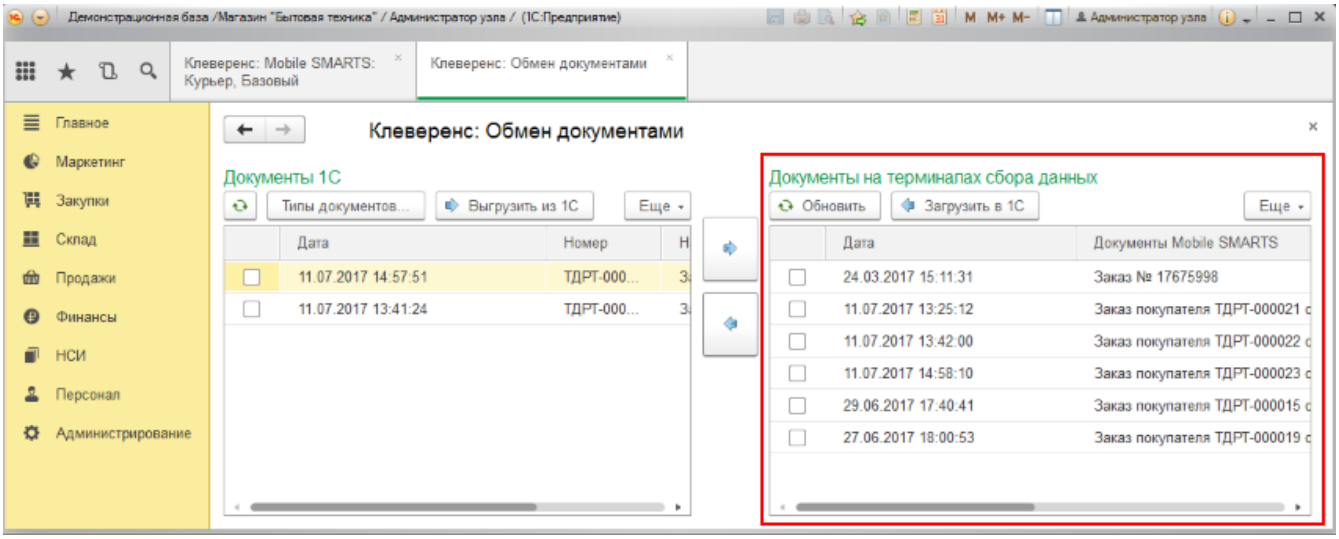
Загрузка документов в «1С:Предприятие» с мобильного устройства из «Курьера 15»

Последние изменения: 2024-03-26

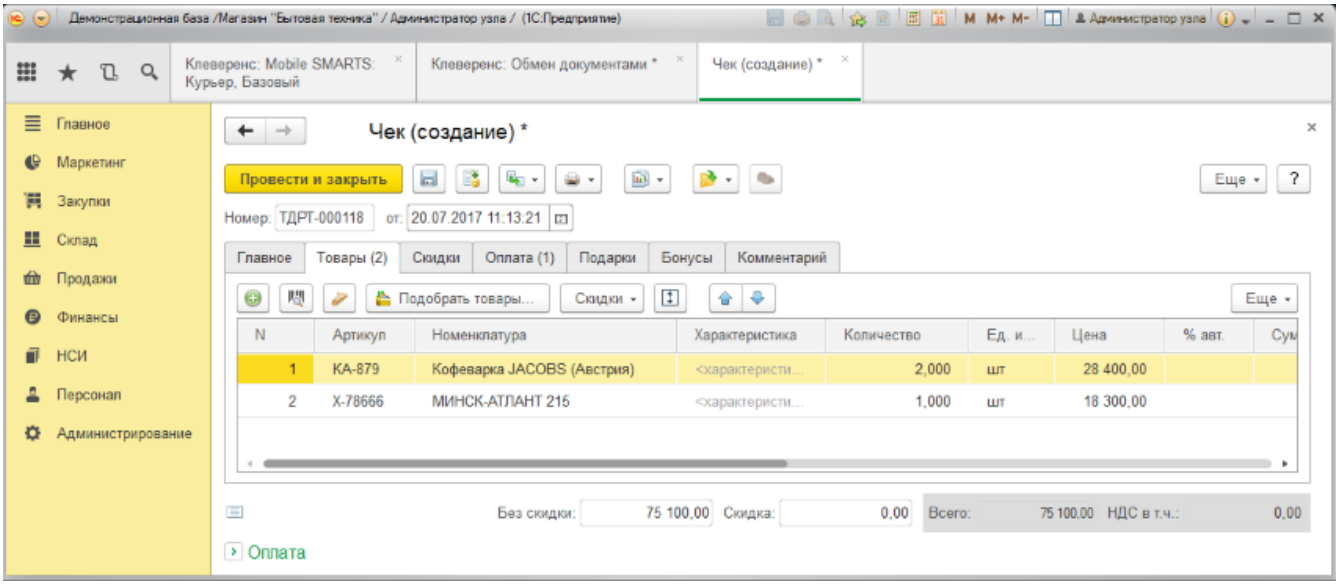
После запуска обработки она автоматически проверит подключение сервера (в батч режиме ТСД должен быть подключен проводом) и покажет список документов на нем. Также этот список можно получить вручную, по нажатию кнопки «Обновить».

Завершенные документы (как выгруженные, так и созданные непосредственно на мобильном устройстве) попадают на сервер Mobile SMARTS, загрузка документов в 1С выполняется с сервера.

При использовании варианта связи «Прямая связь с ТСД» загрузка выполняется напрямую с ТСД, подключаемого через ActiveSync или «Центр мобильных устройств».



Данные в 1С заполнятся автоматически на основе документа загруженного с мобильного устройства.



Курьер, интеграция, 1С, загрузка в 1С

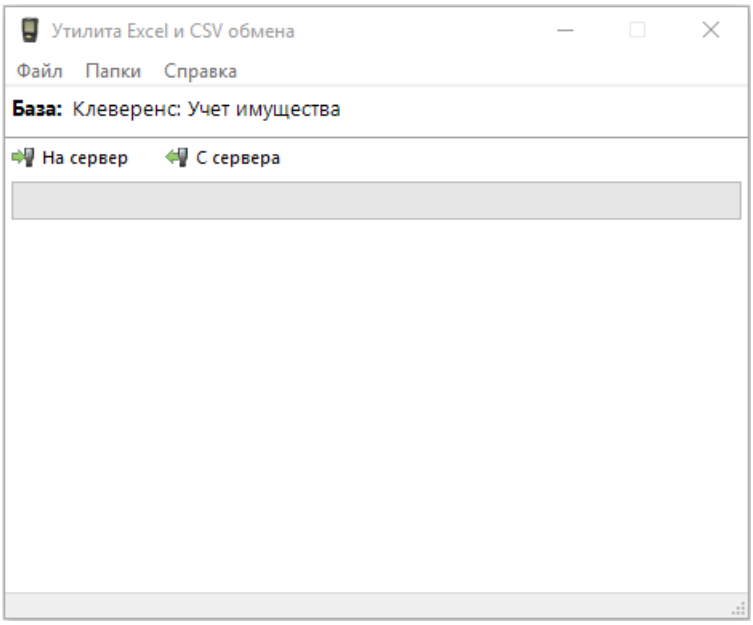
Не нашли что искали?

Задать вопрос в техническую поддержку

Ручной обмен с помощью утилиты файлового обмена (Excel/CSV)

Последние изменения: 2024-03-26

Утилита Excel и CSV обмена позволяет конвертировать документы из нескольких форматов в формат, понятный терминалу, загружать их на ТСД в виде заданий на исполнение, а также скачивать с терминала выполненные задания и конвертировать их обратно в нужный вам формат. В качестве программы для терминала используется клиентское приложение Mobile SMARTS.



Назначение кнопок программы следующее:

- **На сервер** — выполняет только первую половину цикла обмена: отправляет на терминал документы, но ничего с терминала не скачивает.
- **С сервера** — выполняет только вторую половину цикла обмена: забирает с терминала выполненные задания, но ничего на терминал не отправляет.

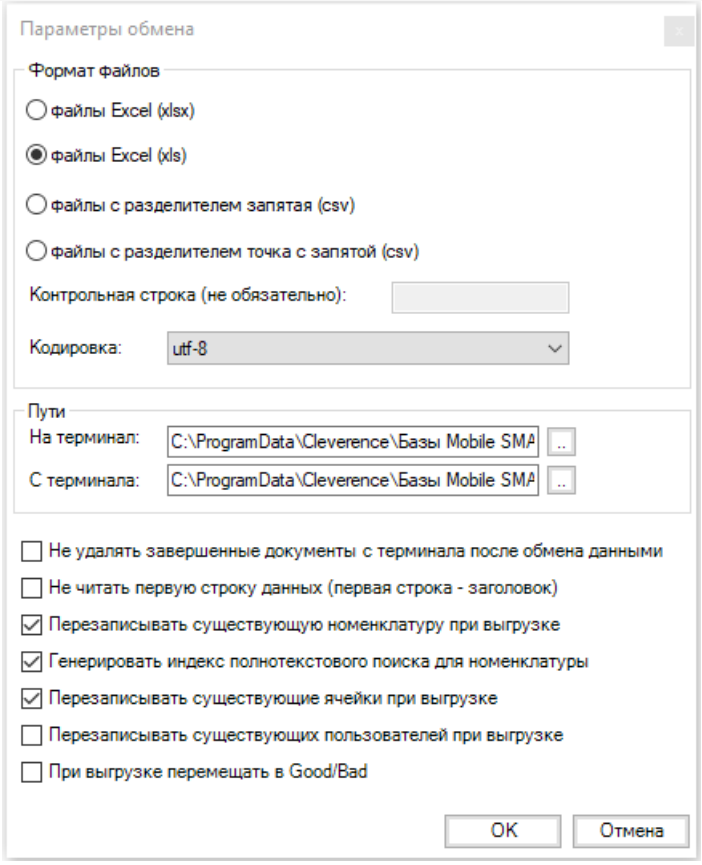
Для обмена документами и номенклатурой в программе предусмотрены папки «**На терминал**» и «**С терминала**», которые можно открыть через интерфейс программы (меню «Папки»). Программа поддерживает следующие форматы файлов:

1. **Файлы Excel (*.xls;*.xlsx)** заданного формата. Примеры документов лежат в папке «На терминал».
2. **CommaSeparatedValue (*.csv)** — простые текстовые файлы, в которых колонки отделены табуляцией, а строки — переводом строки.

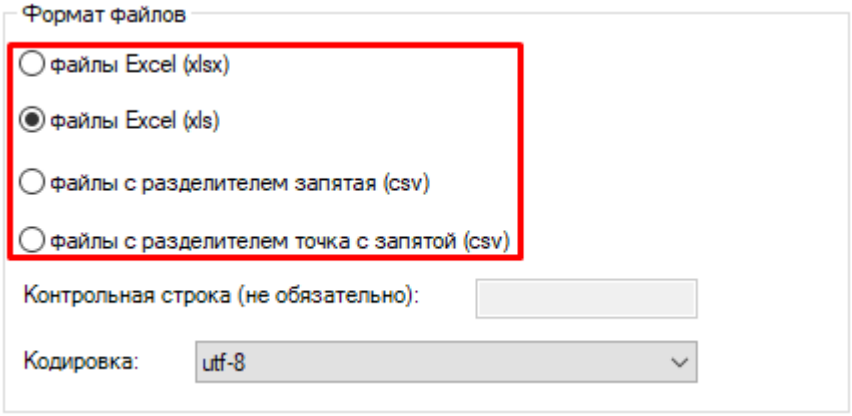
Для правильной работы программе необходимо иметь в папке «На терминал» файл с данными номенклатуры. Формат и наименование файла с номенклатурой жестко заданы, пример файла ставится вместе с программой и уже лежит в папке «На терминал». Файлы документов, которые нужно отправить на терминал, должны наименоваться определенным образом, а именно: сначала имя типа документа согласно конфигурации, а затем его номер в произвольном виде.

Настройки программы

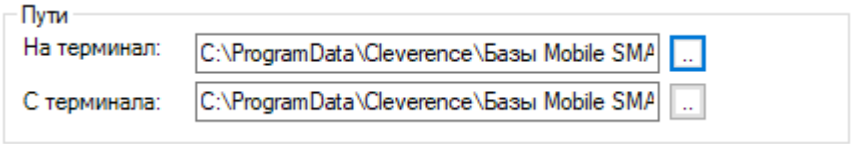
С помощью пункта меню «Файл -> Параметры обмена...» можно настроить используемый формат данных и переопределить пути к папкам «На терминал» и «С терминала».



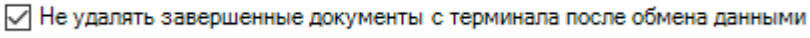
Универсальная программа поддерживает четыре формата данных: файлы Excel (*.xls;*.xlsx), файлы CSV с разделителем «запятая», и файлы CSV с разделителем «точка с запятой». Единовременно может быть выбран только один используемый формат. Файлы других форматов будут игнорироваться. Выбор осуществляется в разделе Формат файлов:



При использовании нестандартных путей для обмена данными, можно поменять адрес по умолчанию на нужный:



Если требуется не удалять завершенные документы с терминала после обмена данными, то необходимо поставить галочку напротив данного пункта:



За чтение или игнорирование заголовков отвечает следующий функционал:

☐ Не читать первую строку данных (первая строка - заголовок)

Для перезаписи существующей номенклатуры используется переключатель из списка:

☑ Перезаписывать существующую номенклатуру при выгрузке

☑ Генерировать индекс полнотекстового поиска для номенклатуры

☑ Перезаписывать существующие ячейки при выгрузке

☐ Перезаписывать существующих пользователей при выгрузке

☐ При выгрузке перемещать в Good/Bad

Не нашли что искали?

Задать вопрос в техническую поддержку

Запуск утилиты файлового обмена (Excel/CSV) из командной строки

Последние изменения: 2024-03-26

```
XlsCsvUtil.exe /id код_базы [full|upload|download] [noclose|close] [nocloseerr|closeerr] [notray|tray]
```

Посмотреть код базы Вы можете через менеджер баз Mobile SMARTS.

Параметр
Описание
full
Последовательное выполнение процесса выгрузки и загрузки
upload
Только выгрузка документов на ТСД
download
Только загрузка документов с ТСД
noclose
Не закрывать утилиту после выполнения
close
Закрыть утилиту после выполнения
nocloseerr
Не закрывать после ошибки
closeerr
Закрыть после ошибки
notray
Не сворачивать в трей
tray
Свернуть в трей

Примеры:

Во всех случаях будет произведено подключение к базе 7a05d8eb-a672-4151-9def-34534534d с последующим обменом данными.

В данном случае будет произведен полный обмен данными (выгрузка и загрузка с ТСД) без закрытия утилиты и сворачивание в трэй:

```
XlsCsvUtil.exe /id 7a05d8eb-a672-4151-9def-34534534d full noclose nocloseerr tray
```

интеграция, утилита, Excel, Csv

Не нашли что искали?

Задать вопрос в техническую поддержку

Автоматический файловый обмен (Excel/CSV) на сервере с помощью коннектора Excel/CSV

Последние изменения: 2024-03-26

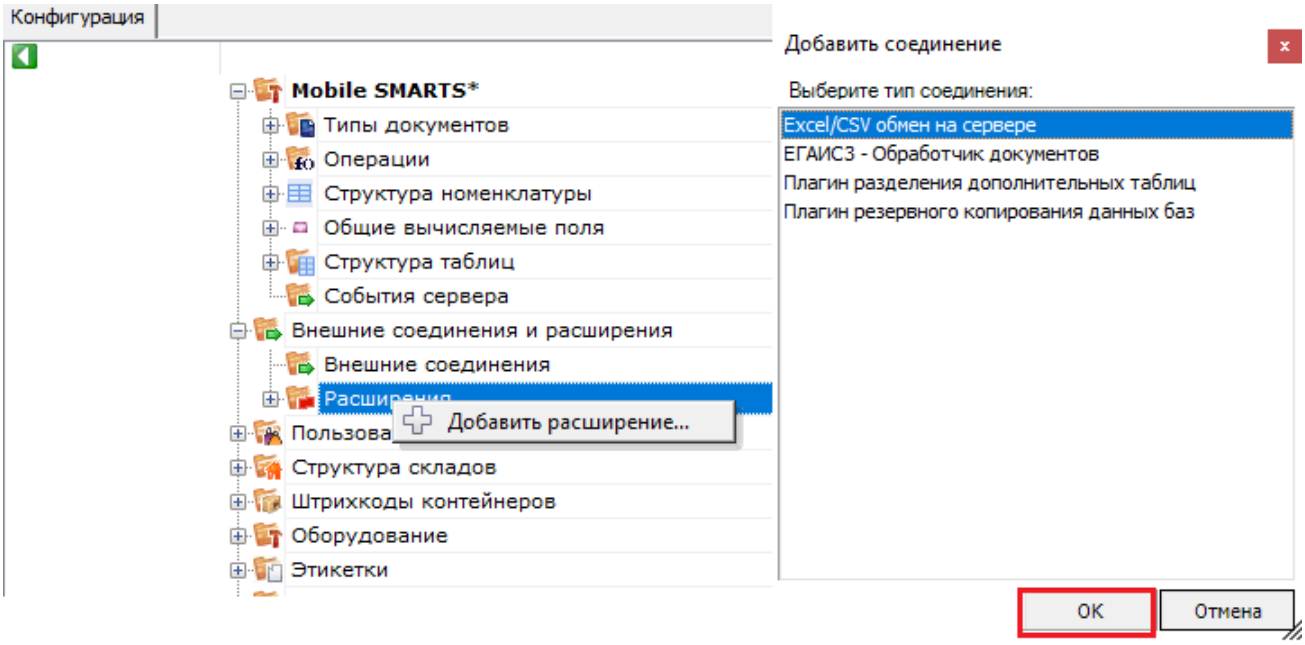
Кроме использования ручного обмена с помощью утилиты для конвертации текстовых файлов в данные Mobile SMARTS, платформа позволяет также настроить автоматический файловый обмен на сервере.

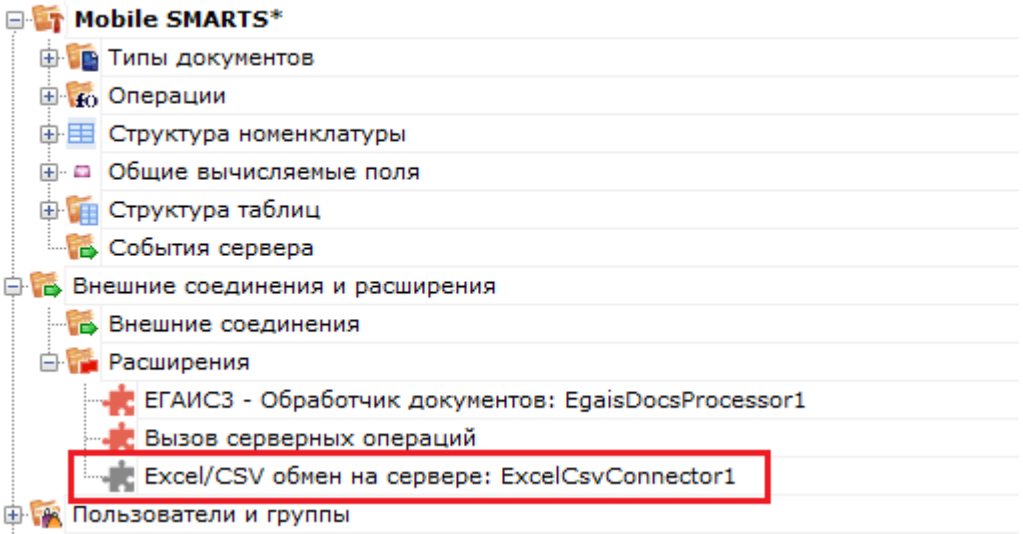
При таком варианте сервер будет следить за указанной папкой с файлами, и как только там будут происходить изменения (добавляться новые файлы или изменяться старые) их конвертация будет происходить автоматически. Точно также, как только документ будет завершаться пользователем, он будет сразу преобразован в итоговый файл Excel или CSV.

Работа такого автоматического файлового обмена осуществляется через Excel/CSV коннектор сервера.

Как настроить и включить:

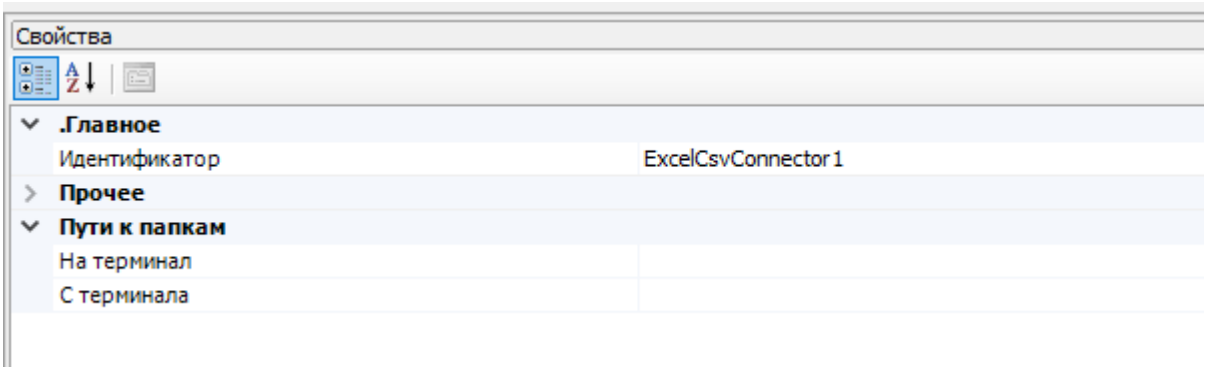
В расширениях создаем новое расширение «ExcelCsvConnector»:



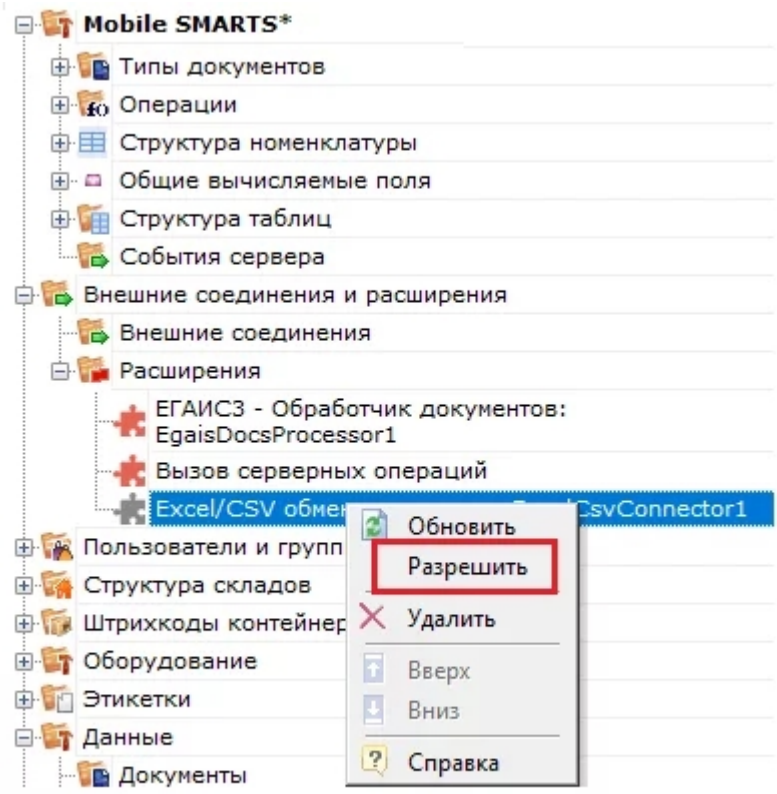


В качестве параметров коннектора используются общие настройки файлового обмена (Excel/CSV) в файле XlsCsvUtil.exe.config.

В настройках самого коннектора мы можем только поменять пути к папкам входящих и исходящих документов.



После создания и сохранения коннектора — необходимо нажать правой клавишей мыши на названии коннектора и в появившемся контекстном меню нажать «Разрешить». В дальнейшем, после перезапуска сервера, он будет включаться автоматически.



Сразу после запуска коннектор начинает слежение за папками и завершенными документами.

интеграция, Excel, CSV

Не нашли что искали?

Задать вопрос в техническую поддержку

Сравнение возможностей утилиты и коннектора для файлового обмена (Excel/CSV)

Последние изменения: 2024-03-26

И утилита, и коннектор обладают абсолютно идентичными возможностями по конвертации данных в/из Mobile SMARTS.

Основные различия в функционале утилиты и коннектора обусловлены только тем, что коннектор работает прямо внутри сервера. Благодаря этому он обеспечивает автоматическую конвертацию данных (как только данные появляются в папке и наоборот, как только завершенный документ попадает с мобильного устройства на сервер). Также коннектор работает быстрее чем утилита, так как все файлы обрабатываются прямо сервером и их не требуется передавать туда по http, как это делает утилита.

Из минусов - коннектор доступен только для серверного режима базы.

Сводная таблица различий:

Функция	Утилита	Коннектор
Конвертация всех типов данных (номенклатура, документы, таблицы, ячейки и т.д.)	Да	Да
Возможность работы без сервера	Да	Нет
Автоматический обмен сразу по появлению файлов в папке	Нет	Да
Скорость конвертации	Медленно	Быстро

интеграция, утилита, коннекторы

Не нашли что искали?

Задать вопрос в техническую поддержку

Структура файлов и папок для файлового обмена (Excel/CSV) с Mobile SMARTS

Последние изменения: 2024-03-26

Все файлы и папки, необходимые для работы файлового обмена с Mobile SMARTS расположены по пути [Папка базы]\XlsCsv.

Содержимое папки XlsCsv

Имя папки \ подпапки
Описание
На терминал
По-умолчанию содержит файлы Excel или CSV, предназначенные для отправки в Mobile SMARTS.
На терминал\Архив
Архив успешно конвертированных файлов Excel и CSV. Если файл «пропал», его можно найти здесь.
На терминал\Good
Архив успешно конвертированных файлов Excel и CSV при включенном режиме IsGoodBad.
На терминал\Bad
Архив неудачно обработанных файлов Excel и CSV при включенном режиме IsGoodBad.
С терминала
По-умолчанию сюда складываются файлы с терминала после конвертации их в Excel или CSV по шаблон.
Templates
Папка с файлами шаблонов для конвертации. Большинство готовых продуктов уже содержат какие-то шаблоны по-умолчанию.
Templates\Upload
Содержит шаблоны, по которым разбираются файлы,загружаемые в Mobile SMARTS.
Templates\Download
Содержит шаблоны, по которым формируются готовые файлы из Mobile SMARTS.с терминала.
XlsCsvUtil.exe.config
Файл конфигурирования параметров файлового обмена с Mobile SMARTS.

интеграция, Excel, CSV

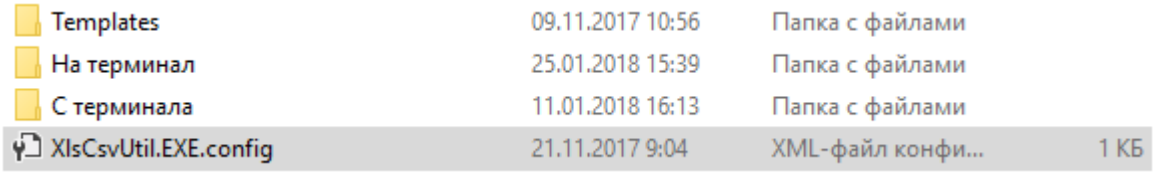
Не нашли что искали?

Задать вопрос в техническую поддержку

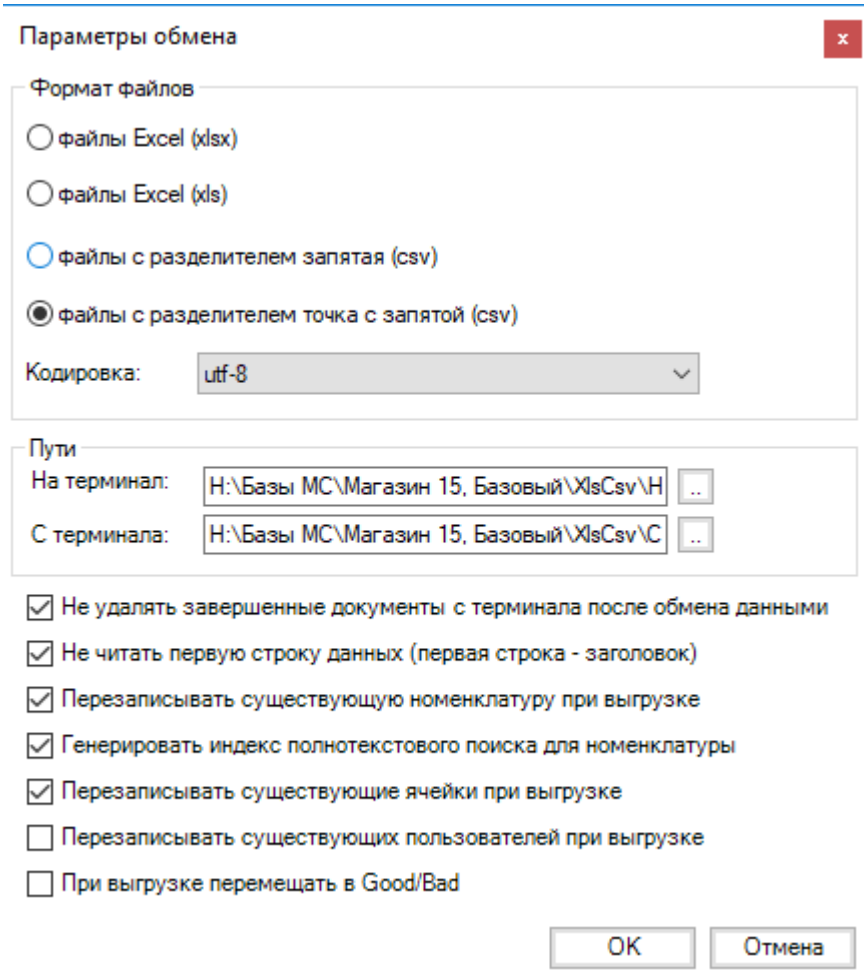
Настройка параметров файлового обмена (Excel/Csv) с Mobile SMARTS

Последние изменения: 2024-03-26

Основные настройки параметров обмена, такие как пути в папкам, формат, кодировка файлов и т. д. хранятся в файле по пути [Папка базы]\XlsCsv\XlsCsvUtil.exe.config.



Часть этих настроек можно также редактировать через окно в утилите обмена.



Полный список доступных настроек:

Указывает содержит ли файл с пользователями весь справочник пользователей, или дополняет уже выгруженный ранее.

- **true — справочник полный**
- **false — дополняет текущий**

Имя
Значения
Описание

exchangeformat
csv, csvcomma, xls, xlsx, xml
Используемый формат файлов
output_exchangeformat
csv, csvcomma, xls, xlsx, xml можно указать несколько через '|'
Опционально позволяет задать выходной формат файлов, или даже несколько. Пример: «csv|xlsx» — завершенные документы будут конвертироваться и в csv и в Excel формат.
encoding
utf-8|windows-1251
Кодировка, в которой сохранены данные и шаблоны конвертации. Параметр актуален только для форматов csv и xml.
cultureInfo
имя локализации https://msdn.microsoft.com/en-us/library/cc233982.aspx
Задает использование нестандартной локализации. Влияет на преобразование в текстовую форму различных типов данных. Например, с русской локализации дробная часть чисел выводится через запятую, а в английской через точку. По умолчанию: текущая локализация системы Пример: «ru-RU», «en-EN»
uploadFolder
путь к папке
Путь к папке с данными для загрузки в Mobile SMARTS.
downloadFolder
путь к папке
Путь к папке для итоговых файлов после работы на мобильном устройстве.
notDeleteCompletedDocuments
true или false
Флаг задает, надо ли удалять документ с сервера или мобильного устройства после успешной конвертации. true — файл не удаляется false — файл удаляется
notReadFirstString
true или false
Флаг задает, должна ли первая строка файла с данными восприниматься как заголовок, или данные в файле начинаются сразу, без заголовка. true — первая строка заголовок false — первая строка является данными

IsGoodBad
true или false
Задает режим, при работе которого исходный файл с данными после конвертации перемещается в «Good» или папку «Bad».
overwriteProducts
true или false
Указывает содержит ли файл с товарами весь справочник товаров, или дополняет уже выгруженный ранее. true — справочник полный false — дополняет текущий
overwriteCells
true или false
Указывает содержит ли файл с ячейками весь справочник товаров, или дополняет уже выгруженный ранее. true — справочник полный false — дополняет текущий
overwriteUsers
true или false
createProductsNameLookup
true или false
Указывает строить ли для справочника товаров индекс для текстового поиска по названию товара.
writeOutputSectionHeaders
true или false
только для csv Указывает, выводить ли в выходные файлы имена секций данных или нет.
writeOutputSectionFieldNames
true или false
только для csv Указывает, выводить ли в выходные файлы заголовки колонок с данными или нет.
productsFileName
маска имени файла
Позволяет задать имя исходного файла с номенглатурой для конвертации в Mobile SMARTS. Поддерживается поиск по маске По умолчанию: Номенклатура.*, Products.*
ИмяТаблицыFileName
маска имени файла
Позволяет задать имя исходного файла с данными таблицы для конвертации в Mobile SMARTS. Поддерживается поиск по маске По умолчанию: ИмяТаблицы.*
documentsFileName
маска имени файла
Позволяет задать общую маску файлов документов для конвертации в Mobile SMARTS. По умолчанию: ИмяТипаДокумента*.*

DocAlias_ИмяТипа
маска имени файла
Позволяет задать различные маски имен файлов для разных типов документов для конвертации в Mobile SMARTS.

Пример файла настроек:

```
<?xmlversion="1.0"?>
<configuration>
<appSettings>
<add key="overwriteUsers" value="True" />
<add key="notReadFirstString" value="True" />
<add key="exchangeformat" value="csv" />
<add key="IsGoodBad" value="True" />
<add key="encoding" value="utf-8" />
<add key="notDeleteCompletedDocuments" value="True" />
</appSettings>
</configuration>
```

интеграция, Excel, CSV

Не нашли что искали?

Задать вопрос в техническую поддержку

Настройка именования файлов для загрузки в Mobile SMARTS с помощью файлового обмена (Excel/CSV)

Последние изменения: 2024-03-26

Ниже приведена таблица имен файлов по-умолчанию для загрузки в Mobile SMARTS:

Тип данных
Возможные имена файлов
Справочник номенклатуры
Номенклатура.xxx Товары.xxx Products.xxx
Справочник мест хранения
Места хранения.xxx Ячейки.xxx Cells.xxx
Справочник палет
Контейнеры.xxx Паллеты.xxx Палеты.xxx Pallets.xxx
Справочник пользователей
Пользователи.xxx Users.xxx
Дополнительные таблицы
ИмяТаблицы.xxx Примеры: Цены.csv Остатки.xlsx
Документы
ИмяТипа*.xxx Примеры: Приемка №3658543.xlsx Инвентаризация от 23.06.2017.csv

Если необходимо задать какие-то нестандартные имена, то это можно сделать с помощью файла настройки параметров.

Примеры

Задача:

Файл товаров имеет нетиповое и нефиксированное имя, и кроме фиксированной части содержит в себе дату, например “products 10.04.2018.gox”. Как настроить файловый обмен на работу с таким файлом?

Решение:

Зададим в XlsCsvUtil.exe.config параметр нестандартного имени файла номенклатуры:

```
<add key="productsFileName" value="products*.gox" />
```

Задача:

Мы пользуемся обменом csv файлами, но исходные файлы документов имеют расширение “ofp”, вместо “csv”

Решение:

Зададим в XlsCsvUtil.exe.config параметр нестандартного имени файлов для всех файлов документов:

```
<add key="documentsFileName" value="*.ofp" />
```

Задача:

Файлы документов приемки вместо “Приемка*.csv” называются у нас “income*.csv”.

Решение:

Зададим в XlsCsvUtil.exe.config параметр нестандартного имени файлов для документов приемки:

```
<add key="docAlias_Приемка" value="income*.csv" />
```

интеграция, Excel, CSV

Не нашли что искали?

Задать вопрос в техническую поддержку

Правила распределения документов пользователям при выгрузке из файла

Последние изменения: 2024-03-26

Если при выгрузке документа в Mobile SMARTS у него оказывается незаполненным поле Appointment, то тогда применяются описанные ниже стандартные правила.

База с прямым обменом

Документ выгружается первому пользователю с правами «Мобильный пользователь» или «Администратор». Выставляется поле UserId.

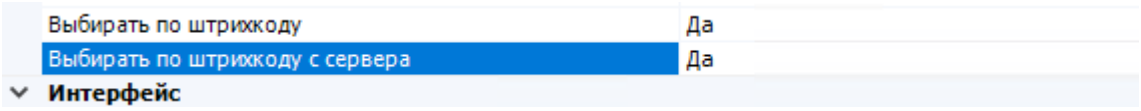
База на сервере

Правила проверяются в указанном порядке. Если какое-то из них срабатывает, то оно выполняется, и остальные правила уже не проверяются!

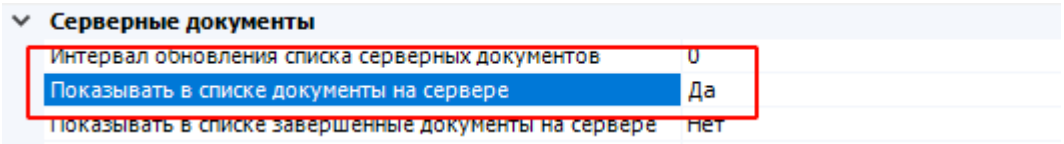
1. Если документ не распределяется автоматически (AutoAppointed == false) или он коллективный (ServerHosted == true), то документ не изменяется и выгружается на сервер в исходном виде.

2. Если пользователь с правами «Мобильный пользователь» или «Администратор» всего один, то документ назначается ему напрямую (Appointment = user.Id).

3. Если у документа заполнено поле штрихкода и у типа документа указано, что документы можно выбирать по штрихкоду и по штрихкоду с сервера, то документ переводится в режим не выдаваемых автоматически (AutoAppointed = false). В списке документов на ТСД он не отображается, но пользователь сможет выбрать его, отсканировав штрихкод документа.



4. Если у типа документа разрешено показывать в списке документы с сервера, то документ переводится в режим не выдаваемых автоматически (AutoAppointed = false). Пользователь сможет выбрать его из списка.



5. Если ни одно из указанных правил не сработало, то никаких изменений не производится. Документ будет распределяться по общим правилам платформы.

Не нашли что искали?

Задать вопрос в техническую поддержку

Как при обмене Excel/csv выгрузить свое поле из файла в строку документа Mobile SMARTS и загрузить это поле обратно

Последние изменения: 2024-03-26

Обмен документами в системе Mobile SMARTS осуществляется с помощью утилиты «Excel Csv обмен», которая входит в пакет установки (см. также Расположение файлов и их назначение).

Как подготовить документы для обмена Excel/CSV файлами

Вариант 1. Подготовить документы и загружать/выгружать не меняя «коробочных решений». Данное решение предполагает заполнение в документах только актуальные поля, не внося изменения в неиспользуемые. Документы содержат множество полей, некоторые из них возможно не будут использоваться при работе.

Вариант 2. Через удаление ненужных полей в (за)выгружаемом документе. Второй вариант предполагает изменение шаблонов документов, используемых платформой Mobile SMARTS. В данном случае возможно получить формы документов, удобные вам — убрать неиспользуемые колонки, поменять их местами, подстроить шаблоны файлов под учетную систему.

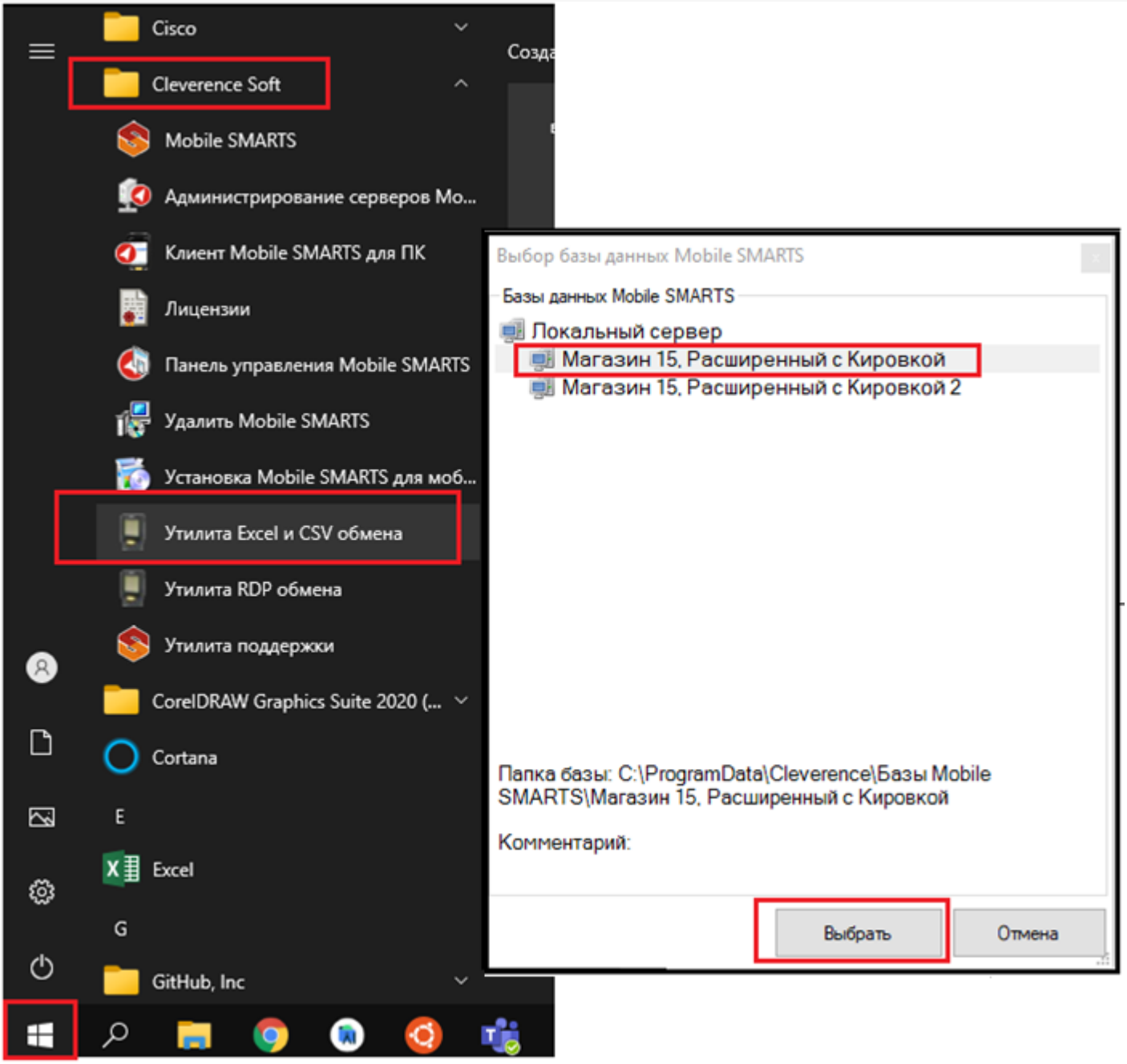
Вариант 3. Через изменения конфигурации Mobile SMARTS, этот способ позволяет внести необходимые изменения в конфигурацию, добавить поля, задать их обработку, изменение или вывод по вашим требованиям, используя как готовые процедуры так и создавая полностью свои нетиповые бизнес-процессы. Также можно установить пустую базу для создания всех процедур и алгоритмов с чистого листа, используя например добавление поля в номенклатуру.

При установке многих продуктов, с обменом через Excel/Csv, копируются «Демо-файлы», во всех форматах (xls, xlsx, csv). В данной статье будем использовать файлы *.xlsx. Получить доступ к папке можно используя входящую в состав платформы Mobile SMARTS утилиту «Excel/Csv обмен».

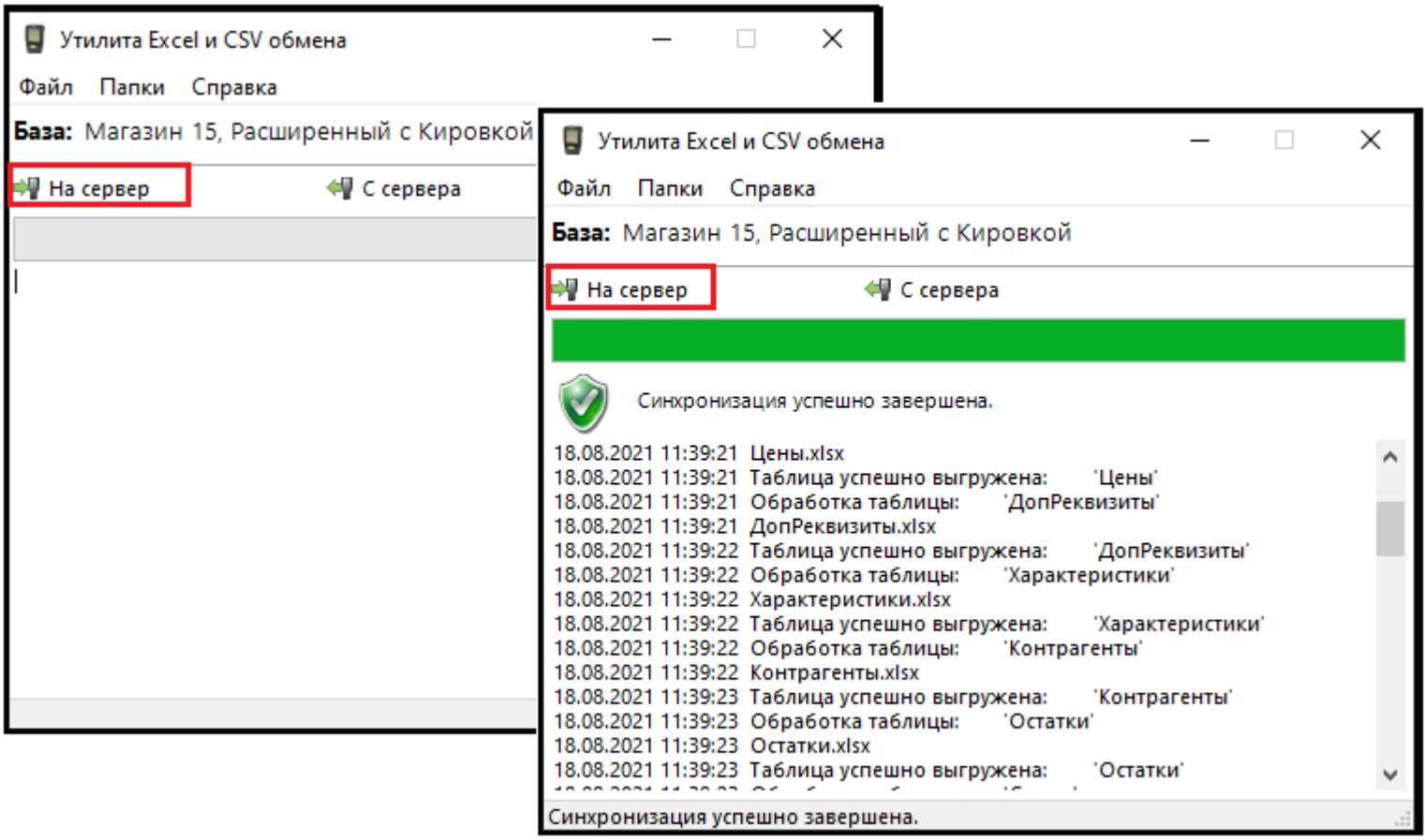
Запуск утилиты Excel/CSV обмена для ОС Windows

Для запуска утилиты нажмите в нижнем левом углу рабочего стола на кнопку «Пуск», откройте главное пользовательское меню операционной системы Windows.

Выберите папку «Cleverence Soft», нажмите на «Утилита Excel и CSV обмена», далее выберите нужную базу данных приложения Mobile SMARTS (рассмотрим на примере «Магазин 15: Расширенный с Кировкой»), нажмите кнопку «Выбрать».



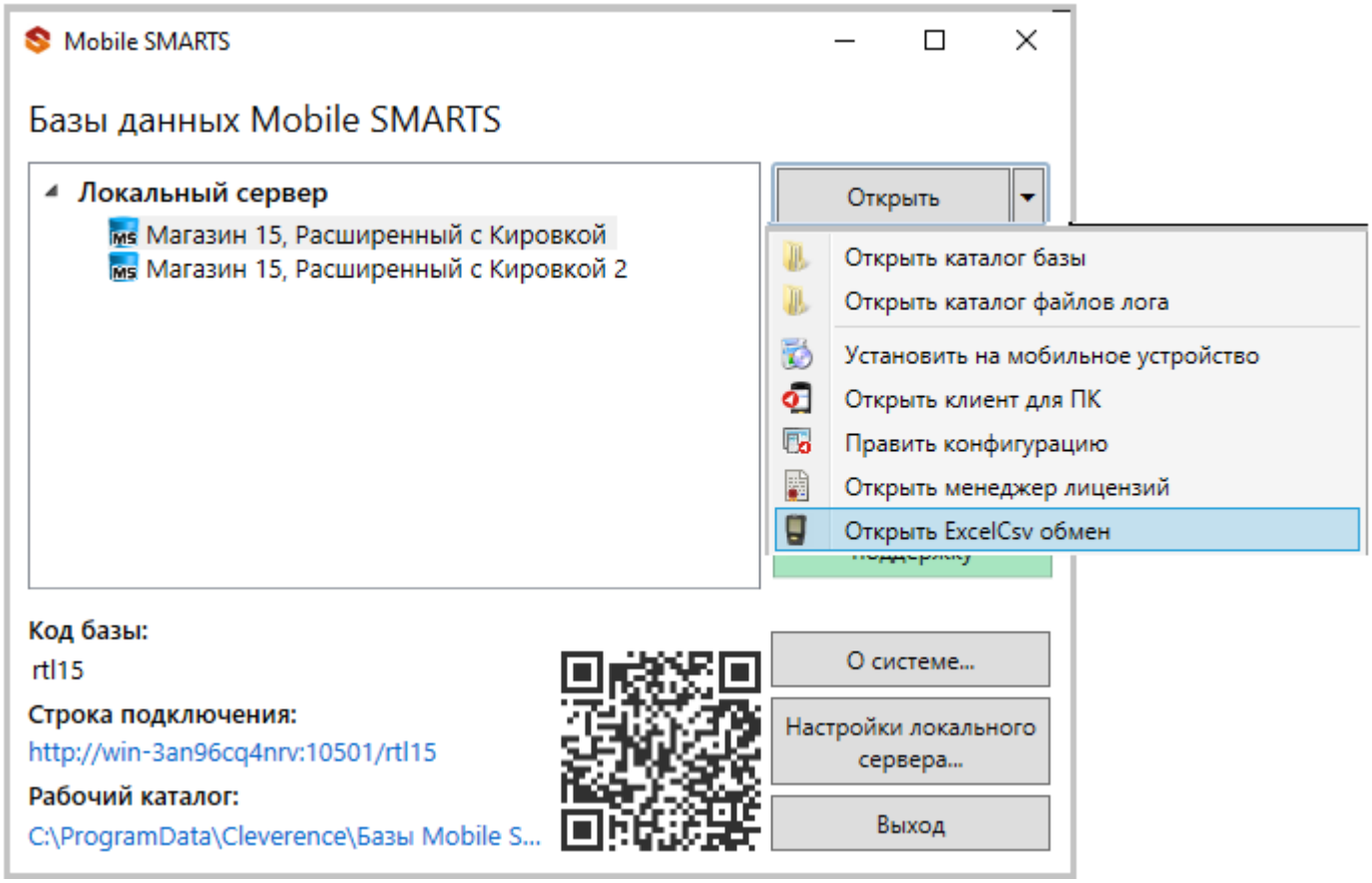
Откроется окно утилиты. Для отправки документов на терминал нажмите кнопку «На сервер», для того чтобы забрать файлы с терминала нажмите на кнопку «С сервера» (при работе базы в режиме прямого подключения к устройству (без сервера) кнопки будут иметь вид «На терминал», «С терминала»).



Утилита Excel/Csv

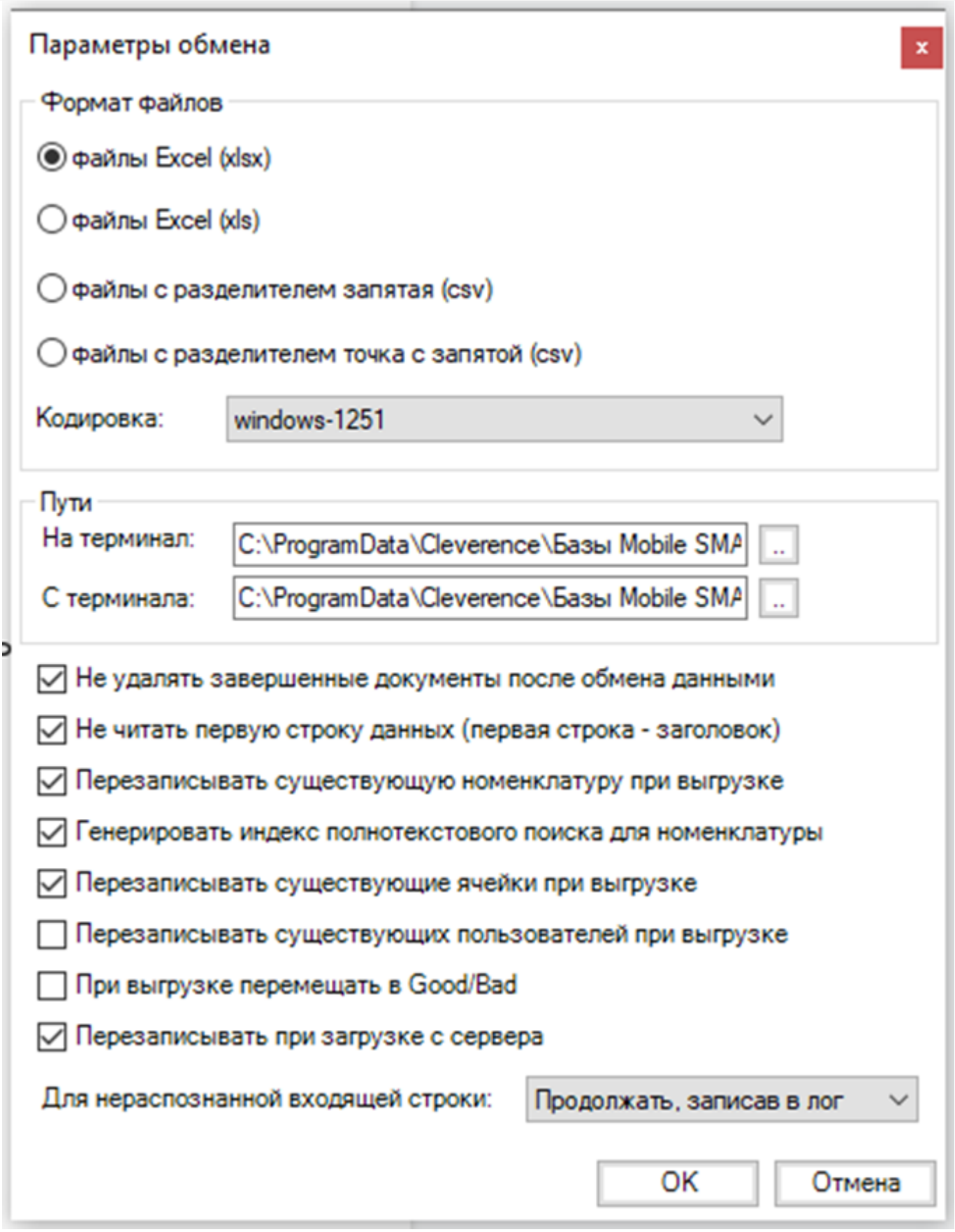
Для обмена документами с ТСД запустите на рабочем столе менеджер баз Mobile SMARTS,

далее выберите нужную базу из списка, например «Склад 15», справа от кнопки «Открыть» нажмите на раскрывающийся список, запустите «ExcelCsv обмен», запускается утилита.



Обратите внимание. В меню «Файл->Параметры обмена» нужно выбрать **формат обмена xls/xlsx/csv**, здесь также можно настроить пути к папкам обмена.

Для этого зайдите в меню «Файл», выберите окно «Параметры обмена», настройте используемый формат данных и переопределите пути к папкам «На терминал» и «С терминала». В пункте «Формат файлов» поставьте галочку на нужном параметре.



Для загрузки документа в базу данных Mobile SMARTS и далее в ТСД файл «Приход» необходимо поместить в папку «На терминал»: «C:\ProgramData\Cleverence\Базы Mobile SMARTS\Имя Базы\XlsCsv\На терминал»

Попасть в эту папку можно непосредственно из утилиты выбрав пункт меню «Папки->На терминал». В этой папке находятся все документы предназначенные к загрузке в систему и на ТСД.

После нажатия кнопки «На сервер» утилита проверяет файл на возможные ошибки и в случае успеха отправляет на сервер, далее документ доступен на ТСД.

Файл из папки «На терминал» переносится в папку «Архив», файл номенклатуры и файлы таблиц остаются

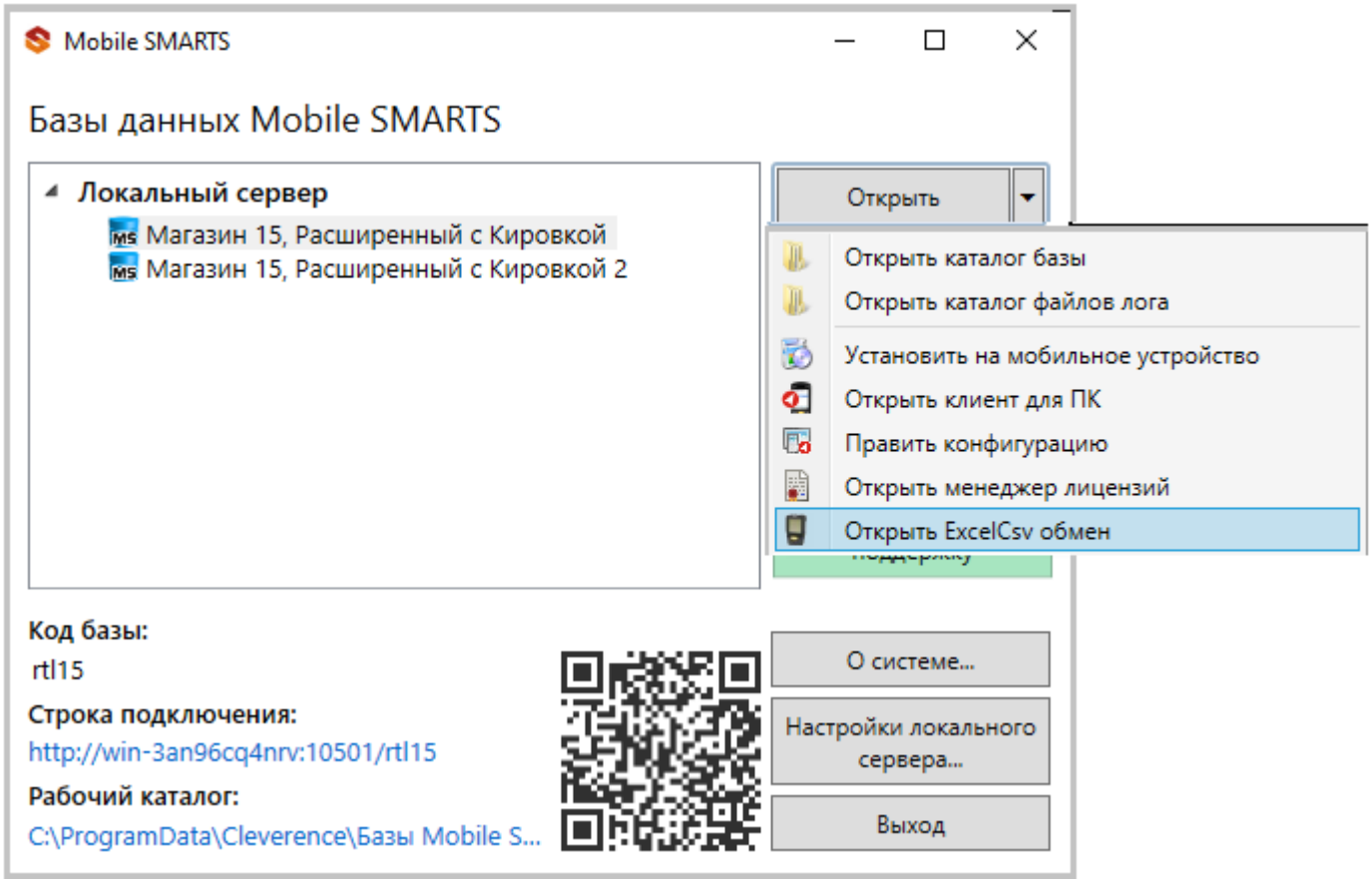
в текущей папке.

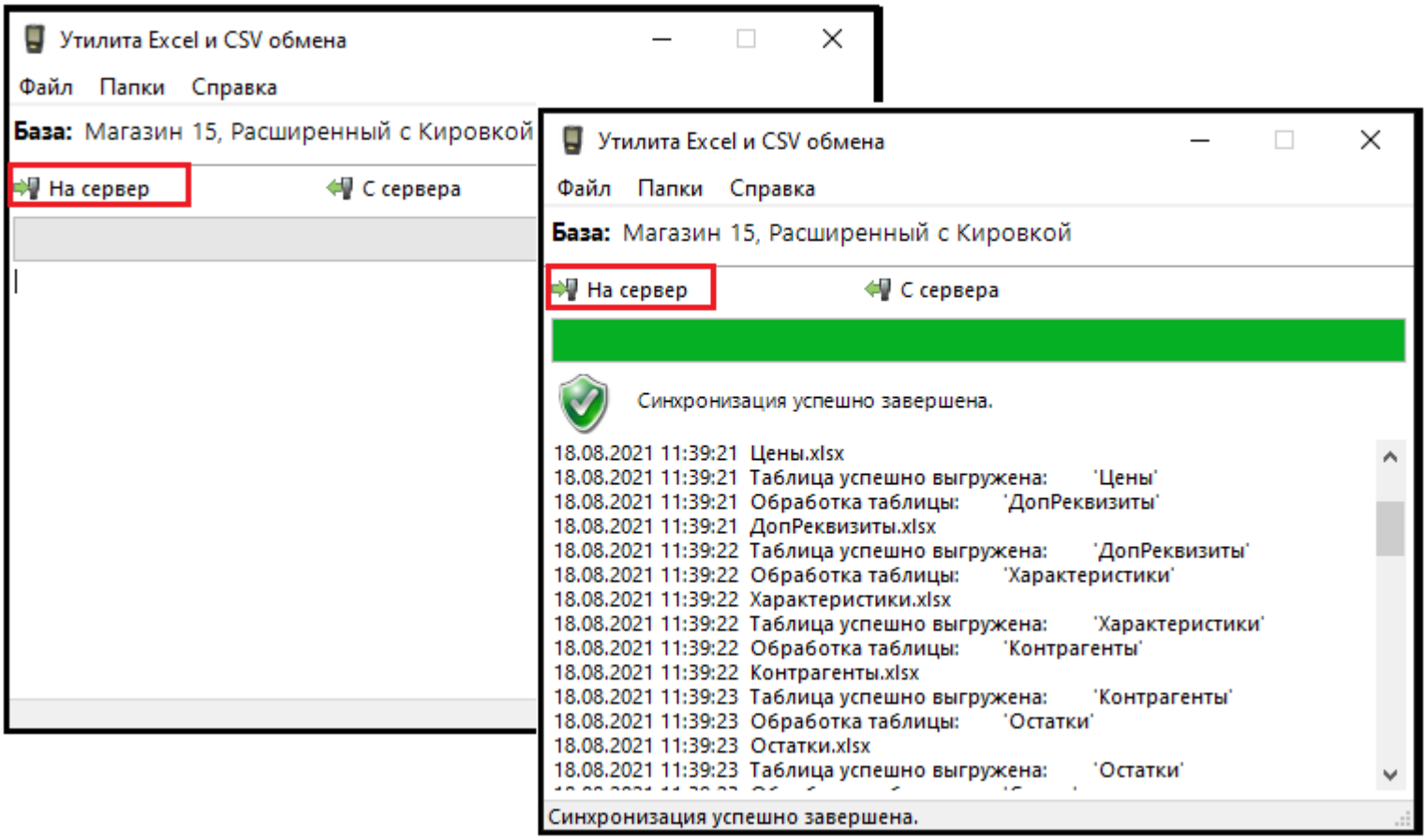
Выгрузка документов из Mobile SMARTS как и модификация шаблонов практически не отличается от загрузки.

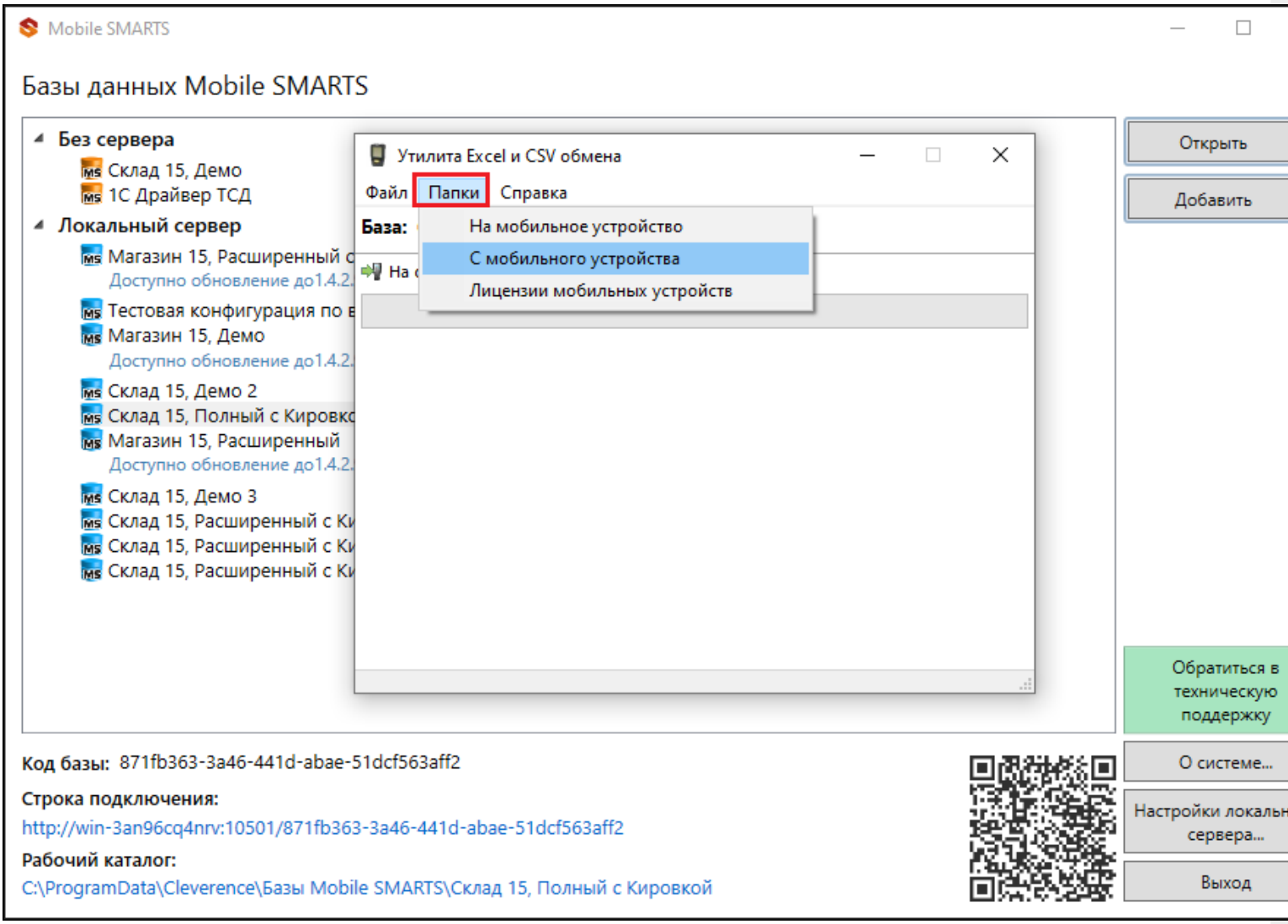
Шаблон выгрузки «Документ» находится по пути «C:\ProgramData\Cleverence\Базы Mobile SMARTS\Имя Базы\XlsCsv\Templates\Download»

Запуск утилиты Excel/CSV обмена через конфигуратор

Выберите нужную базу данных из списка, с которой будет работать утилита обмена, затем справа от кнопки «Открыть» нажмите на раскрывающийся список, выберите «Открыть ExcelCsv обмен».



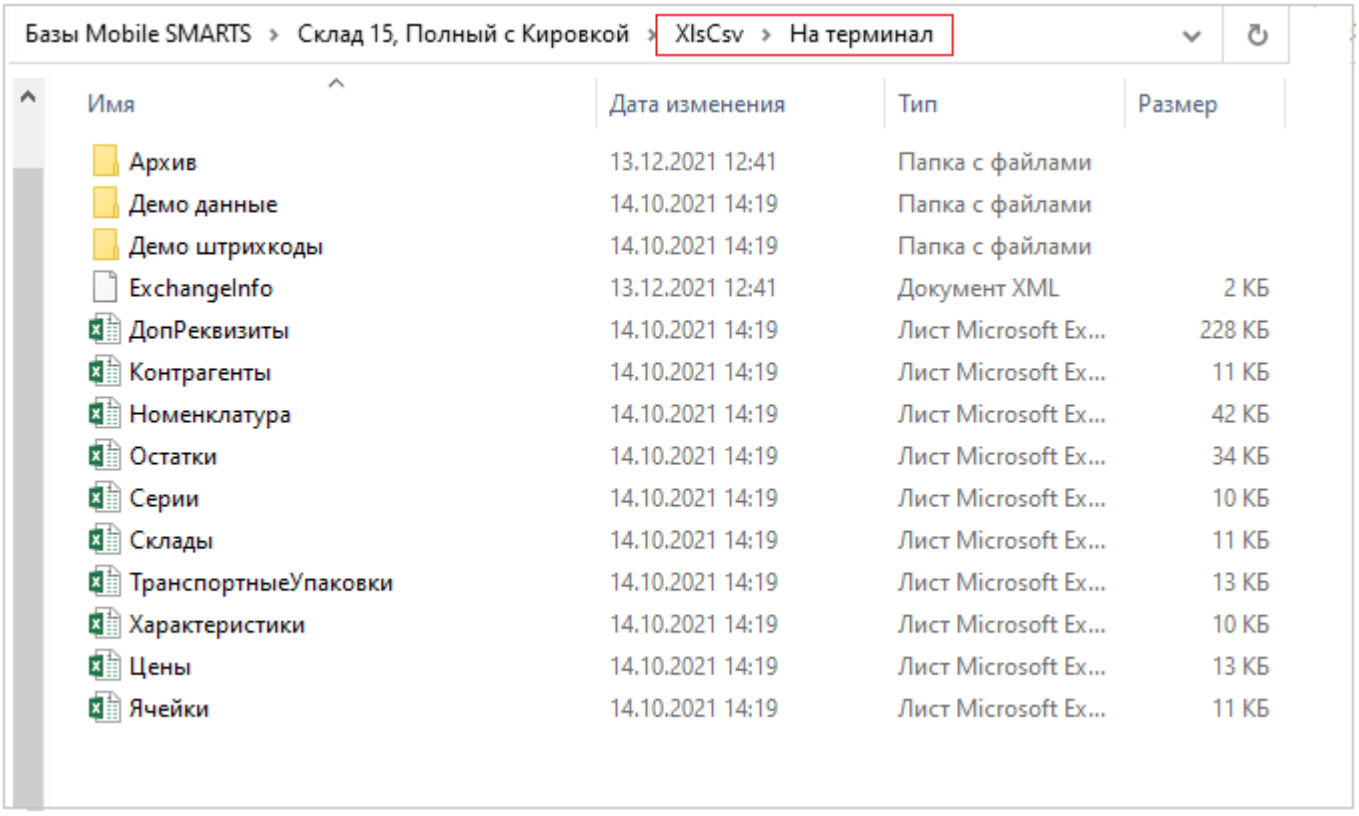




Нажмите в меню утилиты на кнопку «Папки», далее выберите соответствующий вариант загрузки:

«На мобильное устройство» или «С мобильного устройства».

В выбранной базе автоматически создается папка «XlsCsv» с подпапками «На терминал» и «С терминала» (согласно способам загрузки «На мобильное устройство» и «С мобильного устройства»).



После нажатия кнопки «На сервер» (где документы выгружаются/передаются в базу данных Mobile SMARTS, на ТСД документы загружаются при обмене сервера с терминалом) большая часть загружается в систему в папку «Архив» с добавлением даты и времени документа.

В текущей папке остаются файлы справочника номенклатуры и файлы таблиц, а также файлы документов, которые были пропущены из-за настроек выгрузки (например, настроен обмен в формате .xlsx, а загружаемые файлы в формате .csv) и файлы документов, которые не были выгружены из-за системных ошибок.

Откройте файл «Номенклатура.xlsx», первое, что бросается в глаза — поля заполненных числами типа

«253a91e0-269e-4ff1-bb36-b33e878c69ab».

Это GUID (Globally Unique Identifier), который формируется в учетной системе или генерируется производителем.

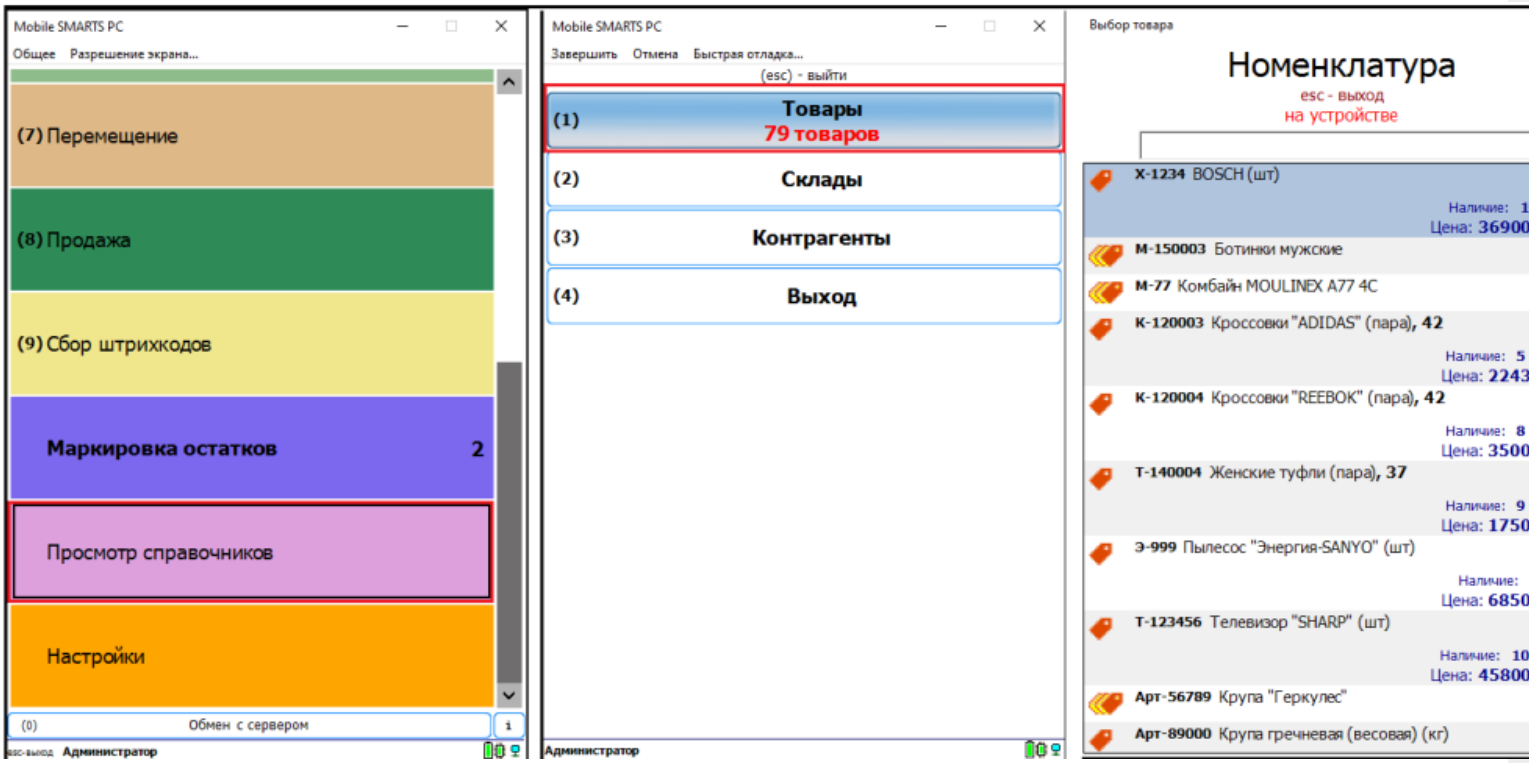
Добавьте свою строку с товаром. Заполните только те поля, которые известны. Пример «Лапти****»

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	ProductId (ид т	Marking (Арт	Наименование товара	Штрихкод товар	Базовая упаковки		Имя Упако	PackingId (и	ИдЕд	Кол.	По се	Ключ	Рабо	По ха	Ключ	Имя	ИдХаракт	Остаток	Цена
151	1b88787c-c4d8-	B-11122	Бутилированная вода «Легенда гор Ар	01000	d58239ca-	5007206	бут	d58239ca-6	d582	1	1	f886db9f-5f88-4d0a-92a8-1e313f60c574						27	495
152	253a91e0-269e-	B-12211	Бутилированная вода «Сделай Мир Дo	01010	8f2bb1ea-	1014187	бут	8f2bb1ea-e	8f2bb	1	1	f886db9f-5f88-4d0a-92a8-1e313f60c574						21	385
153	11111110	Л-77	Лапти московские	01011			пара	456										100	500
154	11111111	Л-78	Лапти ленинградские	01012			пара	456										200	600
155																			

При заполнении важно заполнить поля «A» и «H» (ProductId и PackingId), значение может быть любым, сформировано вручную или вашей учетной системой.

Далее сохраните файл и закройте программу в которой он был создан. Откройте утилиту обмена и нажмите кнопку «На сервер», утилита проверяет правильность файла и в случае успеха, загружает его в платформу Mobile SMARTS.

Проверить загрузилось или нет можно в панели управления платформой вкладка «Данные-Номенклатура», либо запустив «Клиент для ПК» выбрав меню «Просмотр справочников».



> **Обратите внимание.** Несмотря на то, что в файле «Номенклатура» более 150 строк, приложение сообщает, что товаров 79. Это происходит из-за того, что один и тот же товар идет в разной упаковке, например молоко 5 пакетов и одна коробка с 6-ю пакетами. Строк в номенклатуре 2, а позиция одна.

Нажмите на кнопку «Товары» — появляется список товаров. Поиск можно осуществить по «Артикулу», «Наименованию», «Штрихкоду», обратите внимание, что вводить необходимо не менее 3-х символов. В случае совпадения поискового запроса на экран выводятся все позиции имеющие совпадение, далее можете выбрать необходимую.

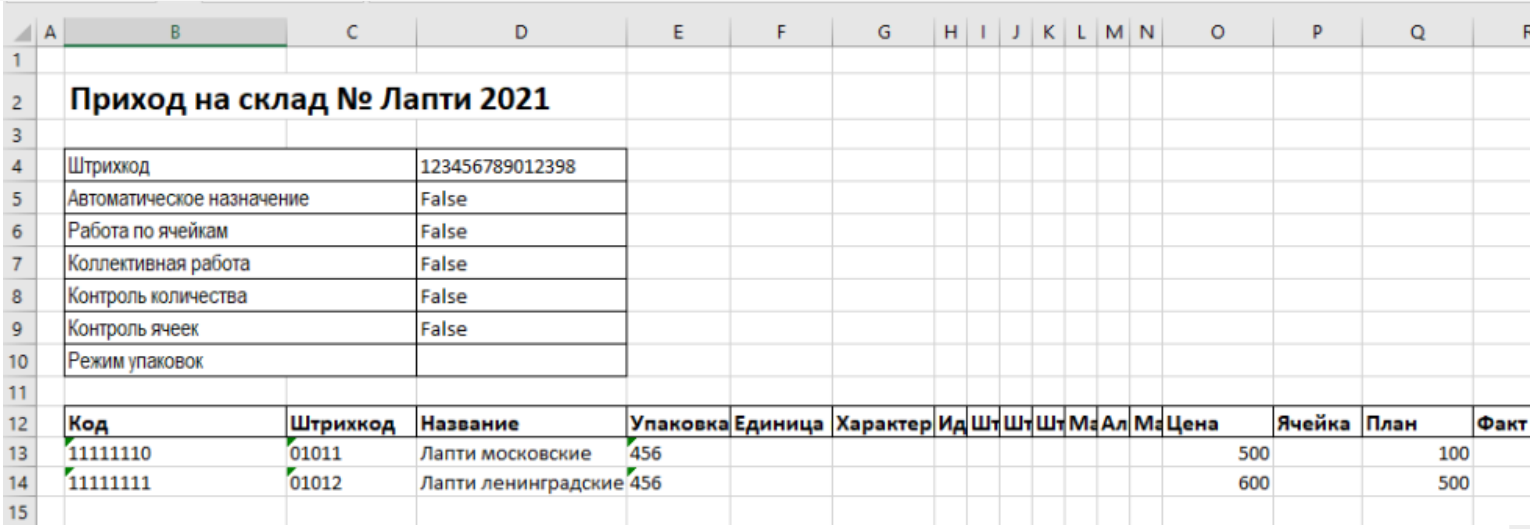
Рассмотрим на примере базы данных «Склад 15: Расширенный с Кировкой» операцию «**Оприходование товара на складе**».

Для этого учетная система должна сформировать файл «Приход на склад *******», здесь вместо ***** — впишите текст, номер накладной, поставщика или все вместе.

Можно создать его самостоятельно на основе демонстрационного файла «Приход на склад демо» согласно полного пути к папке: «C:\ProgramData\Cleverence\Базы Mobile SMARTS\Имя Вашей базы\XlsCsv\На терминал\Демо данные\xlsx»

Откройте его и сохраните в папку: «C:\ProgramData\Cleverence\Базы Mobile SMARTS\Имя Вашей базы\XlsCsv\На терминал»

Имя файла в папке на загрузку («На терминал») должно соответствовать желаемой операции. В данном случае «Приход на склад». Заполните демо-файл данными и удалим все строки с демо-данными.



Приход на склад № Лапти 2021

Штрихкод	123456789012398
Автоматическое назначение	False
Работа по ячейкам	False
Коллективная работа	False
Контроль количества	False
Контроль ячеек	False
Режим упаковок	

Код	Штрихкод	Название	Упаковка	Единица	Характер	Ид	Шт	Шт	Шт	Ма	Ал	Ма	Цена	Ячейка	План	Факт
11111110	01011	Лапти московские	456										500		100	
11111111	01012	Лапти ленинградские	456										600		500	

Здесь следует обратить внимание на изменение ячейки «B2». Ячейка обязательно должна содержать «Приход на склад №», после символа № может быть текст, который будет отображен в названии документа на терминале сбора данных. В данном случае «Лапти 2021». Ячейки ниже заполняются в соответствии с используемыми вами операциями или бизнес-процессами.

Загрузка файла в систему производиться как и в предыдущем случае нажатием на кнопку «На терминал» (при работе базы в режиме прямого подключения к устройству (без сервера)).

> **Обратите внимание.** Если база данных Mobile SMARTS работает в режиме «Подключение к серверу» (см. Настройка базы данных Mobile SMARTS), то загрузка файла производится путем нажатия кнопки «На сервер».

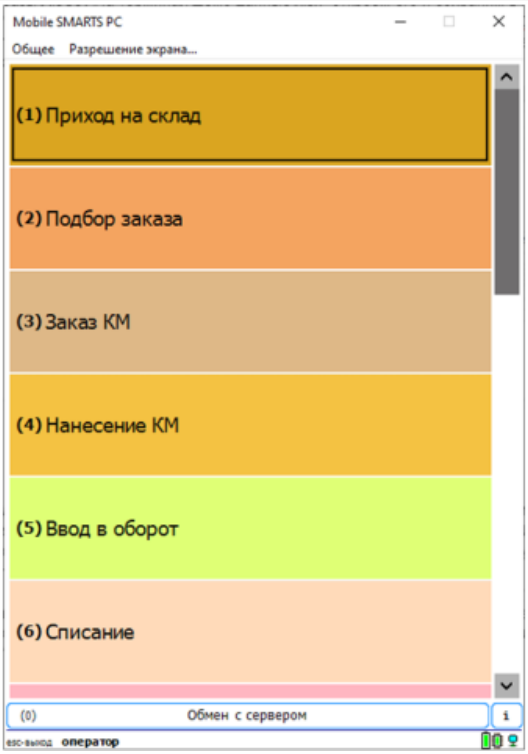
Утилита «ExcelCsv обмен» проверяет файлы и отправляет на сервер и терминал соответственно, после этого на ТСД или на десктопном приложении, загрузится документ.

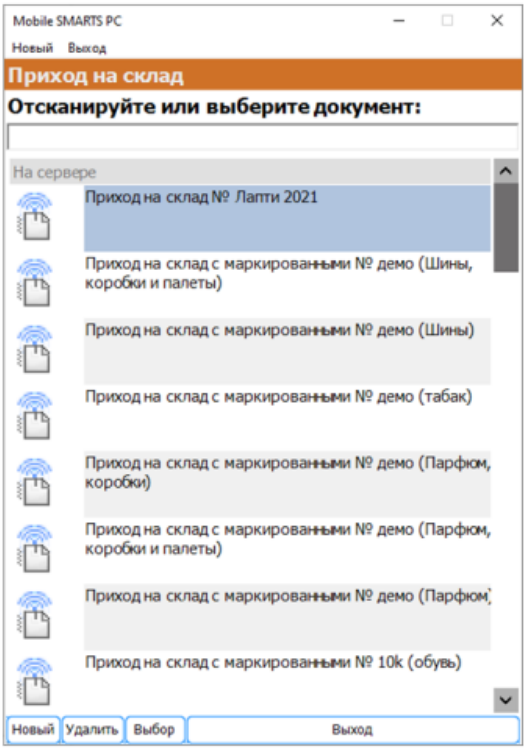
Нажмите «Приход на склад», далее «Приход Лапти 2021» и можете приступить к сканированию. Здесь можно использовать только штрихкоды, которые ввели в систему при добавлении нового товара в «Номенклатуру», где

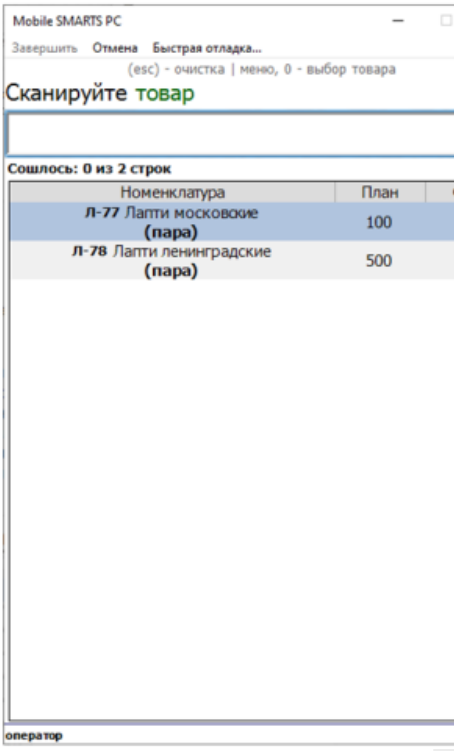
11111110, 11111111 — это код товара,

01011, 01012- это штрихкод.

Поиск в Mobile SMARTS может производится по любому из этих кодов.







Вставить номер штрихкода в окно сканирования можно комбинацией клавиш Ctrl+V. После сканирования, на складе пересчитывайте товар и введите количество на терминале (либо можно сканировать все позиции по одной). Далее нажмите «Esc» два раза и выберите «Завершить документ».

После этого документ с фактически подсчитанным количеством товара попадает на сервер и далее его можно выгрузить обратно, утилитой обмена, нажав кнопку «С терминала» или «С сервера» (см.выше).

В результате в папке «С терминала»

«C:\ProgramData\Cleverence\Базы Mobile SMARTS\Имя базы\XlsCsv\С терминала»

появляется файл «Приход на склад № Лапти 2021». Из этого файла можно увидеть, что фактически принятое количество соответствует заявленному, номер терминала и оператора, если эта функция включена в опциях. Так же в этом файле, выгрузились все поля используемые системой.

Далее можете обработать этот документ вручную или настроить его обработку в учетной системе.

Удаление ненужных полей в (за)выгружаемом документе

	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AB	AC	AD	AE	AF	AG	AH
1																																	
2			Приход на склад № Лапти 2021																														
3																																	
4																																	
5	Выполнил: оператор																																
6	Код ТСД: WIN-D-5E4B6525E66F32807DA602D8F85544EB																																
7	IP ТСД: fe80__61f7_5e36_138c_a5a0_6																																
8																																	
9																																	
10	№ стр	Штрихкод	Артикул	Название	Им	Ид	Ид	Ид	ерика	Заявлено	Принято	а прие	Наз	бъёе	епос	Про	ИНН	КПП	Алк	PDF	Сер	Сер	Дат	Нал	Кодка И	Мар	Штр	Штр	Яче	Блс	Мар	М
11	0	11111110	Л-77	Лапти московские						100,00	100,00	0,00		0	0								-1	###								
12	0	11111111	Л-78	Лапти ленинградские						500,00	500,00	0,00		0	0								-1	###								
13								Сумма:		600,00	600,00																					
14																																	
15																																	

Это вариант позволяет получить более простые документы, избавившись от неиспользуемых полей. Например файл для загрузки на ТСД/Сервер может выглядеть так

Приход на склад № Лапти 2021

Штрихкод	123456789012398
Автоматическое назначение	False
Работа по ячейкам	False
Коллективная работа	False
Контроль количества	False
Контроль ячеек	False
Режим упаковок	

Код	Штрихкод	Название	Упаковка	Цена	План	Факт
11111110	01011	Лапти московские	456	500	100	
11111111	01012	Лапти ленинградские	456	600	500	

Файл после выгрузки выглядит так:

Также для загрузки в систему из приведенного на скриншоте документа можно удалить колонки «Штрихкод», «Наименование», «Цена» и «Факт» и оставить только «Код», «Упаковка» и «План». С учетом того что все эти данные уже есть в номенклатуре, то платформа обработает все корректно.

Приход на склад № Лапти 2021

Выполнил: оператор
Код ТСД: WIN-D-5E4B6525E66F32807DA602D8F85544EB
IP ТСД: fe80__61f7_5e36_138c_a5a0_6

№ строки	Штрихкод	Артикул	Название	Заявлено	Принято
0	01011	Л-77	Лапти московские	100,00	100,00
0	01012	Л-78	Лапти ленинградск	500,00	500,00
				600,00	600,00

Для этого соответственно нужно также изменить файлы шаблонов на загрузку/выгрузку.

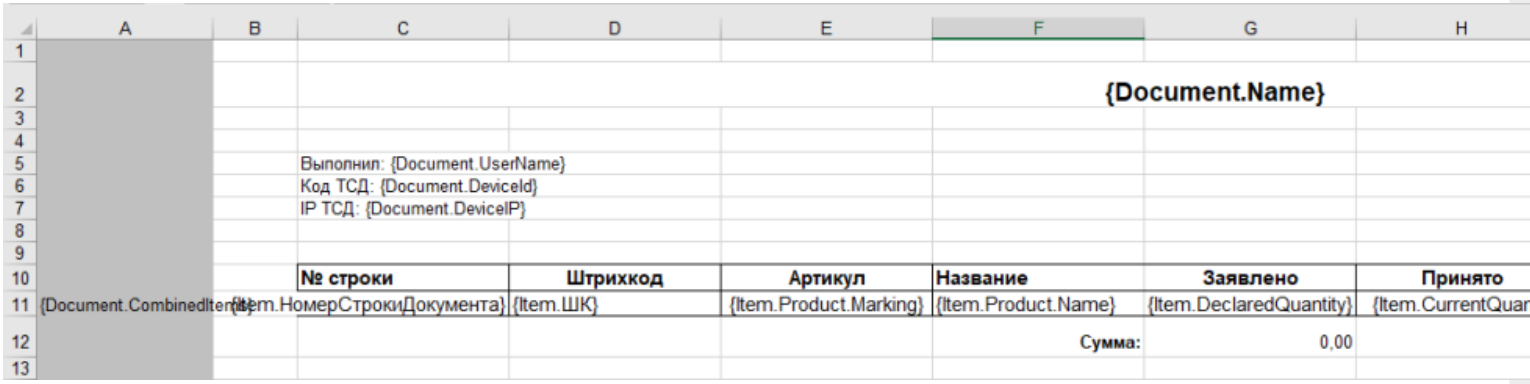
Измененный загружаемый шаблон: «C:\ProgramData\Cleverence\Базы Mobile SMARTS\Имя Базы\XlsCsv\Templates\Upload\Документ.xlsx»

{Document.Name}

Штрихкод	{Document.Barcode}
Автоматическое назначение	{Document.ВыдаватьАвтоматически}
Работа по ячейкам	{Document.ПоЯчейкам}
Коллективная работа	{Document.ИсполняемыйНаСервере}
Контроль количества	{Document.КонтрольКолва}
Контроль ячеек	{Document.КонтрольЯчеек}
Режим упаковок	{Document.РежимУпаковок}

Код	Штрихкод	Название	Упаковка	Цена	План	Факт
{Document.De {Item.ProductId}	{Item.ProductBarcode}	{Item.ProductName}	{Item.PackingId}	{Item.Цена}	{Item.DeclaredQuantity	{Item.CurrentQuantity}

Измененный выгружаемый шаблон: «C:\ProgramData\Cleverence\Базы Mobile SMARTS\Имя Базы\XlsCsv\Templates\Download\Документ.xlsx»



	A	B	C	D	E	F	G	H
1								
2							{Document.Name}	
3								
4								
5			Выполнил: {Document.UserName}					
6			Код ТСД: {Document.DeviceId}					
7			IP ТСД: {Document.DeviceIP}					
8								
9								
10			№ строки	Штрихкод	Артикул	Название	Заявлено	Принято
11	{Document.CombinedItems}	{Item.НомерСтрокиДокумента}		{Item.ШК}	{Item.Product.Marking}	{Item.Product.Name}	{Item.DeclaredQuantity}	{Item.CurrentQuar
12						Сумма:	0,00	
13								

Обратите внимание. Можно использовать разные шаблоны для разных типов документов. Для этого файл шаблона должен называться по имени типа документа. Например, «ПеремещениеПоСкладам.xlsx». Если для какого-либо типа документов не найден соответствующий ему шаблон, используется общий шаблон «Документ.xlsx».

Перед изменением шаблонов, определитесь, какие поля будут использоваться и какие могут понадобиться в будущем, чтобы не пришлось переделывать несколько раз.

Обратите внимание. Помимо удаления неактуальных полей, в платформу Mobile SMARTS можно добавить и новые, полностью уникальные поля и идентификаторы. Добавить можно простое «информационное» или сложное «вычисляемое поле».

С более полной информацией о полях можно ознакомится в статье «Дополнительные поля». Поля могут быть созданы и отображены в системе, из любых исходных полей или данных. Например, может быть создано поле, которое в определенном формате отображает сведения о товаре собранные из трех разных исходных полей.

Как добавить дополнительное поле в строку документа

Иногда возникает необходимость добавить дополнительное поле в строку документа. Например с товаром идет подарок или нигде не указанные дополнительные характеристики товара.

Как добавить дополнительное поле в строку документа можно прочитать в статье «Добавление дополнительного поля в строки документа Mobile SMARTS на примере файла Excel/CSV».

интеграция, Excel, CSV, Курьер, Магазин 15, Склад 15, ЕГАИС 3

Не нашли что искали?

Задать вопрос в техническую поддержку

Добавление дополнительного поля номенклатуры Excel/CSV

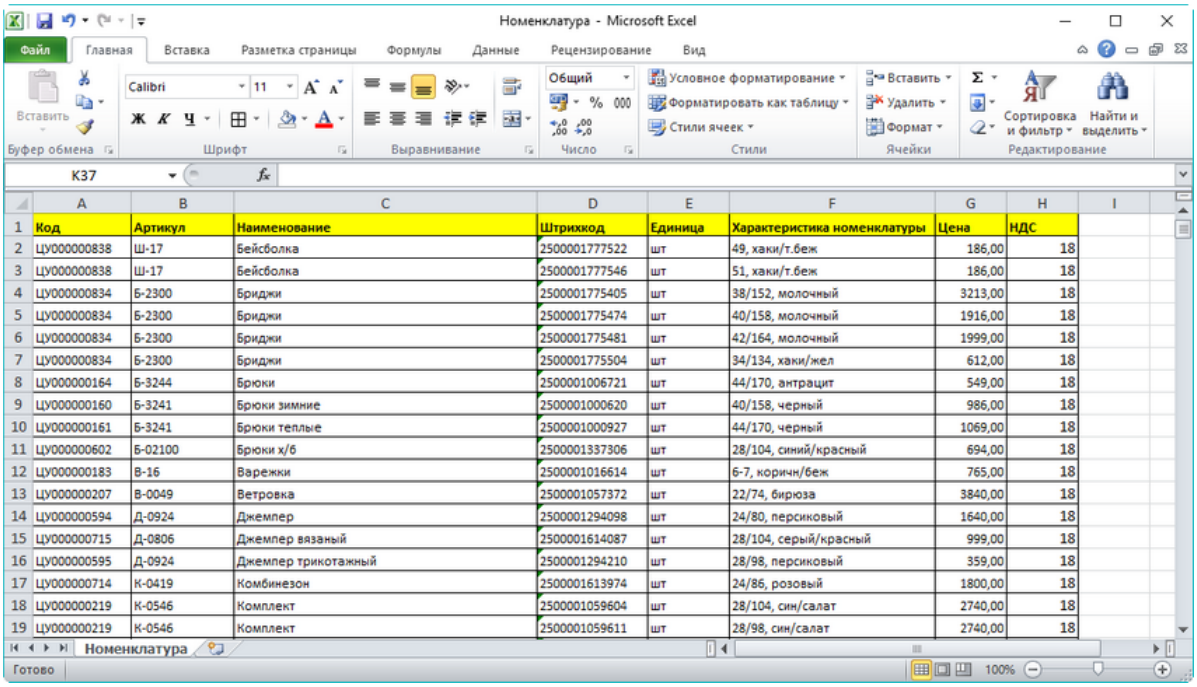
Последние изменения: 2024-03-26

Статья посвящена добавлению дополнительного поля номенклатуры. В качестве примера использован Mobile SMARTS: Курьер, обмен в формате Excel/CSV.

При использовании обмена в формате Excel/CSV, товарный справочник содержится в файле Номенклатура.xls или Номенклатура.csv (в зависимости от выбранного варианта обмена).

Находятся эти файлы в папке базы Mobile SMARTS, по пути «.../Папка базы/XlsCsv/На терминал». Далее в статье для наглядности будем использовать вариант Excel, вариант csv принципиальных отличий не имеет.

Справочник товаров по умолчанию содержит следующую информацию:



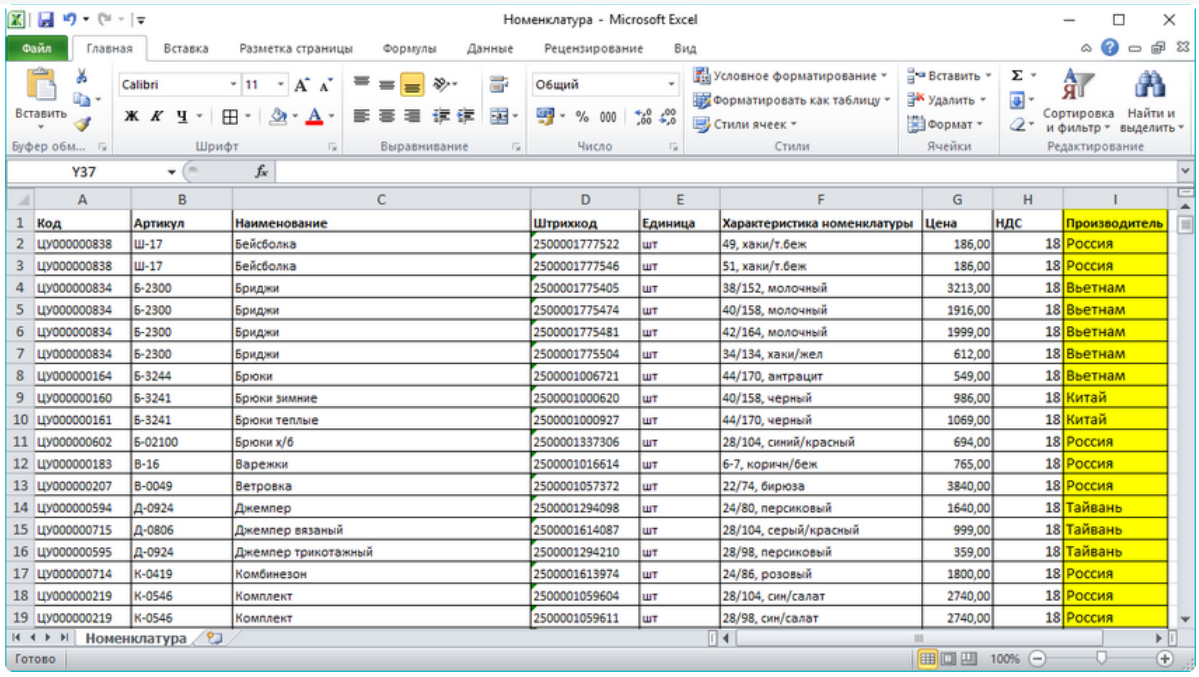
	A	B	C	D	E	F	G	H	I
1	Код	Артикул	Наименование	Штрихкод	Единица	Характеристика номенклатуры	Цена	НДС	
2	ЦУ000000838	Ш-17	Бейсболка	2500001777522	шт	49, хаки/т.беж	186,00	18	
3	ЦУ000000838	Ш-17	Бейсболка	2500001777546	шт	51, хаки/т.беж	186,00	18	
4	ЦУ000000834	Б-2300	Бриджи	2500001775405	шт	38/152, молочный	3213,00	18	
5	ЦУ000000834	Б-2300	Бриджи	2500001775474	шт	40/158, молочный	1916,00	18	
6	ЦУ000000834	Б-2300	Бриджи	2500001775481	шт	42/164, молочный	1999,00	18	
7	ЦУ000000834	Б-2300	Бриджи	2500001775504	шт	34/134, хаки/жел	612,00	18	
8	ЦУ000000164	Б-3244	Брюки	2500001006721	шт	44/170, антрацит	549,00	18	
9	ЦУ000000160	Б-3241	Брюки зимние	2500001000620	шт	40/158, черный	986,00	18	
10	ЦУ000000161	Б-3241	Брюки теплые	2500001000927	шт	44/170, черный	1069,00	18	
11	ЦУ000000602	Б-02100	Брюки х/б	2500001337306	шт	28/104, синий/красный	694,00	18	
12	ЦУ000000183	В-16	Варежки	2500001016614	шт	6-7, коричн/беж	765,00	18	
13	ЦУ000000207	В-0049	Ветровка	2500001057372	шт	22/74, бирюза	3840,00	18	
14	ЦУ000000594	Д-0924	Джемпер	2500001294098	шт	24/80, персиковый	1640,00	18	
15	ЦУ000000715	Д-0806	Джемпер вязаный	2500001614087	шт	28/104, серый/красный	999,00	18	
16	ЦУ000000595	Д-0924	Джемпер трикотажный	2500001294210	шт	28/98, персиковый	359,00	18	
17	ЦУ000000714	К-0419	Комбинезон	2500001613974	шт	24/86, розовый	1800,00	18	
18	ЦУ000000219	К-0546	Комплект	2500001059604	шт	28/104, син/салат	2740,00	18	
19	ЦУ000000219	К-0546	Комплект	2500001059611	шт	28/98, син/салат	2740,00	18	

Excel-файл с демономенклатурой (ставится вместе с продуктом)

Иногда необходимо добавить дополнительную информацию о товаре. Для того, чтобы можно было использовать эту информацию на мобильном устройстве, нужно добавить её в файл номенклатуры.

Например, возникла необходимость показывать страну-производителя. Разберёмся, как это сделать.

В файле «Номенклатура» добавляем колонку «Производитель», в которой будут содержаться данные:

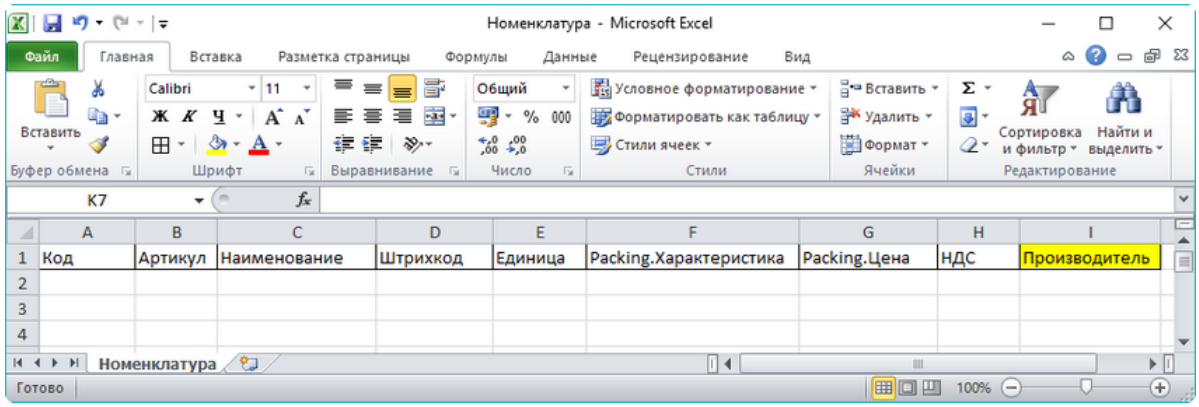


	A	B	C	D	E	F	G	H	I
1	Код	Артикул	Наименование	Штрихкод	Единица	Характеристика номенклатуры	Цена	НДС	Производитель
2	ЦУ000000838	Ш-17	Бейсболка	2500001777522	шт	49, хаки/т.беж	186,00	18	Россия
3	ЦУ000000838	Ш-17	Бейсболка	2500001777546	шт	51, хаки/т.беж	186,00	18	Россия
4	ЦУ000000834	Б-2300	Бриджи	2500001775405	шт	38/152, молочный	3213,00	18	Вьетнам
5	ЦУ000000834	Б-2300	Бриджи	2500001775474	шт	40/158, молочный	1916,00	18	Вьетнам
6	ЦУ000000834	Б-2300	Бриджи	2500001775481	шт	42/164, молочный	1999,00	18	Вьетнам
7	ЦУ000000834	Б-2300	Бриджи	2500001775504	шт	34/134, хаки/жел	612,00	18	Вьетнам
8	ЦУ000000164	Б-3244	Брюки	2500001006721	шт	44/170, антрацит	549,00	18	Вьетнам
9	ЦУ000000160	Б-3241	Брюки зимние	2500001000620	шт	40/158, черный	986,00	18	Китай
10	ЦУ000000161	Б-3241	Брюки теплые	2500001000927	шт	44/170, черный	1069,00	18	Китай
11	ЦУ000000602	Б-02100	Брюки х/б	2500001337306	шт	28/104, синий/красный	694,00	18	Россия
12	ЦУ000000183	В-16	Варежки	2500001016614	шт	6-7, коричн/беж	765,00	18	Россия
13	ЦУ000000207	В-0049	Ветровка	2500001057372	шт	22/74, бирюза	3840,00	18	Россия
14	ЦУ000000594	Д-0924	Джемпер	2500001294098	шт	24/80, персиковый	1640,00	18	Тайвань
15	ЦУ000000715	Д-0806	Джемпер вязаный	2500001614087	шт	28/104, серый/красный	999,00	18	Тайвань
16	ЦУ000000595	Д-0924	Джемпер трикотажный	2500001294210	шт	28/98, персиковый	359,00	18	Тайвань
17	ЦУ000000714	К-0419	Комбинезон	2500001613974	шт	24/86, розовый	1800,00	18	Россия
18	ЦУ000000219	К-0546	Комплект	2500001059604	шт	28/104, син/салат	2740,00	18	Россия
19	ЦУ000000219	К-0546	Комплект	2500001059611	шт	28/98, син/салат	2740,00	18	Россия

Добавленная колонка с информацией о производителе

Теперь нужно поменять шаблон загрузки номенклатуры, который находится по адресу «.../Папка базы/XlsCsv/Templates/Upload». В папке содержится два шаблона, с расширениями .xls (Excel) и .csv.

Посмотрим, что содержится в шаблоне Excel:



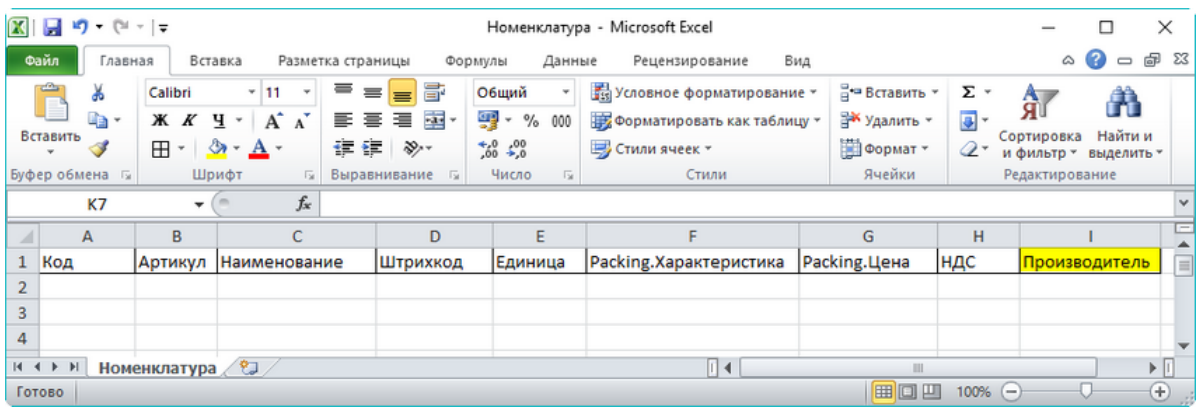
	A	B	C	D	E	F	G	H	I
1	Код	Артикул	Наименование	Штрихкод	Единица	Packing.Характеристика	Packing.Цена	НДС	Производитель
2									
3									
4									

Шаблон номенклатуры по умолчанию

Всё очень просто, шаблон содержит названия полей в том виде, в котором они будут перенесены в справочник номенклатуры на терминале сбора данных.

Этот шаблон позволяет программе понять, в каком столбце какие данные содержатся.

Добавим поле «Производитель»:

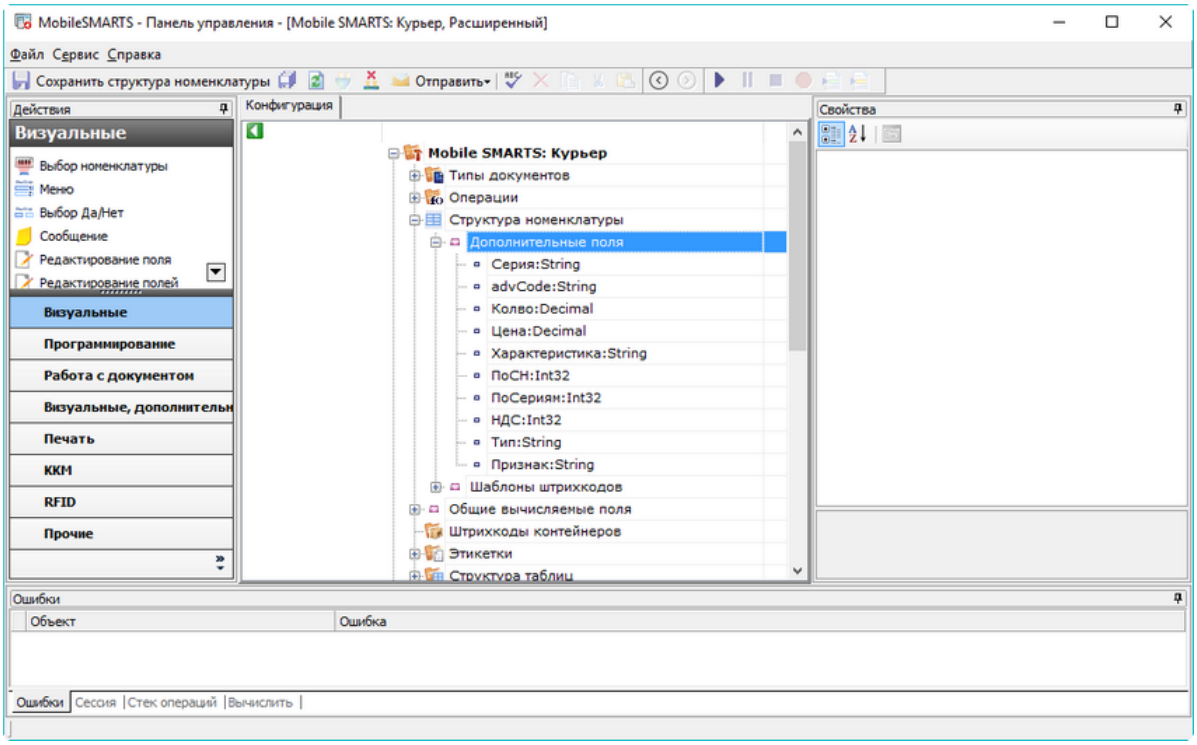


	A	B	C	D	E	F	G	H	I
1	Код	Артикул	Наименование	Штрихкод	Единица	Packing.Характеристика	Packing.Цена	НДС	Производитель
2									
3									
4									

Поле, добавленное в шаблон номенклатуры

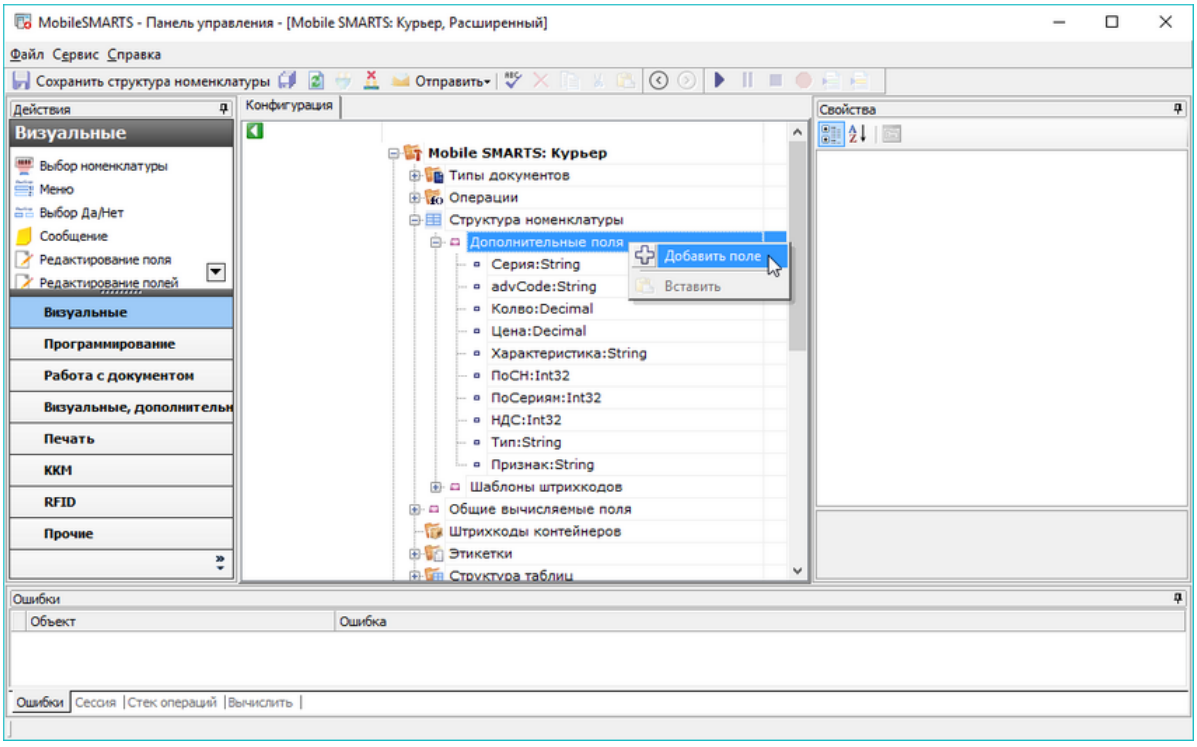
Осталось сообщить программе, что в номенклатуре появились новые данные, которые можно использовать.

Откроем конфигурацию, развернём узел «Структура номенклатуры -> Дополнительные поля»:



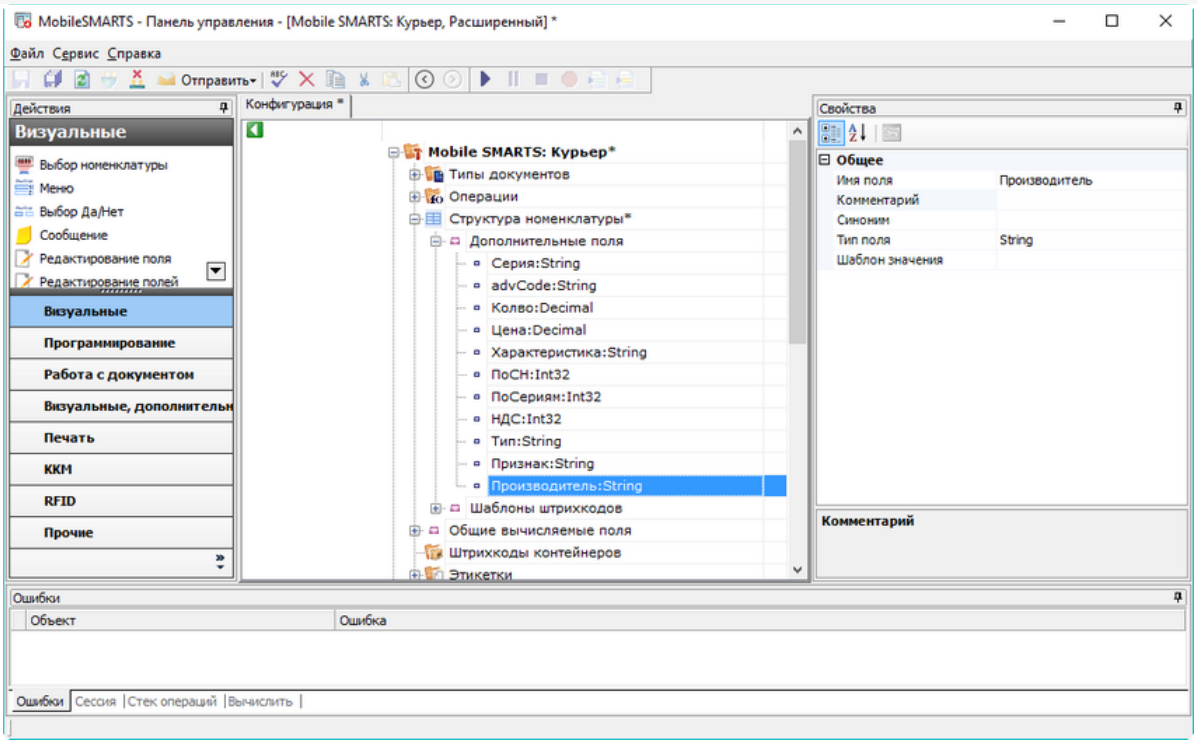
Дополнительные поля номенклатуры в конфигурации Mobile SMARTS

Правым щелчком мыши вызовем меню и добавим новое поле:



Добавление дополнительного поля номенклатуры

Имя поля должно в точности совпадать с именем, которое мы указали в шаблоне номенклатуры:

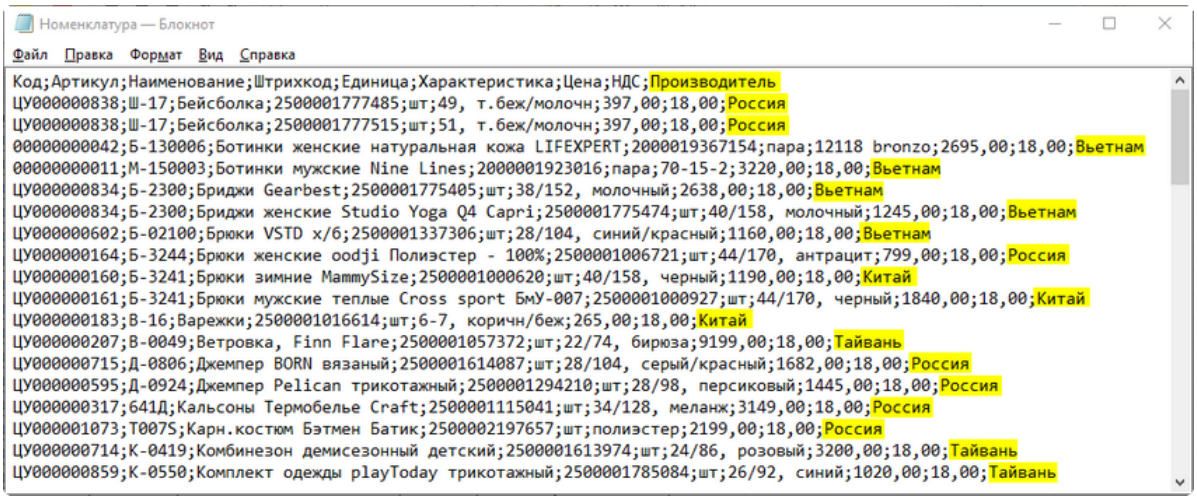


Добавленное поле

Обратите внимание, большие и маленькие буквы учитываются – если в шаблоне мы напишем «производитель», а в конфигурации «Производитель», то это будет ошибкой!

Для варианта с обменом в формате .csv нужно сделать всё то же самое, только используя соответствующий файл шаблона и справочника.

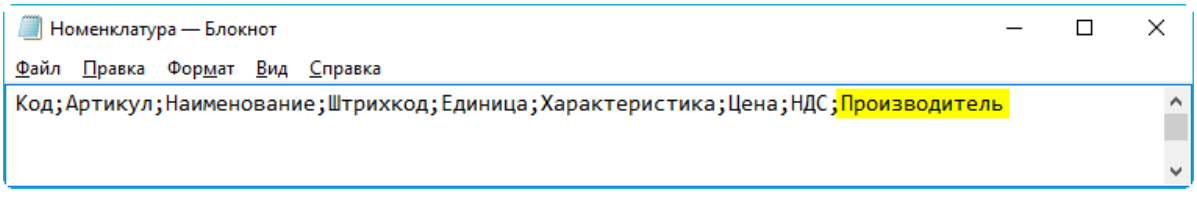
Так будет выглядеть номенклатура:



Номенклатура в формате CSV с добавленными данными

Как видно, новое поле добавлено через точку с запятой к каждой строке.

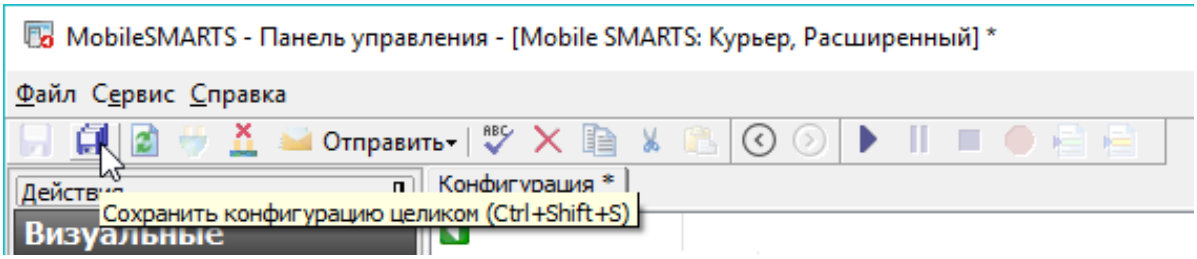
Шаблон выгрузки будет выглядеть так:



Шаблон номенклатуры в формате CSV с добавленным полем

В остальном работа с обменом в формате .csv ничем не отличается от работы с форматом Excel.

Не забудьте сохранить конфигурацию:



Произведите обмен с сервером на терминале сбора данных.

Итак, мы добавили новое поле в номенклатуру, которая выгружается на терминал сбора данных. Этот механизм позволяет дополнять номенклатуру любыми необходимыми данными. Пример использования добавленного поля, вывод его на экран мобильного устройства и запись в документ, можно посмотреть в видео.

Не нашли что искали?

Задать вопрос в техническую поддержку

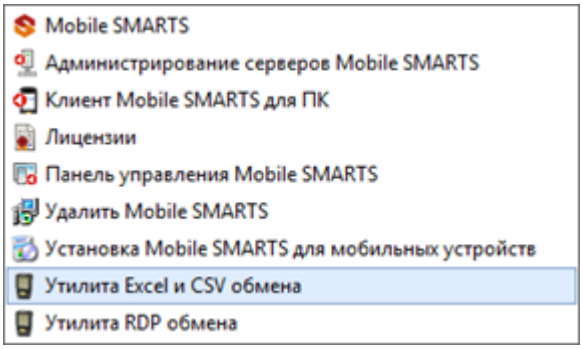
Интеграция «ЕГАИС 3» через файлы Excel/CSV

Последние изменения: 2024-03-26

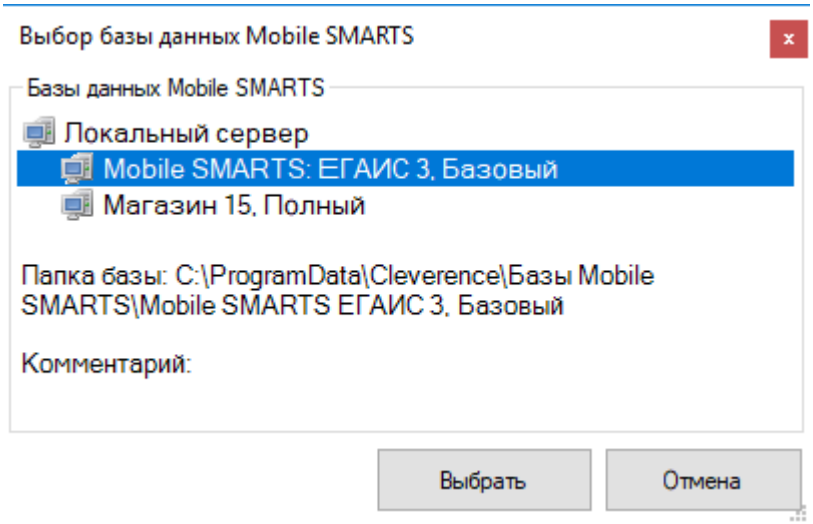
Первоначальная настройка и подключение

1. Устанавливаем платформу Mobile SMARTS и конфигурацию «ЕГАИС 3» (утилита Excel и CSV обмена входит в пакет установки платформы).

2. Запускаем утилиту (Пуск —> Программы —> Cleverence Soft —> Mobile SMARTS —> Утилита Excel и CSV обмена).



3. Выбираем базу «ЕГАИС 3», с которой будет работать утилита обмена.



Если заегистрирована только одна база, то она будет выбрана автоматически.

> В зависимости от того, в каком режиме находится база (прямая работа с ТСД или серверная) утилита также будет работать либо с сервером Mobile SMARTS, либо напрямую загружать файлы в ТСД.
> Термины «на терминал» и «с терминала» подразумевают либо прямую работу с устройством, либо работу с сервером, в зависимости от режима базы!

4. В выбранной нами базе автоматически создается папка XlsCsv с которой будет работать утилита, выгружать оттуда файлы номенклатуры и документов на терминал и загружать обратно выполненные документы.

Содержимое папки XlsCsv

Имя папки \ подпапки
Описание
На терминал
Содержит файлы Excel и CSV, предназначенные для отправки на терминал.
На терминал\Архив
Архив успешно конвертированных файлов Excel и CSV. Если файл «пропал», его можно найти здесь.
С терминала
Сюда складываются файлы с терминала после конвертации их в Excel или CSV по шаблон.
Templates
Папка с файлами шаблонов конвертации. Для «Mobile SMARTS: ЕГАИС3» готовые шаблоны добавляются в папку автоматически при установке.
Templates\Upload
Содержит шаблоны, по которым разбираются файлы для загрузки на сервер Mobile SMARTS или мобильное устройство.
Templates\Download
Содержит шаблоны, по которым формируются готовые файлы при загрузке с мобильного устройства. Для получения документов в определенном виде нужно положить сюда файл шаблона с именем типа документа, для которого предназначен шаблон.

Пример:

Если база расположена по пути «C:\ProgramData\Cleverence\Database\Mobile SMARTS ЕГАИС3», то папка для работы утилиты будет иметь путь «C:\ProgramData\Cleverence\Database\Mobile SMARTS ЕГАИС3\XlsCsv».

Папка для номенклатуры и документов, загружаемых на терминал:
«C:\ProgramData\Cleverence\Database\Mobile SMARTS ЕГАИС3\XlsCsv\На терминал».

Папка с выполненными документами: «c:\ProgramData\Cleverence\Database\Mobile SMARTS ЕГАИС3\XlsCsv\С терминала».

5. Настраиваем параметры обмена.

Выбираем формат файлов для загрузки/выгрузки.

Обмен данными

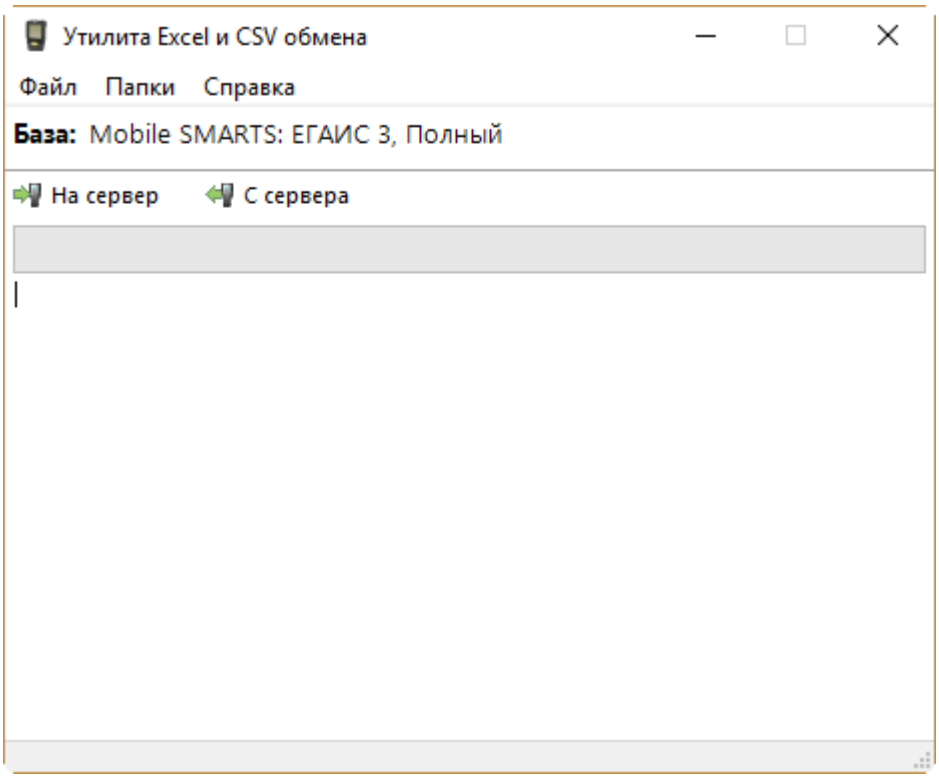
Обмен данными происходит в два этапа:

- Этап первый — выгрузка номенклатуры и документов «На сервер» (1).

  При нажатии на кнопку «На сервер» происходит выгрузка номенклатуры и документов на сервер (на ТСД в батч режиме).

- Этап второй — загрузка документов «С сервера» (2).

  При нажатии на кнопку «С сервера» происходит загрузка выполненных документов с сервера (с ТСД в батч режиме).



Этап первый

Выгрузка номенклатуры

> Для выгрузки номенклатуры на терминал необходимо положить в папку «На терминал» файл «Номенклатура.csv» или «Номенклатура.xls» (также поддерживается формат «.xlsx»), в зависимости от используемого формата файлов.

Формат обмена (какие данные и в каком порядке идут в файле) задаётся в файле «...\XlsCsv\Templates\Upload\Номенклатура.csv» (или «.xls"/».xlsx»).

Формат для CSV по умолчанию имеет следующий вид:

```
Код;Артикул;Наименование;Product.Штрихкод;Product.BasePackingId;Коэффициент;Packing.Barco
```

В шаблоне, через точку с запятой (без пробелов), перечисляются поля номенклатуры и упаковки, которые будем выгружать.

Для Excel формат обмена имеет аналогичную структуру.

Шаблон можно изменять или вообще создать свой, с колонками, которые необходимы для выгрузки.

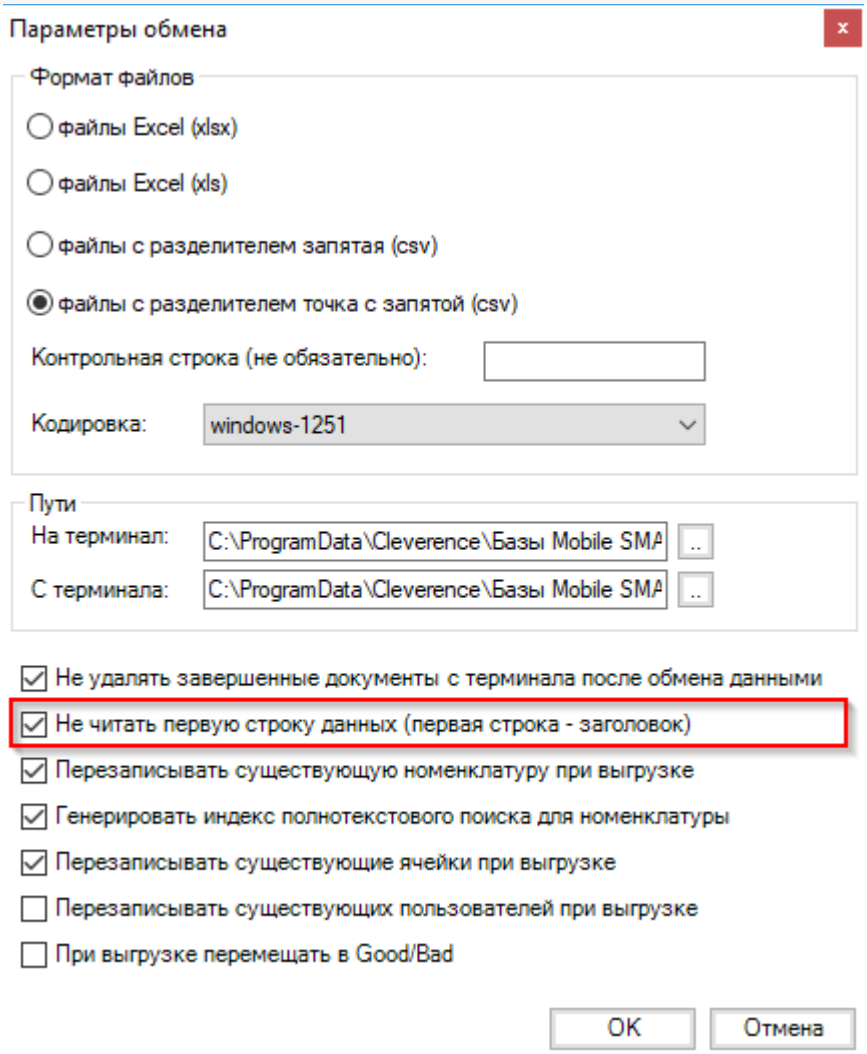
Примеры

Пример корректного входного файла «Номенклатура.csv»:

```
Ид;Артикул;Наименование;Код;Базовая упаковка;Коэфф.;Штрихкод;Упаковка;Кол. ед.
изм.;Остаток;Цена;Весовой;Алкоголь;Маркируется;Код ЕГАИС;Крепость;Объем
(л);Производитель/Импортёр;КодВ;НаимВ;Имя ЕГАИС
0b4389ad-c7d9-46f4-809f-e2f1d9920b1a;;"ЧИСТЫЙ СОСТАВ ВОДКАЯ  40%
0,25Л";40242;шт;0;4601728012175;шт;1;15;175.9;Нет;Да;Да;;40;0.25;"ООО
""Омсквинпром""";;Другие спиртные напитки с содержанием этилового спирта свыше
25%;"ЧИСТЫЙ СОСТАВ ВОДКАЯ  40% 0,25Л"
0b4389ad-c7d9-46f4-809f-e2f1d9920b1a;;"ЧИСТЫЙ СОСТАВ ВОДКАЯ  40%
0,25Л";40242;шт;0;00155450000002458847;шт;1;15;175.9;Нет;Да;Да;00155450000002458847;40;0.2
""Омсквинпром""";;Другие спиртные напитки с содержанием этилового спирта свыше
25%;"ЧИСТЫЙ СОСТАВ ВОДКАЯ  40% 0,25Л"
```

Комментарии:

- **В качестве первой строки можно выгружать произвольные имена колонок, для удобства человека. Для этого в утилите добавлена настройка «Не читать первую строку данных»;**



- Каждая строка должна иметь ровно тоже число колонок, что задано в формате обмена. Если для данной позиции нет данных, то просто пропускаем его, ставя разделитель «;;»;

- Значения, содержащие зарезервированные символы (двойная кавычка, запятая, точка с запятой, новая строка) обрамляются двойными кавычками («); если в значении встречаются кавычки — они представляются в файле в виде двух кавычек подряд;

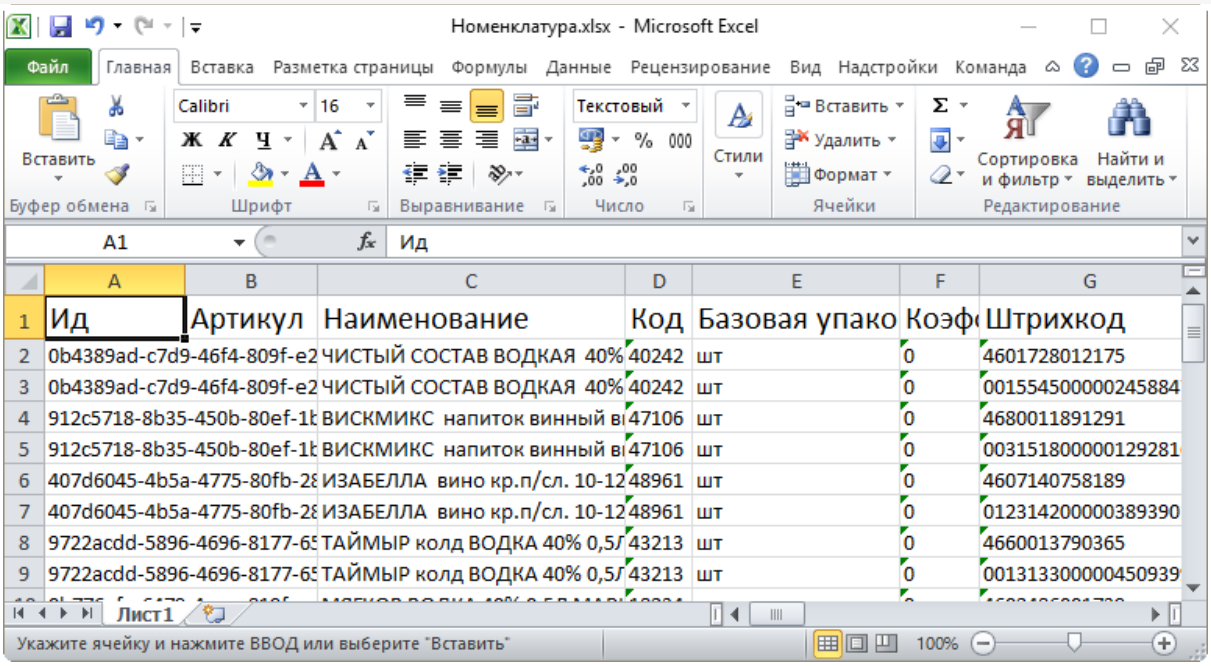
- Подробнее про формат CSV Вы можете прочитать по ссылке https://ru.wikipedia.org/wiki/CSV

Пример файла Excel для выгрузки номенклатуры

Путь к файлам для выгрузки номенклатуры: «...\Database\Mobile SMARTS ЕГАИС3\XlsCsv\На терминал»

> Файлы Excel для выгрузки должны иметь один лист, который содержит загружаемые данные.

Описание строк задается в виде таблицы, содержащей все колонки (даже если какие-нибудь колонки остаются пустыми), которые есть в шаблоне. Каждая колонка может иметь ячейку-заголовок, для удобства при просмотре человеком. Если ваш Excel файл не имеет строки заголовков, то необходимо отключить опцию «Не читать первую строку данных».
Колонки в документе с данными должны идти в том же порядке, что и в файле шаблона.



Выгрузка документов

> Все созданные документы для выгрузки необходимо класть в папку «На терминал («...\XlsCsv\На терминал»).

Файл документа для выгрузки должен иметь имя, начинающееся с имени типа документа (например, ОтгрузкаЕГАИС3).
Примеры:

- **ОтгрузкаЕГАИС3 Без подбора.csv**
- **ПоступлениеЕГАИС3 Полное.xls**

Шаблоны для выгрузки документов

> Шаблон для выгрузки документа должен иметь название, совпадающее с типом документа, например, «ОтгрузкаЕГАИС3.csv» или «ОтгрузкаЕГАИС3.xls», в зависимости от используемого формата файлов.

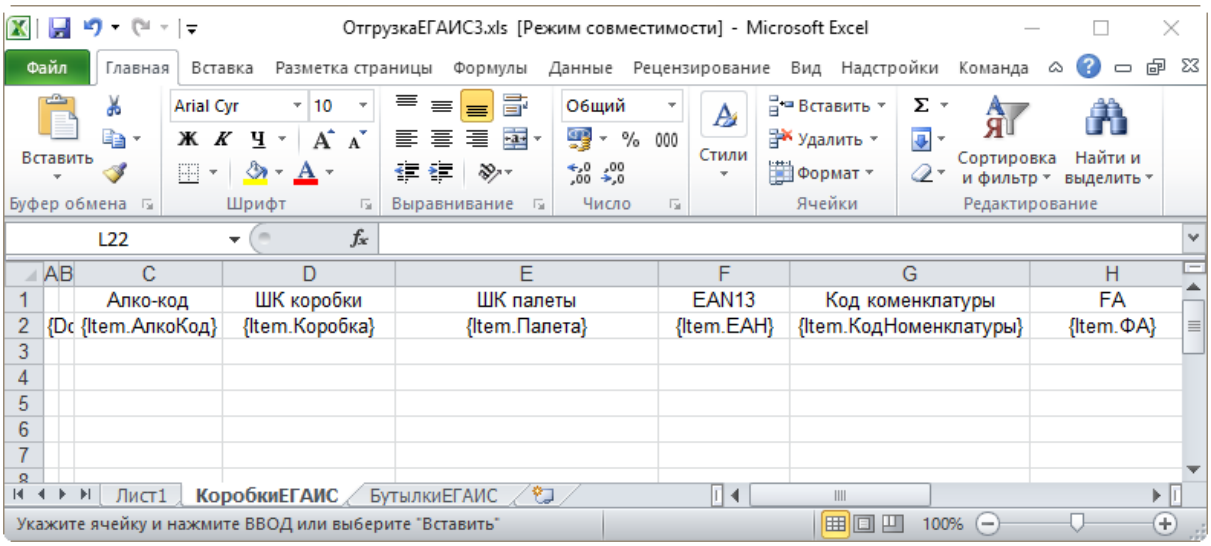
Путь к шаблонам: «...\Database\Mobile SMARTS ЕГАИС 3\XlsCsv\Templates\Upload»
Шаблон для выгрузки документа ОтгрузкаЕГАИС3:

Файл CSV:

```
#{Document}
КоробкиПодбора
#{DeclaredItems}
{Item.СкладСерия};{Item.АлкоКод};{Item.АлкоНаим};{Item.ProductId};{Item.DeclaredQuantity}
#{КоробкиЕГАИС}
{Item.АлкоКод};{Item.Коробка};{Item.Палета};{Item.EAH};{Item.КодНоменклатуры};
{Item.ФА};{Item.ФБ};{Item.ДатаРозлива};{Item.СкладСерия};{Item.АлкоНаим};
{Item.Количество}
#{БутылкиЕГАИС}
{Item.АлкоПДФ};{Item.АлкоКод};{Item.Коробка};{Item.Палета};{Item.ДМ};{Item.EAH};
{Item.КодНоменклатуры};{Item.ФА};{Item.ФБ};{Item.ДатаРозлива};{Item.СкладСерия};
{Item.АлкоНаим}
```

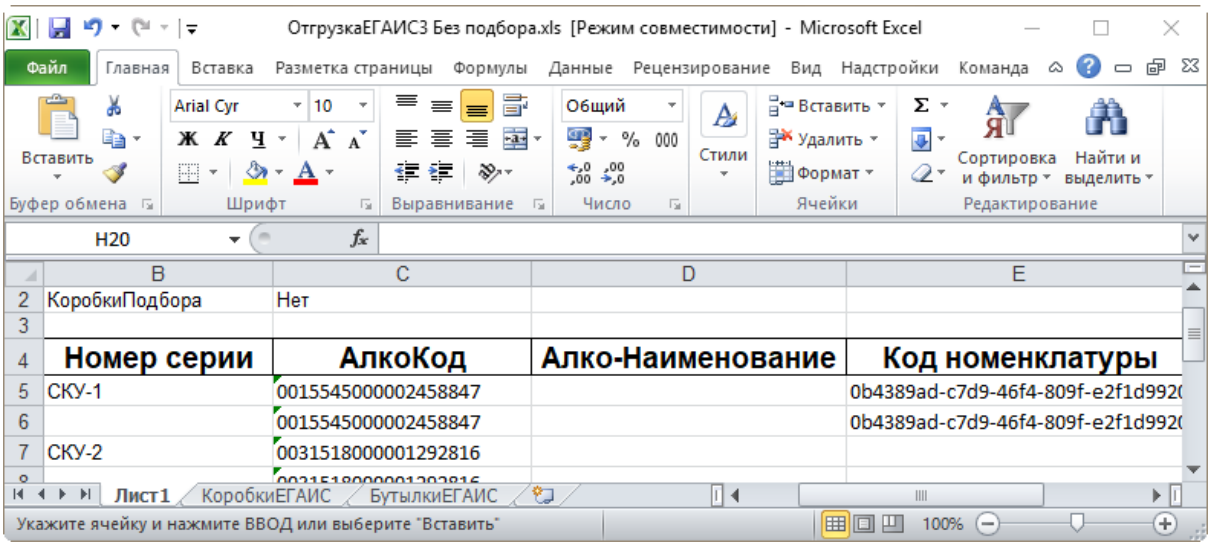
В шаблоне, через точку с запятой (без пробелов), перечисляются поля, которые будем выгружать.

Для Excel формат обмена имеет следующую структуру:



Примеры

Пример корректного входного файла Excel документа «ОтгрузкаЕГАИС3 Без подбора»:



> **Файлы Excel для выгрузки документа должны иметь несколько листов, которые содержат выгружаемые данные.**

Этап второй

Загрузка документов

Файл документа для загрузки будет иметь имя, начинающееся с имени типа документа (например, ПоступлениеЕГАИС3).

Итоговые загруженные файлы документа создаются автоматически по шаблону и попадают в папку «... \XlsCsv\С терминала».

Шаблон документа для загрузки с ТСД описывает какие данные и в каком порядке попадут в итоговый файл Excel или csv.

Шаблон для загрузки документов с ТСД на примере документа Поступление

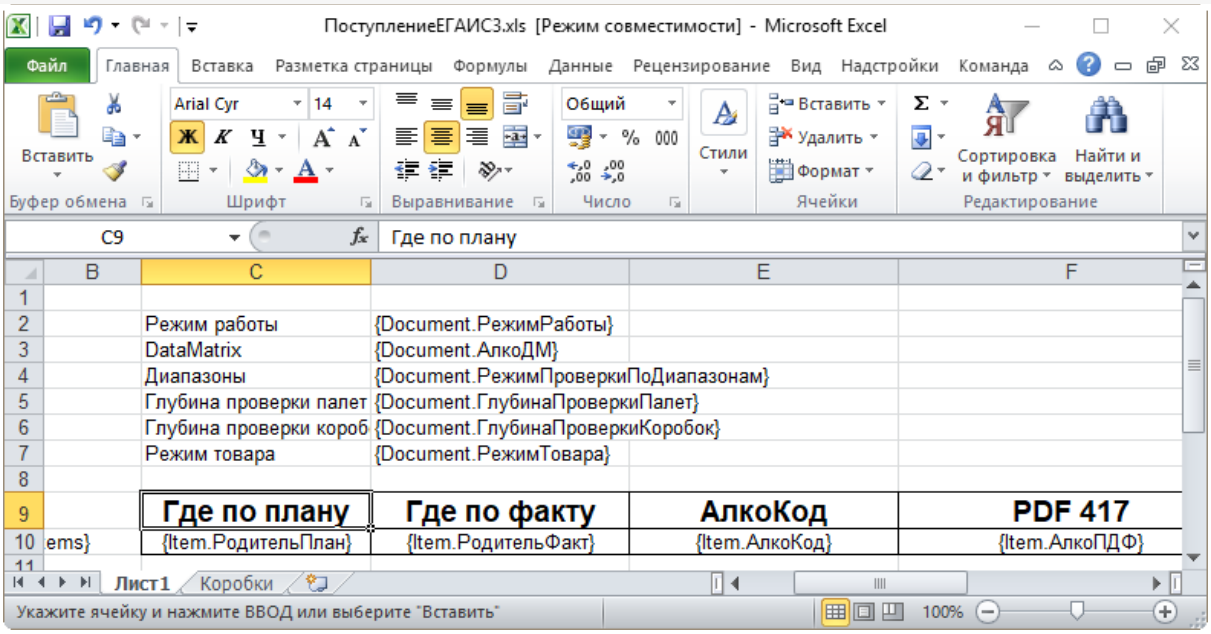
Путь к шаблонам «...\XlsCsv\Templates\Download».

Файл CSV:

```
#{Document}
РежимРаботы;АлкоДМ;РежимПроверкиПоДиапазонам;ГлубинаПроверкиПалет;ГлубинаПроверки

#{CombinedItems}
{Item.РодительПлан};{Item.РодительФакт};{Item.АлкоКод};{Item.АлкоПДФ};{Item.ProductId};
{Item.DeclaredQuantity};{Item.CurrentQuantity}
#{Коробки}
{Item.Ид};{Item.СерийныйНомерКоробки};{Item.РодительПлан};{Item.РодительФакт};
{Item.Товар};{Item.Количество};{Item.Партия};{Item.СН};{Item.ЭтоПалета};
{Item.Номенклатура};{Item.План};{Item.Факт}
```

В шаблоне, через точку с запятой (без пробелов), перечисляются поля шапки и табличных частей документа, которые будут загружаться в итоговый файл.

Файл Excel:



Шаблоны можно изменять или вообще создать свой, с колонками, которые необходимы вам для загрузки.

Не нашли что искали?

Задать вопрос в техническую поддержку

Интеграция «Mobile SMARTS: КИЗ» через CSV и Excel

Последние изменения: 2024-03-26

Перед тем, как приступить к работе, необходимо выполнить первоначальную настройку и подключение. После этого можно приступать к обмену данными, который происходит в два этапа:

Этап первый – выгрузка номенклатуры «На сервер».

Этап второй – загрузка документов «С сервера».

После проведения маркировки и загрузки документа, результаты нужно выгрузить в систему Маркировка (ФНС).

Настройка утилиты Excel и CSV обмена

Настройка шаблона для GLN

Обмен данными Excel

Обмен данными CSV

Загрузка результатов на сайт Маркировки

КИЗ, интеграция

Не нашли что искали?

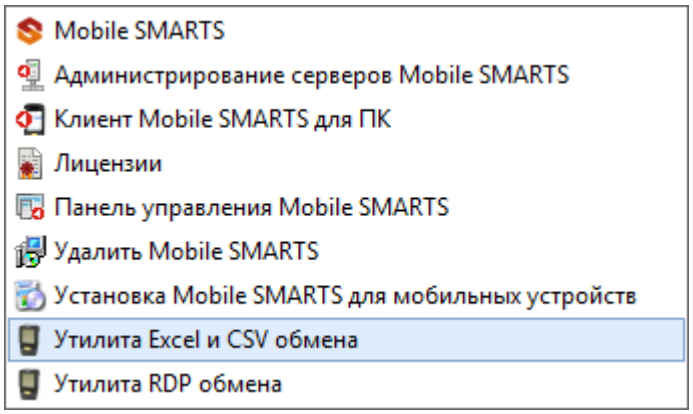
Задать вопрос в техническую поддержку

Интеграция «Mobile SMARTS: ЕГАИС» через CSV и Excel

Последние изменения: 2024-03-26

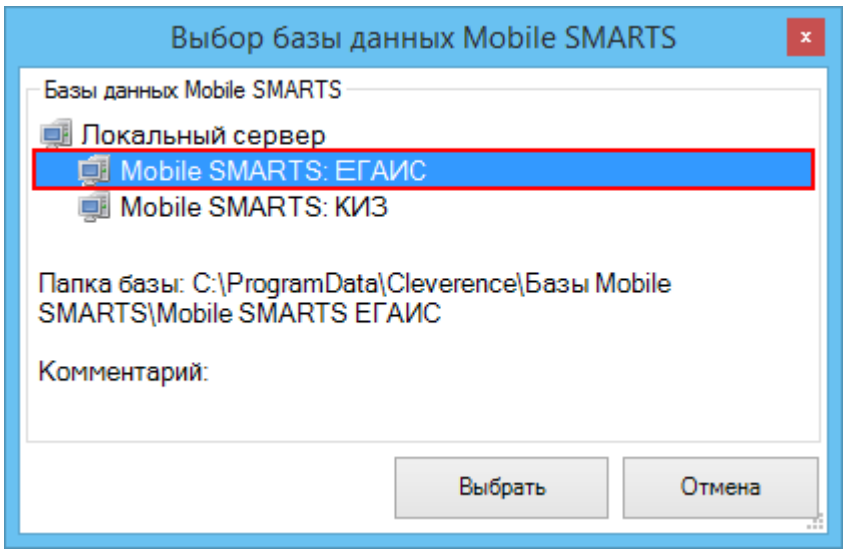
Первоначальная настройка и подключение

1. Устанавливаем платформу Mobile SMARTS и конфигурацию ЕГАИС (утилита Excel и CSV обмена входит в пакет установки платформы).
2. Запускаем утилиту (Пуск — Программы — Cleverence Soft — Mobile SMARTS — Утилита Excel и CSV обмена).



3. Выбираем базу «Mobile SMARTS: ЕГАИС» с которой будет работать утилита обмена.

> Если зарегистрирована только одна база, то она будет выбрана автоматически.



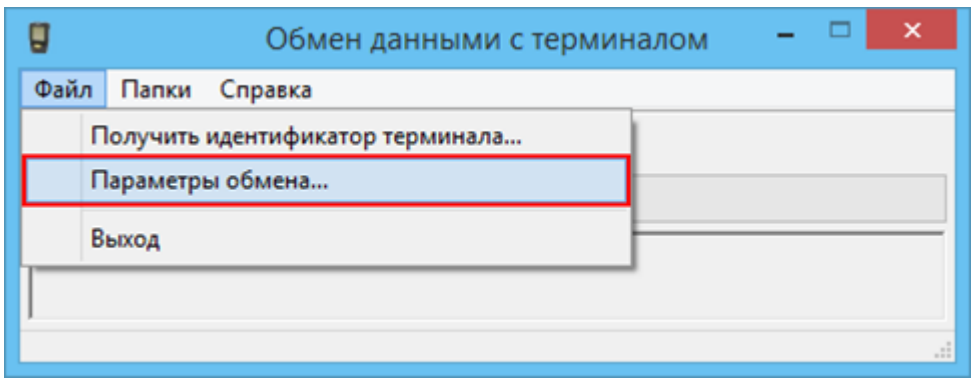
> В зависимости от того в каком режиме находится база (прямая работа с ТСД или серверная) утилита также будет работать либо с сервером Mobile SMARTS, либо напрямую загружать файлы в ТСД.
> **Термины «на терминал» и «с терминала» подразумевают либо прямую работу с устройством, либо работу с сервером, в зависимости от режима базы!**

4. В выбранной нами базе автоматически создается папка XlsCsv с которой будет работать утилита, выгружать оттуда файлы номенклатуры и документов на терминал и загружать обратно выполненные документы.
   Содержимое папки XlsCsv

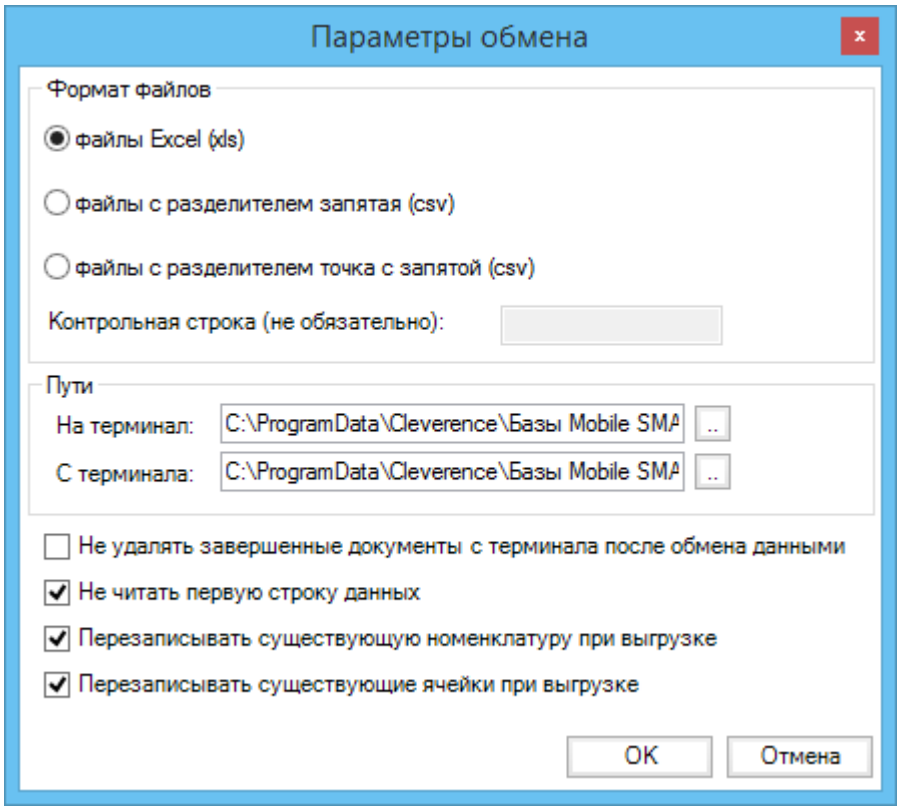
Имя папки \ подпапки	
Описание	
На терминал	
Содержит файлы Excel и CSV, предназначенные для отправки на терминал.	
На терминал\Архив	
Архив успешно конвертированных файлов Excel и CSV. Если файл «пропал», его можно найти здесь.	
С терминала	
Сюда складываются файлы с терминала после конвертации их в Excel или CSV по шаблон.	
Templates	
Папка с файлами шаблонов конвертации. Для «Mobile SMARTS: ЕГАИС» **готовые шаблоны добавляются в папку автоматически** при установке.	
Templates\Upload	
Содержит шаблоны, по которым разбираются файлы для терминала.	
Templates\Download	
Содержит шаблоны, по которым формируются готовые файлы с терминала. Для получения документов в определенном виде нужно положить сюда файл шаблона с именем типа документа, для которого предназначен шаблон.	

Пример:
Если база расположена по пути «c:\ProgramData\Cleverence\Databases\Mobile SMARTS ЕГАИС», то папка для работы утилиты будет иметь путь «c:\ProgramData\Cleverence\Databases\Mobile SMARTS ЕГАИС\XlsCsv».
Папка для номенклатуры и документов, загружаемых на терминал: «c:\ProgramData\Cleverence\Databases\Mobile SMARTS ЕГАИС\XlsCsv\На терминал».
Папка с выполненными документами: «c:\ProgramData\Cleverence\Databases\Mobile SMARTS ЕГАИС\XlsCsv\С терминала».

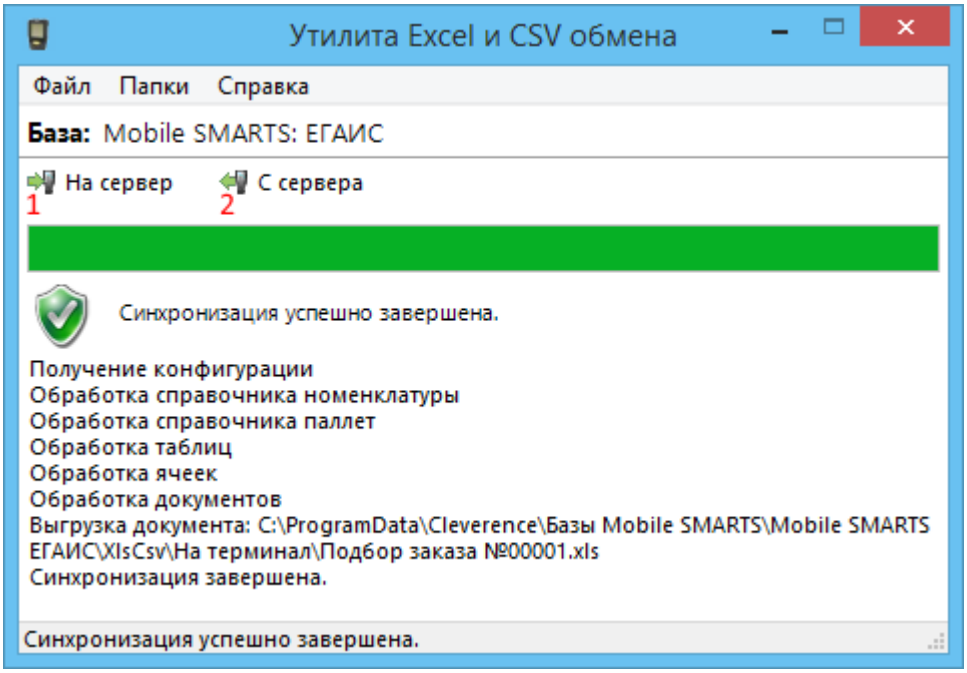
5. Настраиваем параметры обмена.



Выбираем формат файлов для загрузки/выгрузки.



Обмен данными



Обмен данными происходит в два этапа:

Этап первый — выгрузка номенклатуры и документов «На сервер» (1).

При нажатии на кнопку «На сервер» происходит выгрузка номенклатуры и документов на сервер (на ТСД в батч режиме).

Этап второй — загрузка документов «С сервера» (2).

При нажатии на кнопку «С сервера» происходит загрузка выполненных документов с сервера (с ТСД в батч режиме).

Этап первый

Выгрузка номенклатуры

Для выгрузки номенклатуры на терминал необходимо положить в папку «На терминал» файл «Номенклатура.csv» или «Номенклатура.xls», в зависимости от используемого формата файлов.

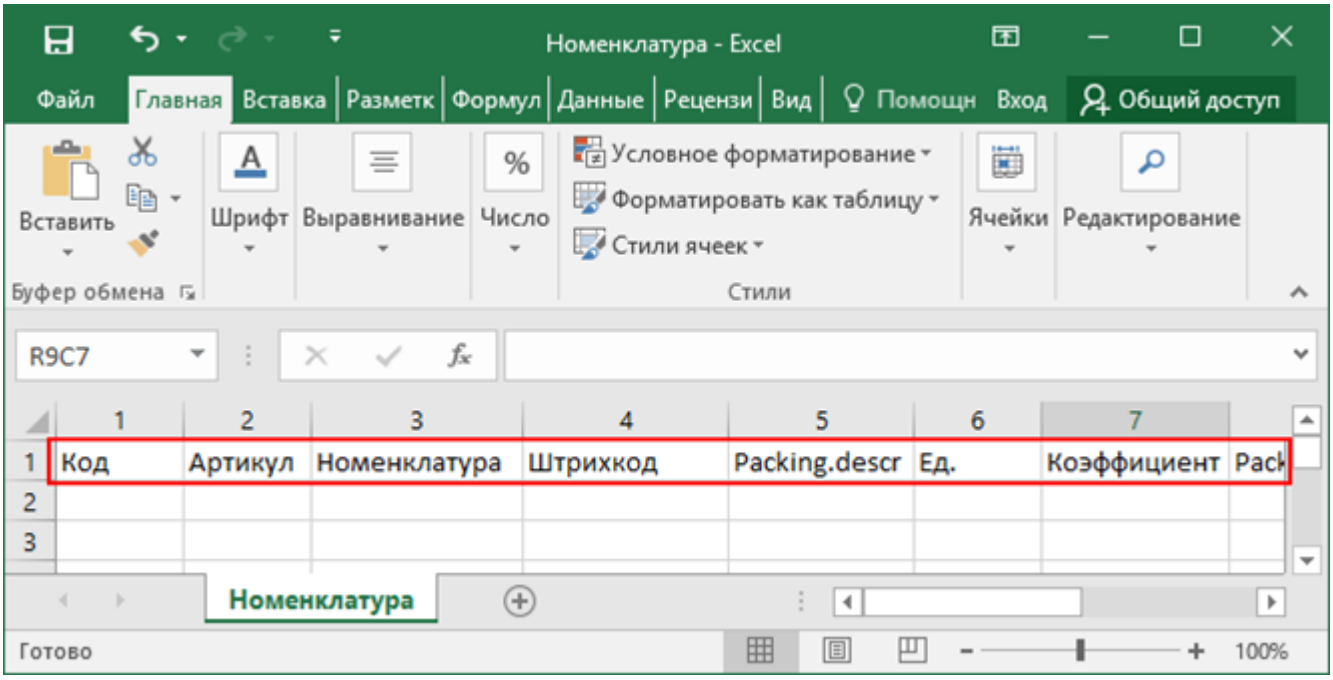
Формат обмена (какие данные и в каком порядке идут в файле) задаётся в файле «... \XlsCsv\Templates\Upload\Номенклатура.csv» (или «.xls»).

Формат для CSV по умолчанию имеет следующий вид:

```
Код;Артикул;Номенклатура;Штрихкод;Packing.descr;Ед.;Коэффициент;Packing.serial;Packing.qty;

Packing.price;Product.Алко;Product.АлкоМарк;Packing.АлкоКод;Packing.АлкоКрепость;
Packing.АлкоОбъем;Packing.Производитель
```

В шаблоне, через точку с запятой (без пробелов), перечисляются поля номенклатуры и упаковки, которые будем выгружать.

Для Excel формат обмена имеет аналогичную структуру.



Шаблон можно изменять или вообще создать свой, с колонками, которые необходимы для выгрузки.

Колонки для шаблона номенклатуры:

Название колонки	Описание
Код	Код

Артикул	**Содержит артикул товара, что позволяет искать товар на терминале по его артикулу.**
Номенклатура	**Наименование товара.**
Штрихкод	**Штрихкод товара (обычно EAN13).**
Packing.descr	**Характеристика товара (если ведется учет с характеристиками).**
Ед.	**Единицы измерения (например, бут., шт.).**
Коэффициент	**Коэффициент пересчета единиц измерения продукции.**
Packing.serial;	**Серия, привязанная к упаковке (для серийного учета).**
Packing.qty	**Количество товара на складе.**
Packing.price	**Стоимость единицы товара.**
Product.Алко	**Является алкоголе содержащей продукцией (Да/Нет).**
Product.АлкоМарк	**Является маркируемой продукцией (Да/Нет).**
Packing.АлкоКод	**Код упаковки алкогольной продукции.**
Packing.АлкоКрепость	**Крепость.**
Packing.АлкоОбъем	**Объем.**
Packing.Производитель	**Производитель.**

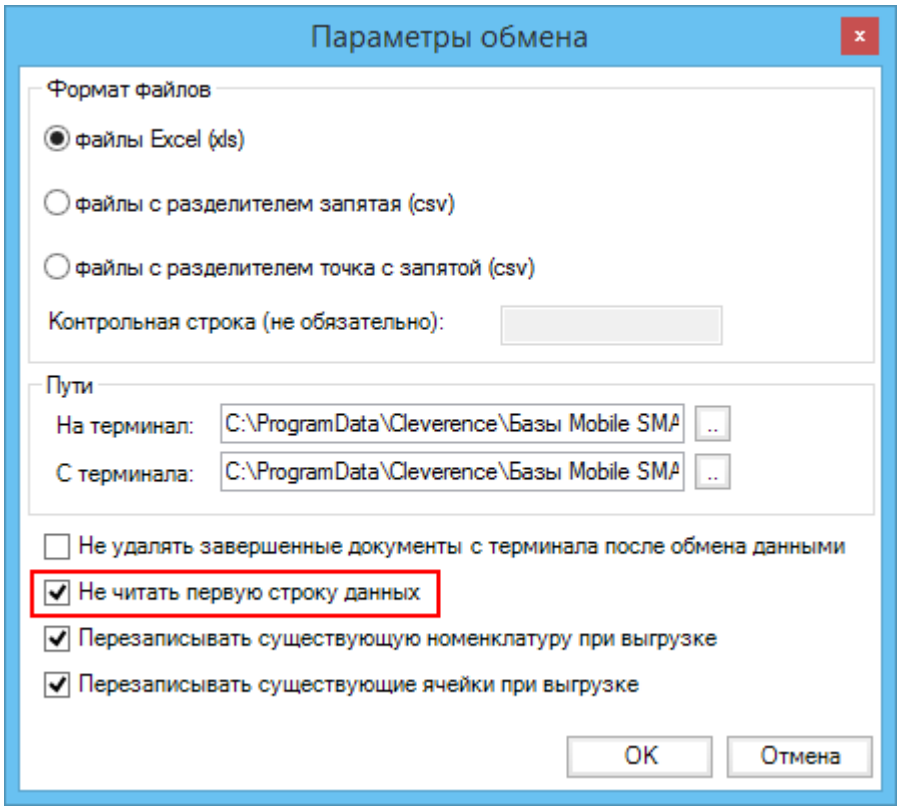
Примеры

Пример корректного входного файла «Номенклатура.csv»:

```
Код;Артикул;Номенклатура;Штрихкод;Характеристика;Единица
измерения;Коэффициент;Серия;Количество;Цена;Алкоголь;Наличие марки;Код
ЕГАИС;Крепость;Объём;Производитель/импортёр

00000005879;;"Водка
""Зимняя""";4680012890095;;бт;1,00;;;;Да;Да;1127448000053545176;40;0,1;

00000005880;Арт435455;"Водка""Сордис""";4601126221995;;бт;1,00;;;;Да;Да;05505120000472062
40;0,1;"ООО ""СОРДИС"""
```

Комментарии:

- **В качестве первой строки можно выгружать произвольные имена колонок, для удобства человека. Для этого в утилите добавлена настройка «Не читать первую строку данных»;**



- Каждая строка должна иметь ровно тоже число колонок, что задано в формате обмена. Если для данной позиции нет данных, то просто пропускаем его, ставя разделитель «;;»;
- Значения, содержащие зарезервированные символы (двойная кавычка, запятая, точка с запятой, новая строка) обрамляются двойными кавычками («); если в значении встречаются кавычки — они представляются в файле в виде двух кавычек подряд;
- Подробнее про формат CSV Вы можете прочитать по ссылке https://ru.wikipedia.org/wiki/CSV.

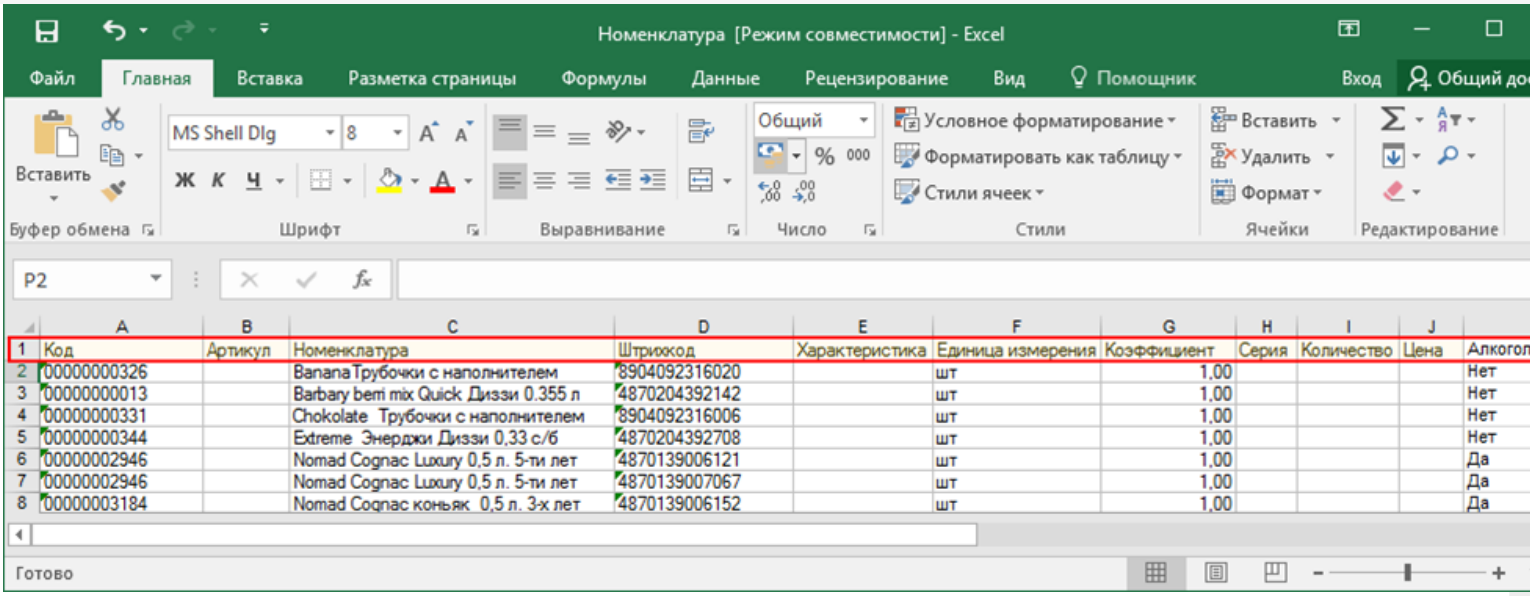
Пример файла Excel для выгрузки номенклатуры

Путь к файлам для выгрузки номенклатуры: «...\Databases\Mobile SMARTS ЕГАИС\XlsCsv\На терминал»

> Файлы Excel для выгрузки должны иметь **один лист**, который содержит загружаемые данные.

Описание строк задается в виде таблицы, содержащей все колонки (даже если какие-нибудь колонки остаются пустыми), которые есть в шаблоне. Каждая колонка может иметь ячейку заголовок, для удобства при просмотре человеком. Если ваш excel файл не имеет строки заголовков, то необходимо отключить опцию «Не читать первую строку данных».

Колонки в документе с данными должны идти в том же порядке, что и в файле шаблона.



	A	B	C	D	E	F	G	H	I	J	
1	Код	Артикул	Номенклатура	Штрихкод	Характеристика	Единица измерения	Коэффициент	Серия	Количество	Цена	Алкогол
2	00000000326		Banana Трубочки с наполнителем	8904092316020		шт	1,00				Нет
3	00000000013		Barbary berri mix Quick Дизи 0.355 л	4870204392142		шт	1,00				Нет
4	00000000331		Chokolate Трубочки с наполнителем	8904092316006		шт	1,00				Нет
5	00000000344		Extreme Энерджи Дизи 0,33 с/б	4870204392708		шт	1,00				Нет
6	00000002946		Nomad Cognac Luxury 0,5 л. 5-ти лет	4870139006121		шт	1,00				Да
7	00000002946		Nomad Cognac Luxury 0,5 л. 5-ти лет	4870139007067		шт	1,00				Да
8	00000003184		Nomad Cognac коньяк 0,5 л. 3-х лет	4870139006152		шт	1,00				Да

Выгрузка ФормыА

Некоторые операции, например, приход на склад позволяют проводить автоматическую проверку и подбор номеров форм «А», если конечно данные об этих формах выгрузить заранее.

> **Для выгрузки таблицы форм «А» на терминал необходимо положить в папку «На терминал» файл «ФормыА.csv» или «ФормыА.xls», в зависимости от используемого формата файлов.**

Шаблон для выгрузки «ФормыА»

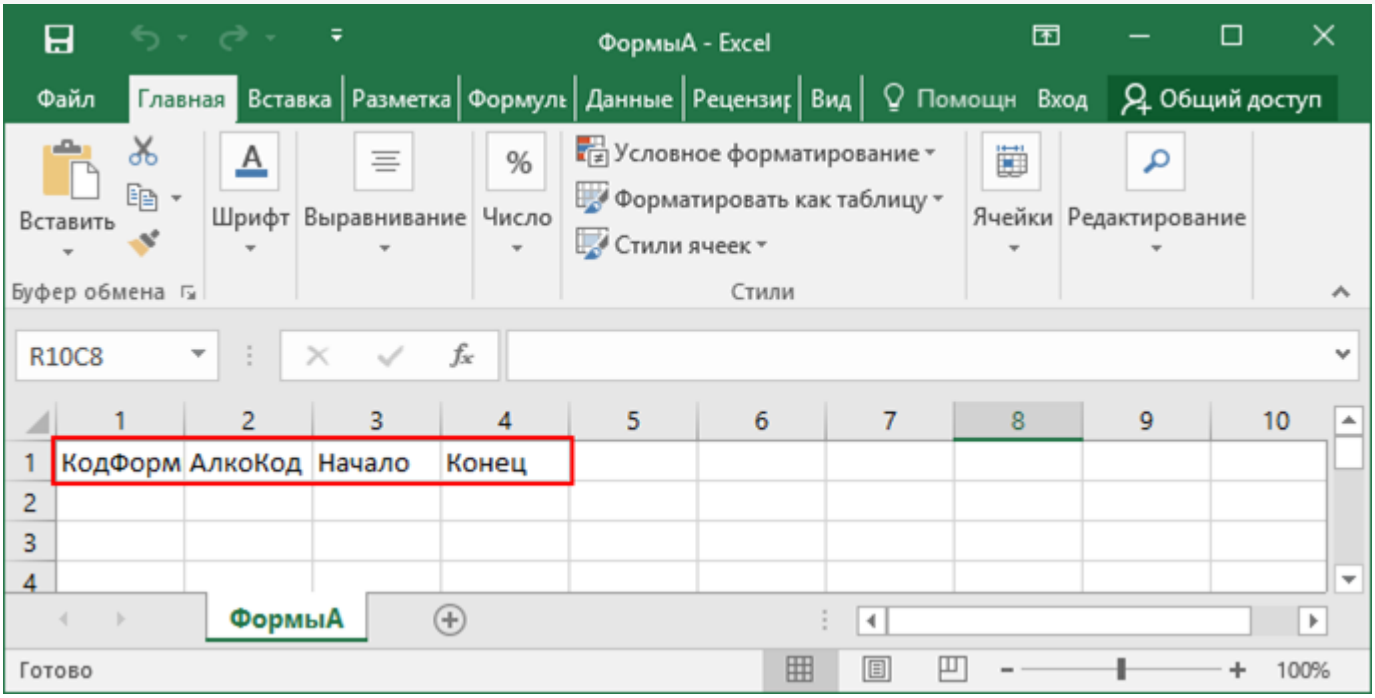
Путь к шаблону: «...\Databases\Mobile SMARTS ЕГАИС\XlsCsv\Templates\Upload»

Файл CSV:

```
КодФормы;АлкоКод;Начало;Конец
```

В шаблоне, через точку с запятой (без пробелов), перечисляются поля Формы А, которые будем выгружать.

Для Excel формат обмена имеет аналогичную структуру.



Колонки для шаблона «ФормыА»:

Название колонки	Описание
КодФормы	Код формы в системе ЕГАИС.
АлкоКод	Строка с кодом алкогольной продукции в ЕГАИС.
Начало	Начало диапазона серийных номеров.
Конец	Конец диапазона серийных номеров.

Примеры

Пример корректного входного файла «ФормыА.csv»:

```
КодФормы;АлкоКод;Начало;Конец

123456;00313180000001257558;102033827000;102033828000

654321;00313180000001257580;100684707541;100684707762
```

Выгрузка документов

> Все созданные Вами документы для выгрузки необходимо класть в папку «На терминал» («...\XlsCsv\На терминал»).

Файл документа для выгрузки должен иметь имя, начинающееся с имени типа документа (например, Сбор начальных остатков), после которого идет небуквенный и нецифровой символ, например, «№» или «#», после которого можно указать номер или дату документа.

Примеры:

- Сбор начальных остатков № 003
- Приход на склад # 26.01.2016
- Инвентаризация # 00078
- Подбор заказа № 0053 от 10.02.2016

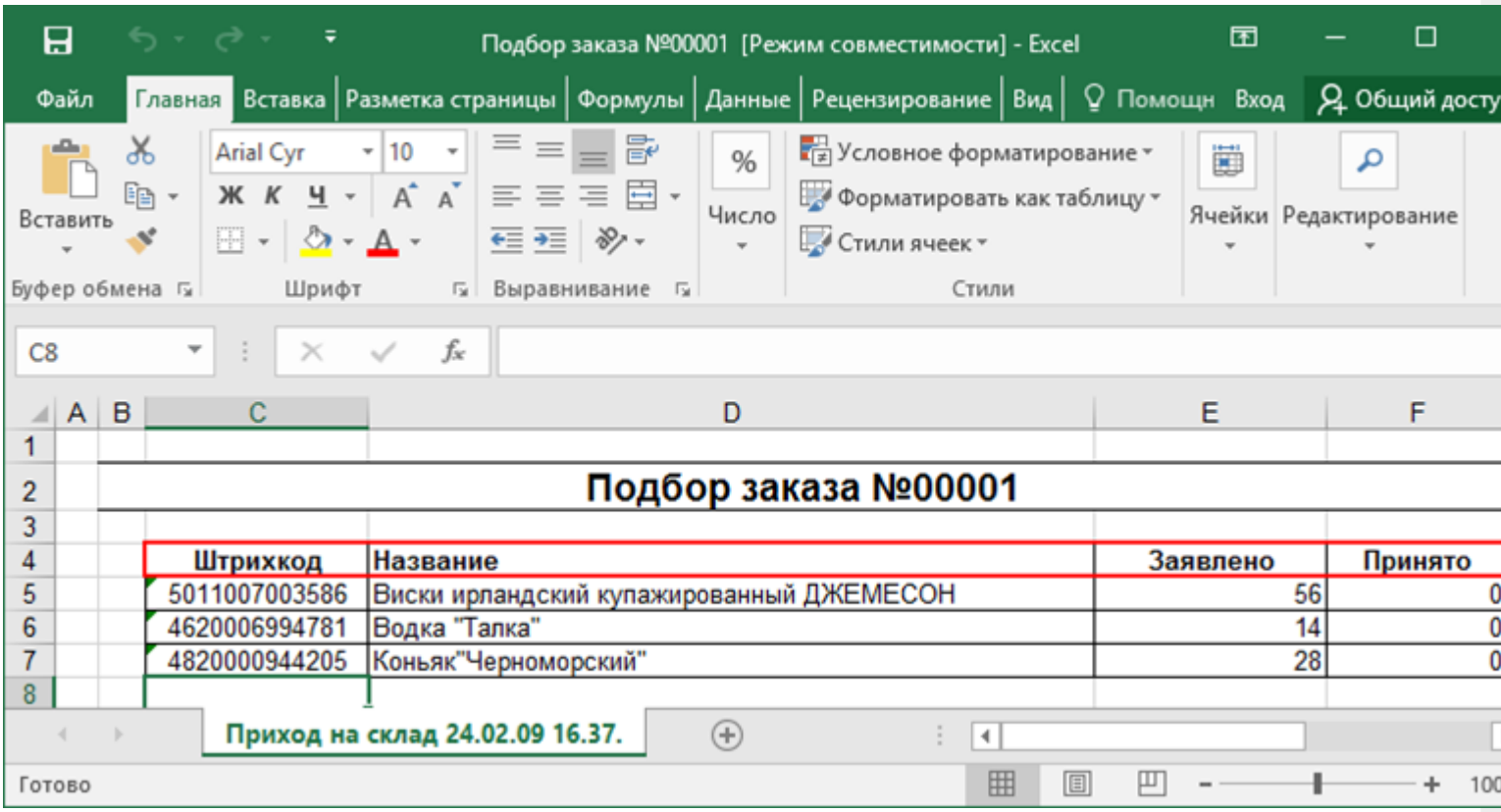
> Шаблон для выгрузки документов на ТСД выгружать не обязательно. Выгружаем готовый документ, который одновременно может быть шаблоном, тогда обязательно каждая колонка должна иметь ячейку заголовок, содержащую название колонки.

Примеры

Пример корректного входного файла Excel документа «Подбор заказа №00001»:

> Файлы Excel для выгрузки документа должны иметь один лист, который содержит выгружаемые данные.

Описание строк задается в виде таблицы, содержащей колонки. Каждая колонка имеет ячейку заголовок, содержащую название колонки и данные.



Этап второй

Загрузка документов

Файл документа для загрузки будет иметь имя, начинающееся с имени типа документа (Сбор начальных остатков), после которого идет небуквенный и нецифровой символ, например, «№» или «#».

> Итоговые загруженные файлы документа создаются автоматически по шаблону и попадают в папку «... \XlsCsv\С терминала».

Шаблон документа для загрузки с ТСД описывает какие данные и в каком порядке попадут в итоговый файл excel или csv.

Шаблон для загрузки документов с ТСД

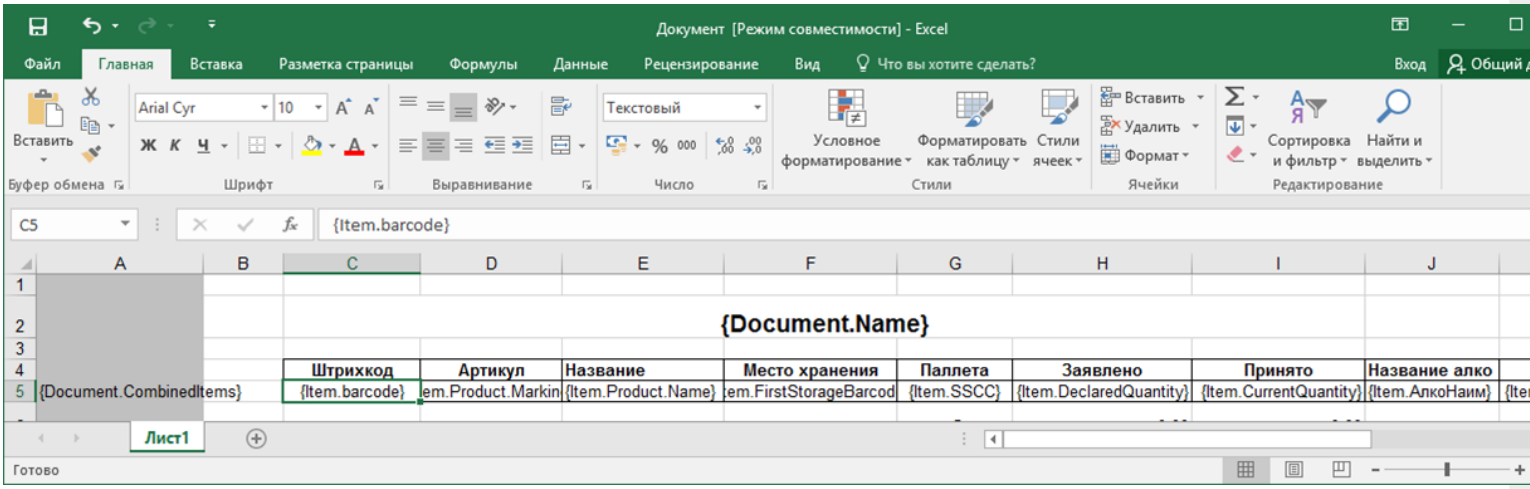
Путь к шаблонам «...\XlsCsv\Templates\Download».

Файл CSV:

```
#{Document};;;;;;;;;;;;;;;;;
Имя;;;;;;;;;;;;;;;;;
#{Document.CombinedItems};;;;;;;;;;;;;;;;;
barcode;Артикул;Наименование;Ячейка;SSCC;План;Факт;АлкоНаим;АлкоОбъем;АлкоКрепость;
Производитель;ПроизвИНН;ПроизвКПП;АлкоКод;АлкоПДФ;АлкоСН;ДатаРозлива;АлкоЕстьВФорма
```

В шаблоне, через точку с запятой (без пробелов), перечисляются поля шапки и табличной части документа, которые будут загружаться в итоговый файл.

Файл Excel:



Шаблоны можно изменять или вообще создать свой, с колонками, которые необходимы Вам для загрузки.

Колонки для шаблона загрузки документа:

Название колонки	Описание
#{Document}	В CSV файле указание, что далее идут поля шапки документа для загрузки.

# {Document.CombinedItems} #{Document.CurrentItems} # {Document.DeclaredItems}	В CSV файле указание, из какой табличной части документа загружаются итоговые данные.
Штрихкод	Штрихкод товара (обычно EAN13).
Артикул	Содержит артикул товара.
Наименование	Наименование товара.
Ячейка	Место хранения.
SSCC	Палета.
План	Заявленное количество.
Факт	Фактическое количество, то которое отсканировали.
АлкоНаим	Наименование товара из ЕГАИС, если уже известно для данного товара, либо если было получено из CheckMark.
АлкоОбъем	Ёмкость тары в литрах из ЕГАИС, если уже известна для данного товара, либо если была получена из CheckMark.
АлкоКрепость	Процентное содержание спирта из ЕГАИС, если уже известно для данного товара, либо если был получен из CheckMark.
Производитель	Наименование производителя продукции в ЕГАИС, если уже известно для данного товара, либо если было получено из CheckMark.
ПроизвИНН	ИНН производителя, если уже известен для данного товара, либо если был получен из CheckMark.
ПроизвКПП	КПП производителя, если уже известен для данного товара, либо если был получен из CheckMark.
АлкоКод	Код алкогольной продукции в ЕГАИС, полученный из отсканированных марок.
АлкоПДФ	Строка с PDF 417.
АлкоСН	Серийный номер бутылки, если сканировался.
ДатаРозлива	Дата розлива позиции, если вводилась, иначе null.
АлкоЕстьВФормах	Признак того, что данная марка была найдена в выгруженных формах А.
КодФормыА	Код формы А, где была найдена марка.

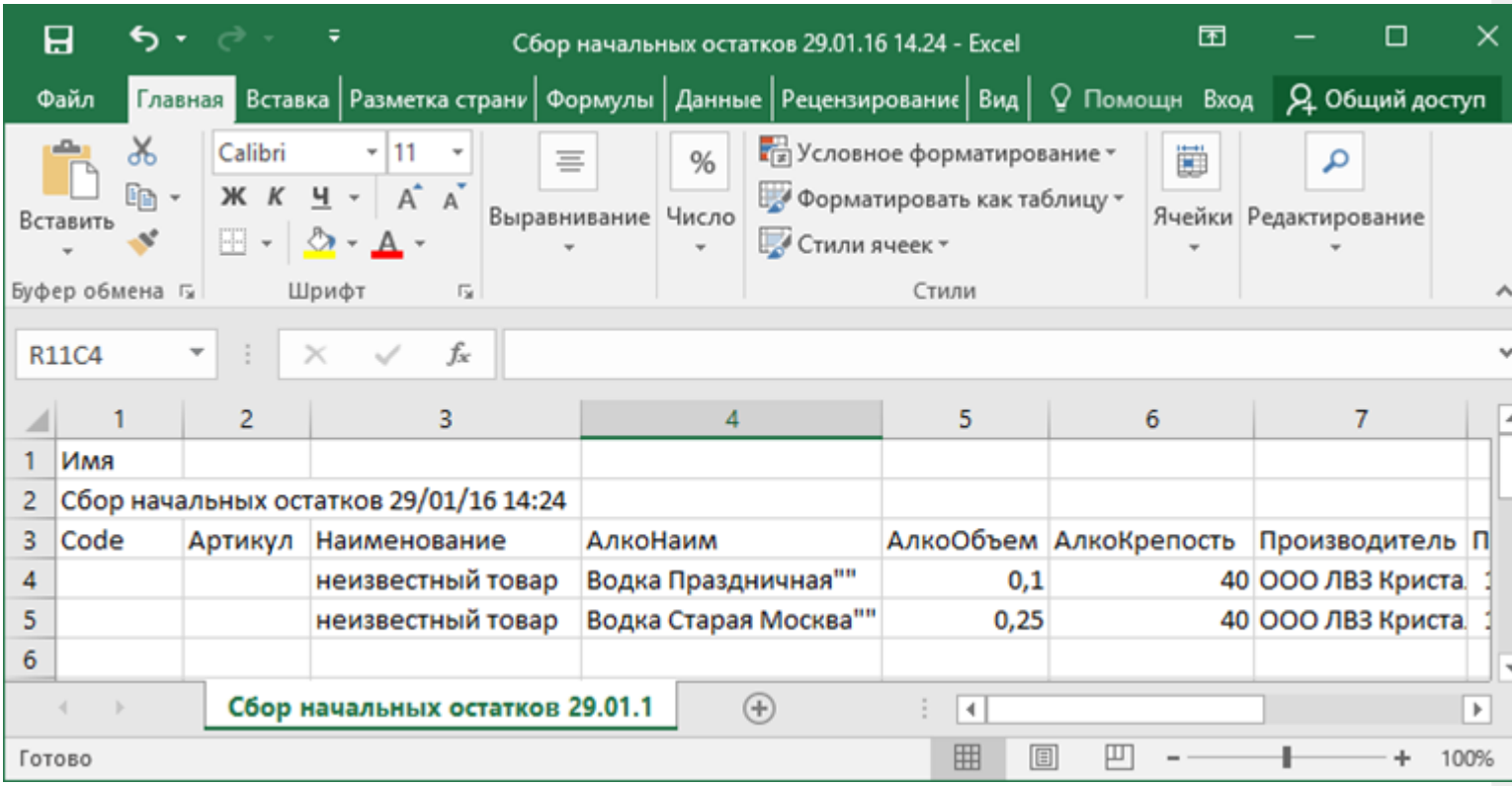
Примеры

Файл CSV загруженный с ТСД (например, Сбор начальных остатков 29.01.16 14.24.csv)

```
Имя;;;;;;;;;;;;;;;;

"Сбор начальных остатков 29/01/16 14:24";"";"";"";"";"";"";"";"";"";"";"";"";"";"";"";""

Code;Артикул;Наименование;АлкоНаим;АлкоОбъем;АлкоКрепость;Производитель;ПроизвИНН;Пр

АлкоКод;АлкоПДФ;АлкоСН;ДатаРозлива;АлкоЕстьВФормах;КодФормыА;DeclaredQuantity;Current(

"";"";"неизвестный товар";"Водка "Праздничная"";"0,1";"40,000";"ООО "ЛВЗ Кристалл-
Лефортово"";"1328005717";"132801001";"0031318000001257558";
"22N000008KDAYKXY8GM0O5Y51021002014736N7MLO27PKF77030CEGBIJZZRG7HDPIQ";"102033829

"";"0";"1"

"";"";"неизвестный товар";"Водка "Старая Москва"";"0,25";"40,000";"ООО "ЛВЗ Кристалл-
Лефортово"";"1328005717";"132801001";"0031318000001257580";
"22N000008KDAYKXY8H80O5Y509160030475418YBZSTRII1S4SQPEJRYYNBDQ5WX51YR";"100684707
```

Незаполненные поля загружаются в файл в виде разделителя ««««;».

Файл Excel загруженный с ТСД (например, Сбор начальных остатков 29.01.16 14.24. xls)



ЕГАИС, интеграция, утилита

Не нашли что искали?

Задать вопрос в техническую поддержку

Интеграция «Магазина 15» через CSV и Excel

Последние изменения: 2024-03-26

Первоначальная настройка и подключение

1. Устанавливаем платформу Mobile SMARTS и конфигурацию «Магазин 15» (утилита Excel и CSV обмена входит в пакет установки платформы).
2. Запускаем утилиту (Пуск — Программы — Cleverence Soft — Mobile SMARTS — Утилита Excel и CSV обмена).

3. Выбираем базу «Магазин 15» с которой будет работать утилита обмена.
   Если зарегистрирована только одна база, то она будет выбрана автоматически.

> В зависимости от того в каком режиме находится база (прямая работа с ТСД или серверная) утилита также будет работать либо с сервером Mobile SMARTS, либо напрямую загружать файлы в ТСД.
> **Термины «на терминал» и «с терминала» подразумевают либо прямую работу с устройством, либо работу с сервером, в зависимости от режима базы!**

4. В выбранной нами базе автоматически создается папка XlsCsv с которой будет работать утилита, выгружать оттуда файлы номенклатуры и документов на терминал и загружать обратно выполненные документы.
   Содержимое папки XlsCsv

Имя папки \ подпапки	
Описание	
На терминал	
Содержит файлы Excel и CSV, предназначенные для отправки на терминал.	
На терминал\Архив	
Архив успешно конвертированных файлов Excel и CSV. Если файл «пропал», его можно найти здесь.	
С терминала	
Сюда складываются файлы с терминала после конвертации их в Excel или CSV по шаблон.	
Templates	
Папка с файлами шаблонов конвертации. Для «Mobile SMARTS: Магазин 15» **готовые шаблоны добавляются в папку автоматически** при установке.	
Templates\Upload	
Содержит шаблоны, по которым разбираются файлы для терминала.	

Templates\Download
Содержит шаблоны, по которым формируются готовые файлы с терминала. Для получения документов в определенном виде нужно положить сюда файл шаблона с именем типа документа, для которого предназначен шаблон.

Пример:
Если база расположена по пути «c:\ProgramData\Cleverence\Databases\Mobile SMARTS Магазин 15», то папка для работы утилиты будет иметь путь «c:\ProgramData\Cleverence\Базы Mobile SMARTS\Mobile SMARTS Магазин 15\XlsCsv».
Папка для номенклатуры и документов, загружаемых на терминал: «c:\ProgramData\Cleverence\Databases\Mobile SMARTS Магазин 15\XlsCsv\На терминал».
Папка с выполненными документами: «c:\ProgramData\Cleverence\Databases\Mobile SMARTS Магазин 15\XlsCsv\С терминала».

5. **Настраиваем параметры обмена.**

Выбираем формат файлов для загрузки/выгрузки.

Обмен данными

Обмен данными происходит в два этапа:

Этап первый — выгрузка номенклатуры и документов «На сервер» (1).

При нажатии на кнопку «На сервер» происходит выгрузка номенклатуры и документов на сервер (на ТСД в батч режиме).

Этап второй — загрузка документов «С сервера» (2).

При нажатии на кнопку «С сервера» происходит загрузка выполненных документов с сервера (с ТСД в батч режиме).

Этап первый

Выгрузка номенклатуры

Для выгрузки номенклатуры на терминал необходимо положить в папку «На терминал» файл «Номенклатура.csv» или «Номенклатура.xls», в зависимости от используемого формата файлов.

Формат обмена (какие данные и в каком порядке идут в файле) задаётся в файле «... \XlsCsv\Templates\Upload\Номенклатура.csv» (или «.xls»).

Формат для CSV имеет следующий вид:

```
Код;Артикул;Наименование;Штрихкод;Единица;Коэффициент;Характеристика номенклатуры;Остаток;
```

```
Цена;Алкоголь;Маркируется маркой;Код ЕГАИС;Крепость;Объем (л);Производитель/Импортёр
```

В шаблоне, через точку с запятой (без пробелов), перечисляются поля номенклатуры и упаковки, которые будем выгружать.

> **Поля для алкогольной продукции** необходимы только при работе с алкоголем.

Для Excel формат обмена имеет аналогичную структуру.

Шаблон можно изменять или вообще создать свой, с колонками, которые необходимы для выгрузки.

Колонки для шаблона номенклатуры:

Название колонки	Описание
Код	Код
Артикул	Содержит артикул товара, что позволяет искать товар на терминале по его артикулу.
Наименование	Наименование товара.
Штрихкод	Штрихкод товара (обычно EAN13).
Единица	Единицы измерения (например, бут., шт.)
Коэффициент	
Характеристика номенклатуры	Характеристика товара (если ведется учет с характеристиками).
Остаток	Количество товара на складе.
Цена	Стоимость единицы товара.
Алкоголь	Является алкоголе содержащей продукцией (Да/Нет).
Маркируется маркой	Является маркируемой продукцией (Да/Нет).
Код ЕГАИС	Код упаковки алкогольной продукции.
Крепость	Крепость алкогольной продукции.
Объем (л)	Объем алкогольной продукции.
Производитель/Импортер	Производитель, импортер алкогольной продукции.

Примеры

Пример корректного входного файла «Номенклатура.csv»:

```
Код;Артикул;Наименование;Штрихкод;Единица;Коэффициент;Характеристика номенклатуры;Остаток;Цена;Алкоголь;
```

```
Маркируется маркой;Код ЕГАИС;Крепость;Объем (л);Производитель/Импортёр

00000005879;;"Водка ""Зимняя дорога""";4680012890095;бут;1;;;;Да;Да;11274480000535451 76;40;0,1;"ООО ""СОРДИС"""

ЦУ000000110;80;STEP Пазл 80;4602827770010;шт;1;;115,00;26,00;;;;;;

ЦУ000000167;5826;Агуша пюре Груша 115г;4602541004156;шт;1;;33,00;23,00;;;;;;
```

Комментарии:

- **В качестве первой строки можно выгружать произвольные имена колонок, для удобства человека. Для этого в утилите добавлена настройка «Не читать первую строку данных»;**

- **Каждая строка должна иметь ровно тоже число колонок, что задано в формате обмена. Если для данной позиции нет данных, то просто пропускаем его, ставя разделитель «;;»;**
- **Значения, содержащие зарезервированные символы (двойная кавычка, запятая, точка с запятой, новая строка) обрамляются двойными кавычками («); если в значении встречаются кавычки — они представляются в файле в виде двух кавычек подряд;**
- **Подробнее про формат CSV Вы можете прочитать по ссылке https://ru.wikipedia.org/wiki/CSV.**

Пример файла Excel для выгрузки номенклатуры

Путь к файлам для выгрузки номенклатуры: «...\Databases\Mobile SMARTS Магазин 15\XlsCsv\На терминал».

> **Файлы Excel для выгрузки должны иметь один лист, который содержит выгружаемые данные.**

Описание строк задается в виде таблицы, содержащей все колонки (даже если какие-нибудь колонки остаются пустыми), которые есть в шаблоне. Каждая колонка может иметь ячейку заголовок, для удобства при просмотре человеком. Если ваш excel файл не имеет строки заголовков, то необходимо отключить опцию «Не читать первую строку данных».

Колонки в документе с данными должны идти в том же порядке, что и в файле шаблона.

Выгрузка ФормыА

> **Выгрузка данных форм «А» нужна только для работы с алкогольной продукцией. Если Вы не работаете с алкоголем, выгружать не нужно.**

Некоторые операции, например, приход на склад позволяют проводить автоматическую проверку и подбор номеров форм «А», если конечно данные об этих формах выгрузить заранее.

> Для выгрузки таблицы форм «А» на терминал необходимо положить в папку «На терминал» файл «ФормыА.csv» или «ФормыА.xls», в зависимости от используемого формата файлов.

Шаблон для выгрузки «ФормыА»

Путь к шаблону: «...\Databases\Mobile SMARTS Магазин 15\XlsCsv\Templates\Upload»

Файл CSV:

```
КодФормы;АлкоКод;Начало;Конец
```

В шаблоне, через точку с запятой (без пробелов), перечисляются поля Формы А, которые будем выгружать.

Для Excel формат обмена имеет аналогичную структуру.

Колонки для шаблона «ФормыА»:

Название колонки	Описание
КодФормы	Код формы в системе ЕГАИС.
АлкоКод	Строка с кодом алкогольной продукции в ЕГАИС.
Начало	Начало диапазона серийных номеров.
Конец	Конец диапазона серийных номеров.

Примеры

Пример корректного входного файла «ФормыА.csv»:

```
КодФормы;АлкоКод;Начало;Конец

123456;00313180000001257558;102033827000;102033828000

654321;00313180000001257580;100684707541;100684707762
```

Выгрузка документов

> Все созданные Вами документы для выгрузки необходимо класть в папку «На терминал» («...\XlsCsv\На терминал»).

Файл документа для выгрузки должен иметь имя, начинающееся с имени типа документа (например, Сбор начальных остатков), после которого идет небуквенный и нецифровой символ, например, «№» или «#», после которого можно указать номер или дату документа.

Примеры:

- Возврат № 003
- Поступление # 26.01.2016
- Инвентаризация # 00078
- Перемещение № 0053 от 10.02.2016

> Шаблон для выгрузки документов на ТСД выгружать не обязательно. Выгружаем готовый документ, который одновременно может быть шаблоном, тогда обязательно каждая колонка должна иметь ячейку заголовок, содержащую название колонки.

Примеры

Пример корректного входного файла Excel документа «Поступление №00001»:

> Файлы Excel для выгрузки документа должны иметь один лист, который содержит выгружаемые данные.

Описание строк задается в виде таблицы, содержащей колонки. Каждая колонка имеет ячейку заголовок, содержащую название колонки и данные.

Этап второй

Загрузка документов

Файл документа для загрузки будет иметь имя, начинающееся с имени типа документа (Сбор начальных остатков), после которого идет небуквенный и нецифровой символ, например, «№» или «#».

> Итоговые загруженные файлы документа создаются автоматически по шаблону и попадают в папку «... \XlsCsv\С терминала».

Шаблон документа для загрузки с ТСД описывает какие данные и в каком порядке попадут в итоговый файл excel или csv.

Шаблон для загрузки документов с ТСД

Путь к шаблонам «...\XlsCsv\Templates\Download».

Файл CSV:

```
#{Document};;;;;;
Имя;;;;;;
#{Document.CombinedItems};;;;;;
ШК;Артикул;Наименование;План;Факт;Ячейка;SSCC
```

В шаблоне, через точку с запятой (без пробелов), перечисляются поля шапки и табличной части документа, которые будут загружаться в итоговый файл.

> Поля для алкогольной продукции добавляем в шаблон только при работе с алкоголем.

Файл Excel:

Шаблоны можно изменять или вообще создать свой, с колонками, которые необходимы Вам для загрузки.

Колонки для шаблона загрузки документа:

Название колонки	Описание
#{Document}	В CSV файле указание, что далее идут поля шапки документа для загрузки.
#{Document.CombinedItems} #{Document.CurrentItems} #{Document.DeclaredItems}	В CSV файле указание, из какой табличной части документа загружаются итоговые данные.
Штрихкод	Штрихкод товара (обычно EAN13).
Артикул	Содержит артикул товара.
Название	Наименование товара.
Заявлено	Заявленное количество.
Принято	Фактическое количество, то которое отсканировали.
Серия	Серия.
Цена	Цена.
Характеристика	Характеристика товара (если ведется учет с характеристиками).
Название алко	Наименование товара из ЕГАИС, если уже известно для данного товара, либо если было получено из CheckMark.
Объем	Ёмкость тары в литрах из ЕГАИС, если уже известна для данного товара, либо если была получена из CheckMark.
Крепость	Процентное содержание спирта из ЕГАИС, если уже известно для данного товара, либо если был получен из CheckMark.
Производитель	Наименование производителя продукции в ЕГАИС, если уже известно для данного товара, либо если было получено из CheckMark.
ИНН	ИНН производителя, если уже известен для данного товара, либо если был получен из CheckMark.
КПП	КПП производителя, если уже известен для данного товара, либо если был получен из CheckMark.
Алко код	Код алкогольной продукции в ЕГАИС, полученный

PDF417
Из отсканированных марок.

Строка с PDF 417.

Серийный номер
Серийный номер бутылки, если сканировался.

Дата розлива
Дата розлива позиции, если вводилась, иначе null.

Наличие в формах А
Признак того, что данная марка была найдена в выгруженных формах А.

Код формы А
Код формы А, где была найдена марка.

Примеры

Файл CSV загруженный с ТСД (например, Поступление №00001.csv)

```
Имя;;;;;

"Поступление №00001";"";"";"";"";""

ШК;Артикул;Название;Заявлено;Принято;Серия

"4602827790520";"79052";"STEP Пазл 1000 Английский коттедж";"";"1";""

"4602827790759";"79075";"STEP Пазл 1000 Красная площадь";"";"1";""

"4602827790780";"79078";"STEP Пазл 1000 Гаваи";"";"1";""
```

Незаполненные поля загружаются в файл в виде разделителя «««;».

Файл Excel загруженный с ТСД (например, Поступление №00001. xls)

магазин, интеграция

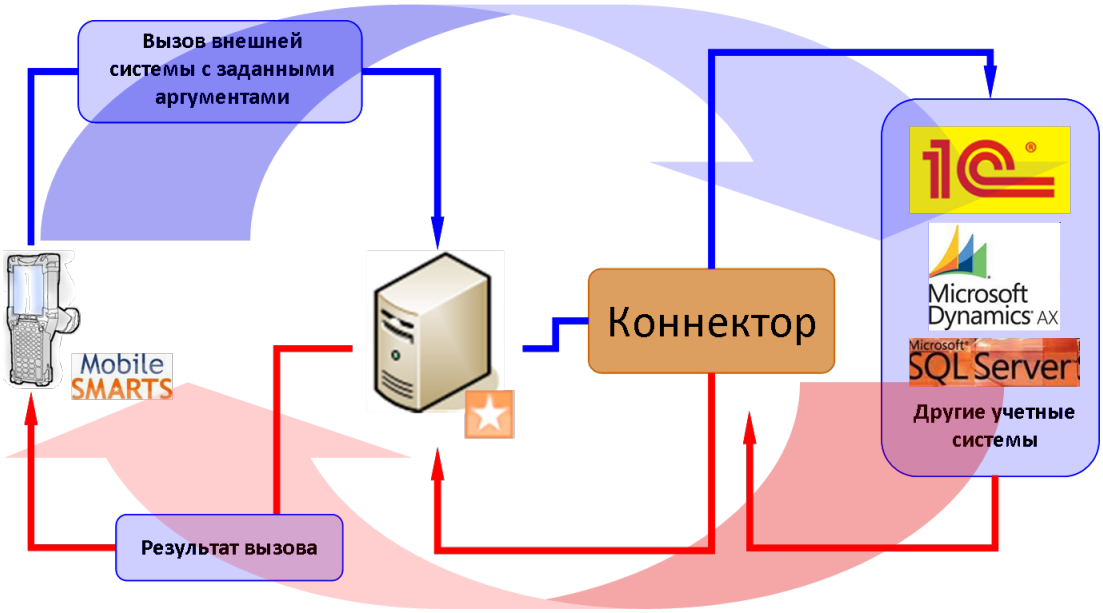
Не нашли что искали?

Задать вопрос в техническую поддержку

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

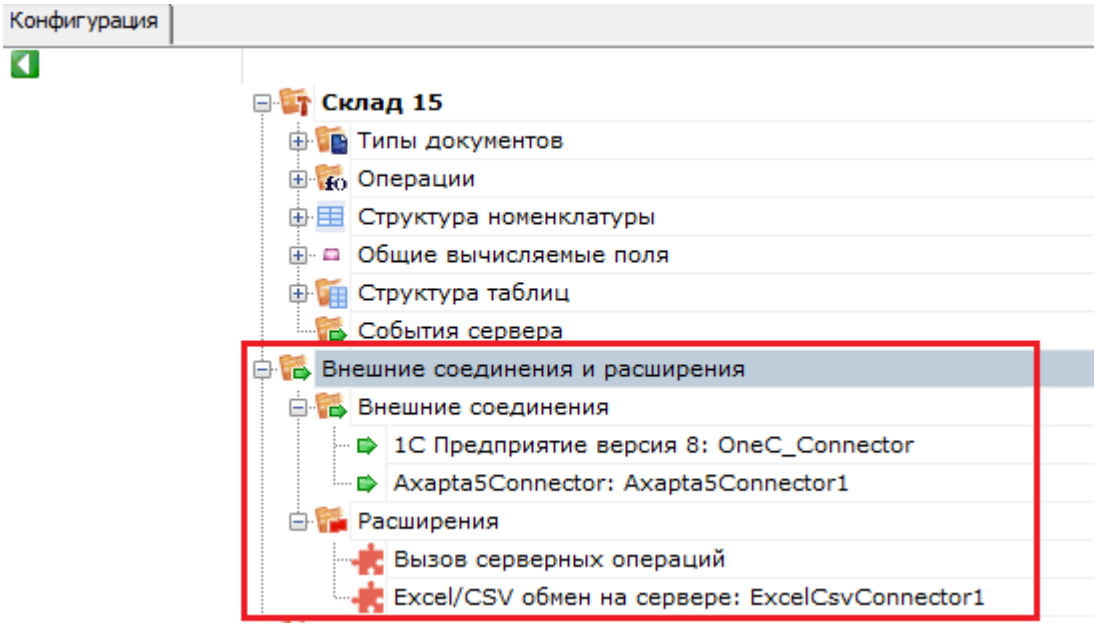
Не нашли что искали?

Задать вопрос в техническую поддержку

Регистрация и запуск коннекторов

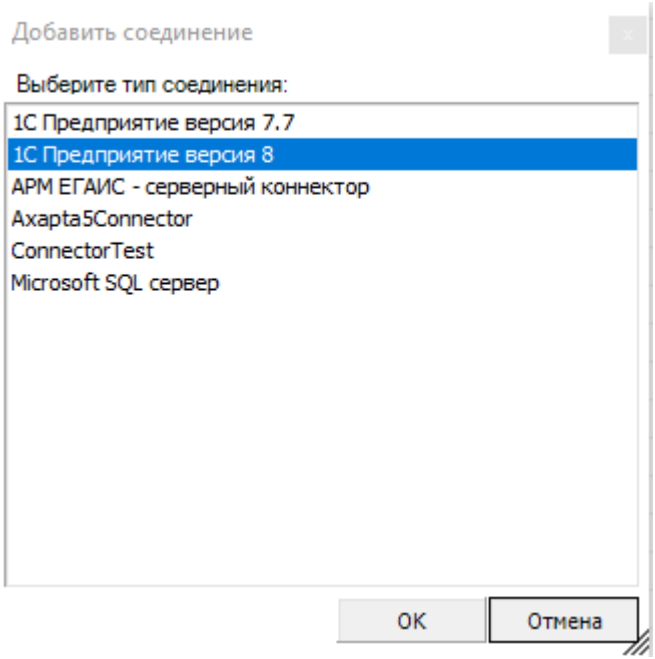
Последние изменения: 2024-03-26

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».

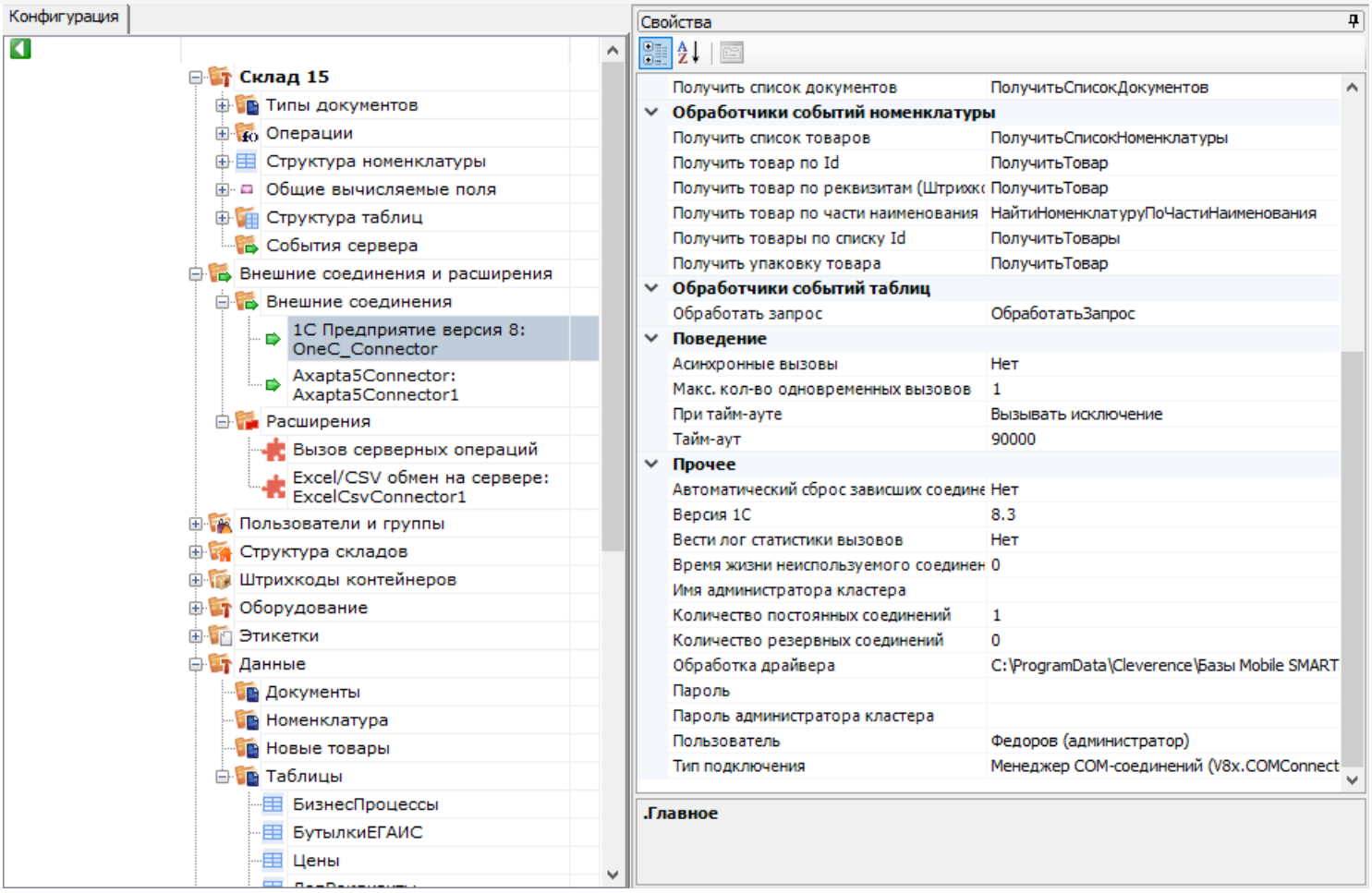


Регистрация коннектора

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:

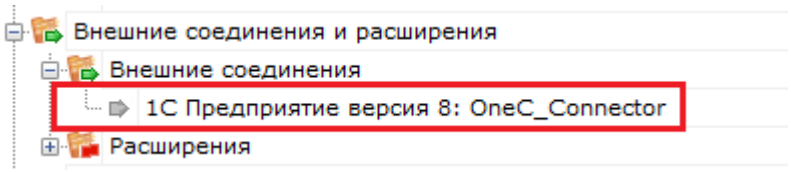


Настройка параметров коннектора выполняется через боковую панель свойств:

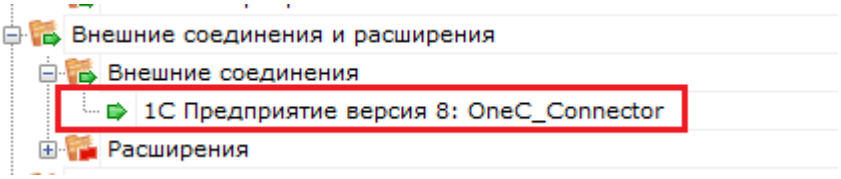


После добавления коннектора и настройки параметров нужно сохранить конфигурацию.

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

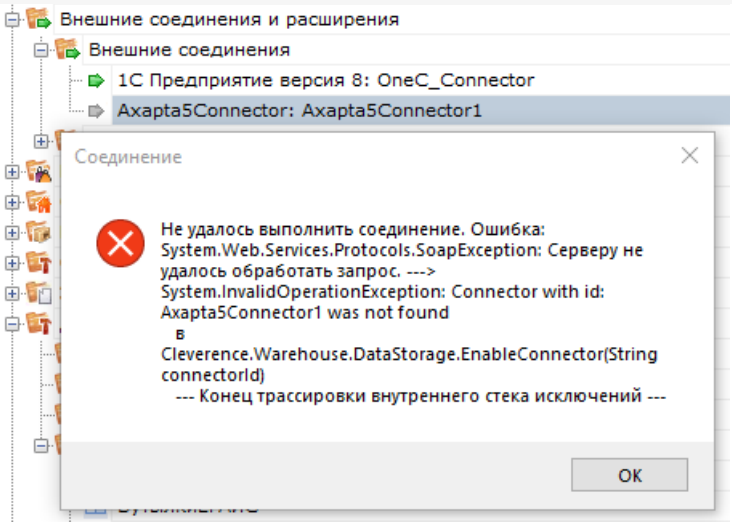


Для того, чтобы разрешить вызовы, нажмите правой кнопкой мыши на узле дерева, выберите в контекстном меню «Разрешить». Модуль должен перейти в разрешенное состояние (зеленая стрелка):

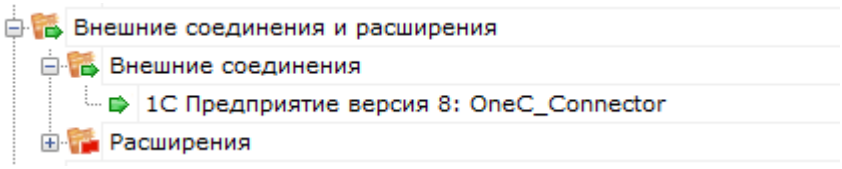


Для работы модуля под управлением сервера Mobile SMARTS и настройки параметров модуля через панель управления используются две версии библиотеки (файлы dll). Первая dll, предназначенная для сервера, размещается в <Папка базы Mobile SMARTS>\Server\DataService\bin\. Вторая dll, для панели управления - 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.



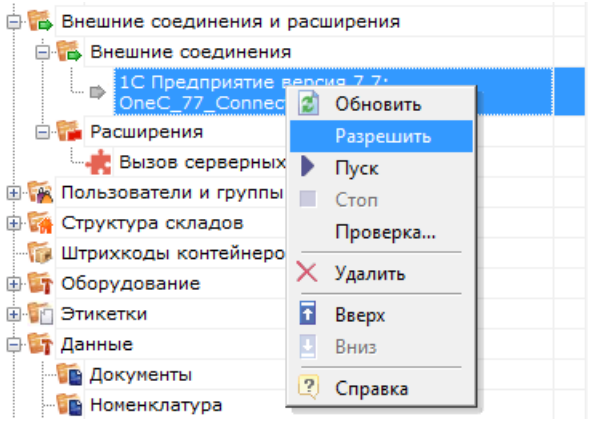
Если загрузить конфигурацию с сервера (используя кнопку «Обновить» в панели управления), отображаются данные о том, что добавленного модуля нет:

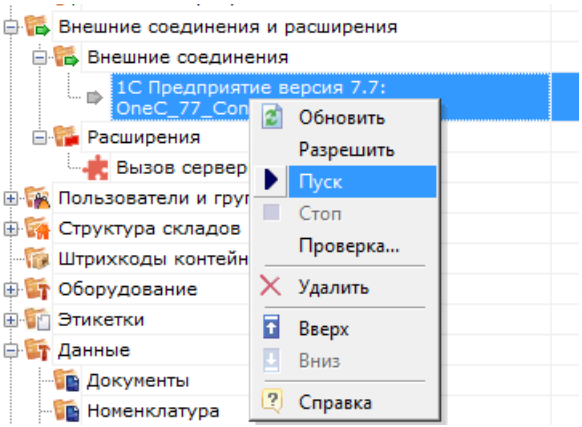


Для устранения ошибки необходимо переместить серверную версию dll модуля в <Папка базы Mobile SMARTS>\Server\DataService\bin\, перезапустить сервер базы данных Mobile SMARTS и заново добавить модуль через панель управления.

Запуск коннектора

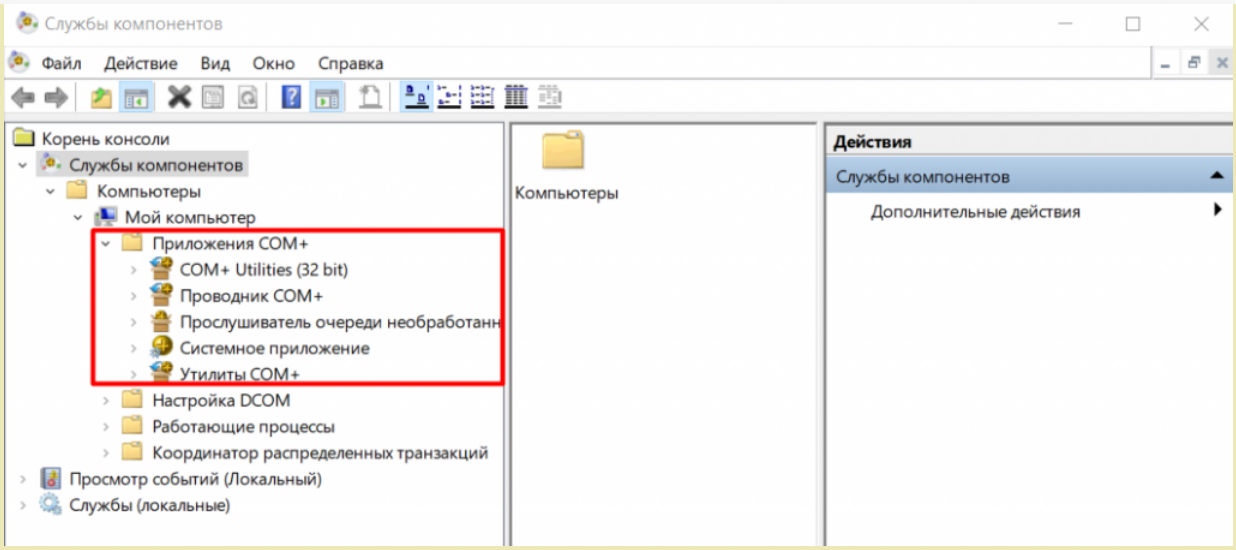
После регистрации коннектора следует проверить, выполняется ли подключение. Для этого следует разрешить доступ к коннектору («Разрешить»), и затем запустить его («Пуск»).





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?

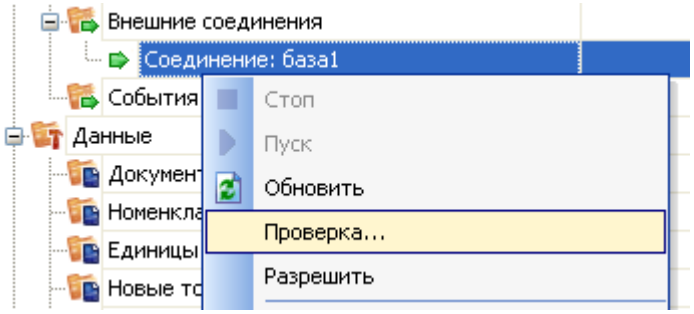
Задать вопрос в техническую поддержку

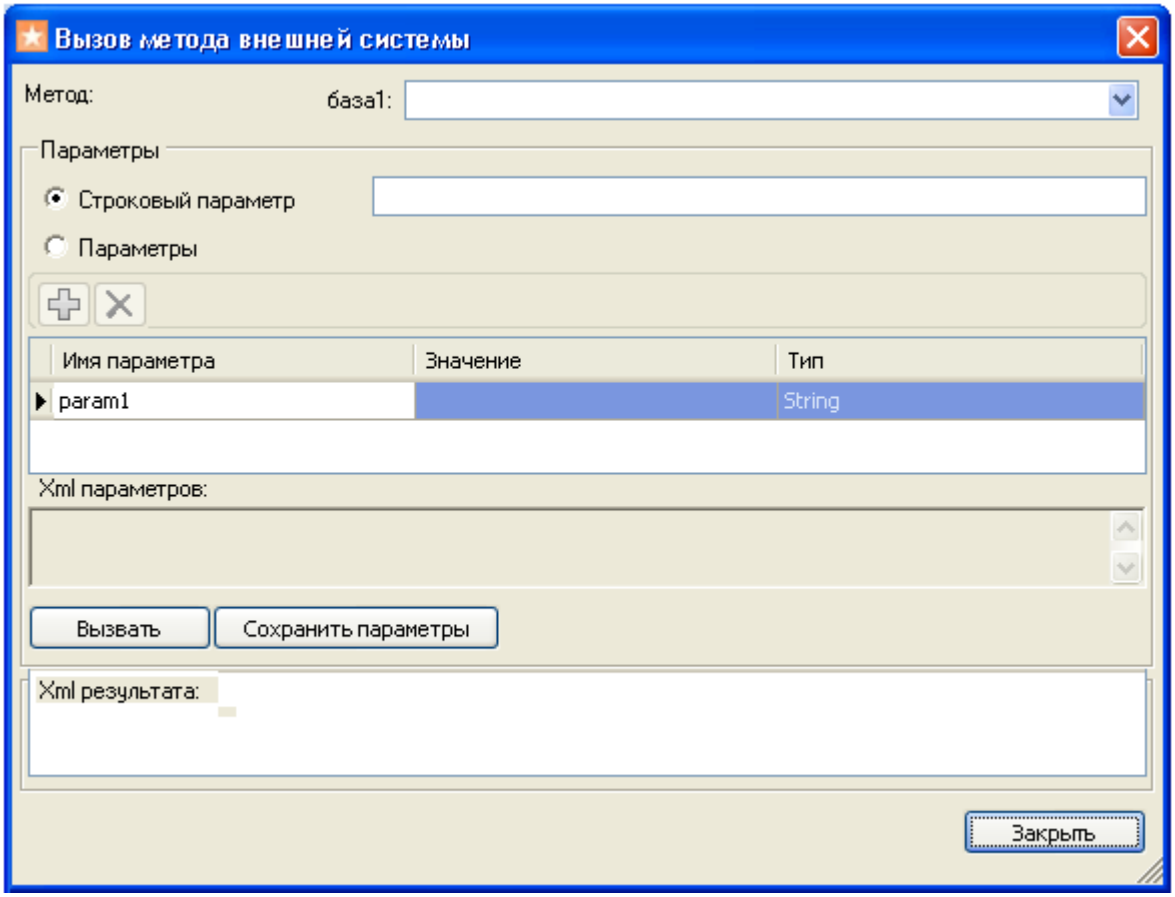
Отладка вызовов к внешней системе через коннекторы

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

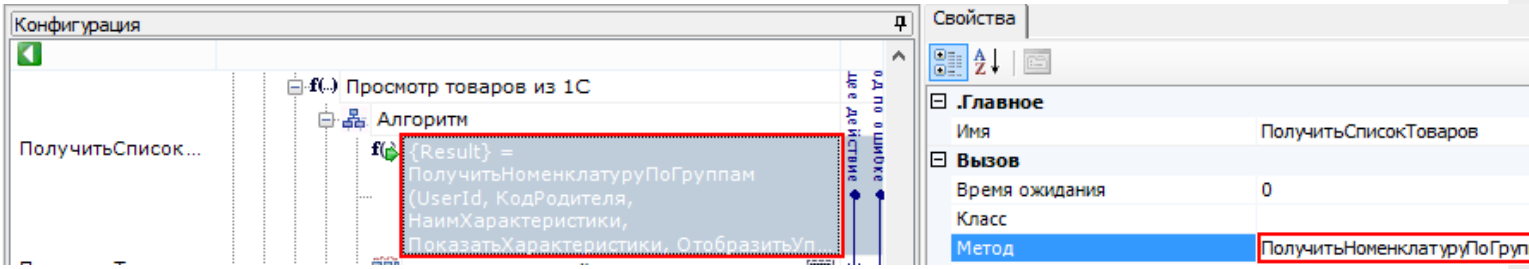
XML с сериализованным объектом Cleverence.Warehosue.InvokeArgs; это выглядит примерно как «<?xml version="1.0" encoding="utf-8" ?><InvokeArgs xmlns:clr="http://schemas...».

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.

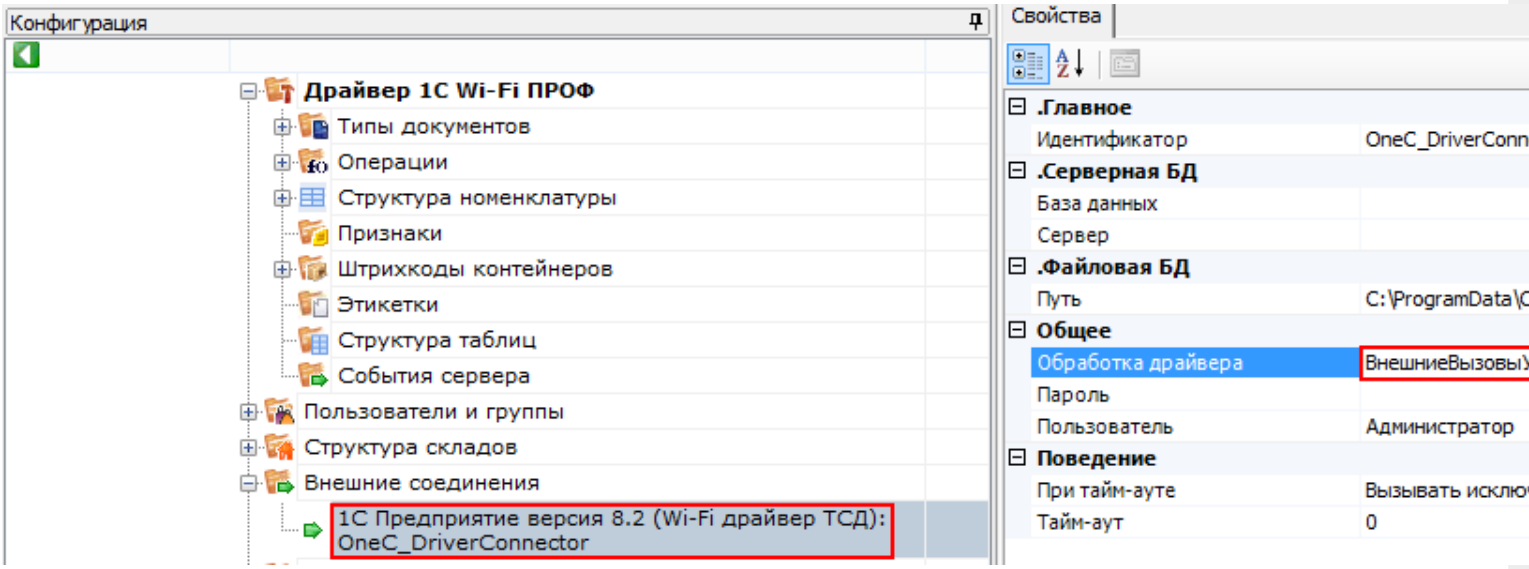
Рассмотрим подключение на основе операции «Просмотр товаров из 1С» из стандартной поставки Wi-Fi ПРОФ драйвера. Разберем, что вызывается и отладим процедуру, которая вызывается.

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



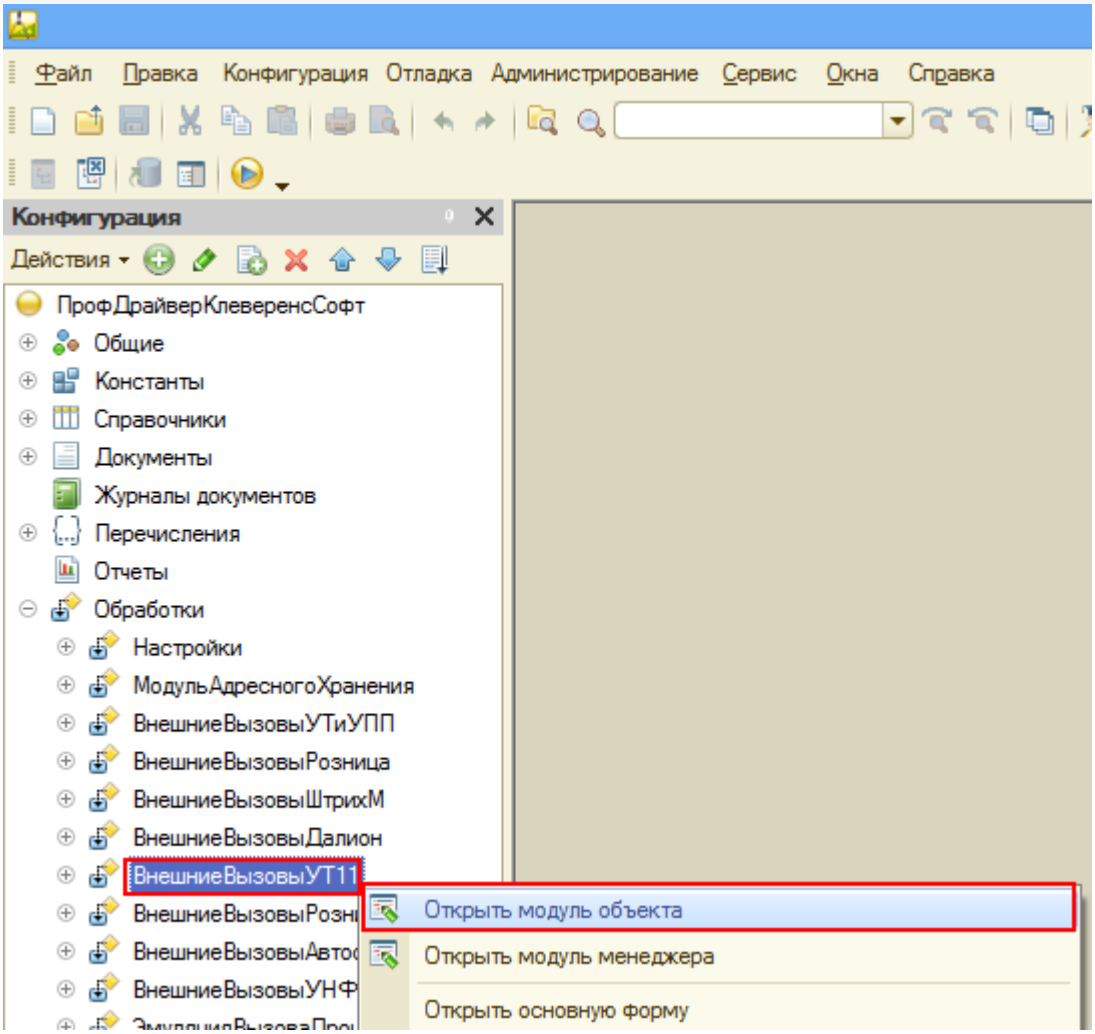
Для данной операции метод, который вызывается из 1С называется «ПолучитьНоменклатуруПоГруппам».

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

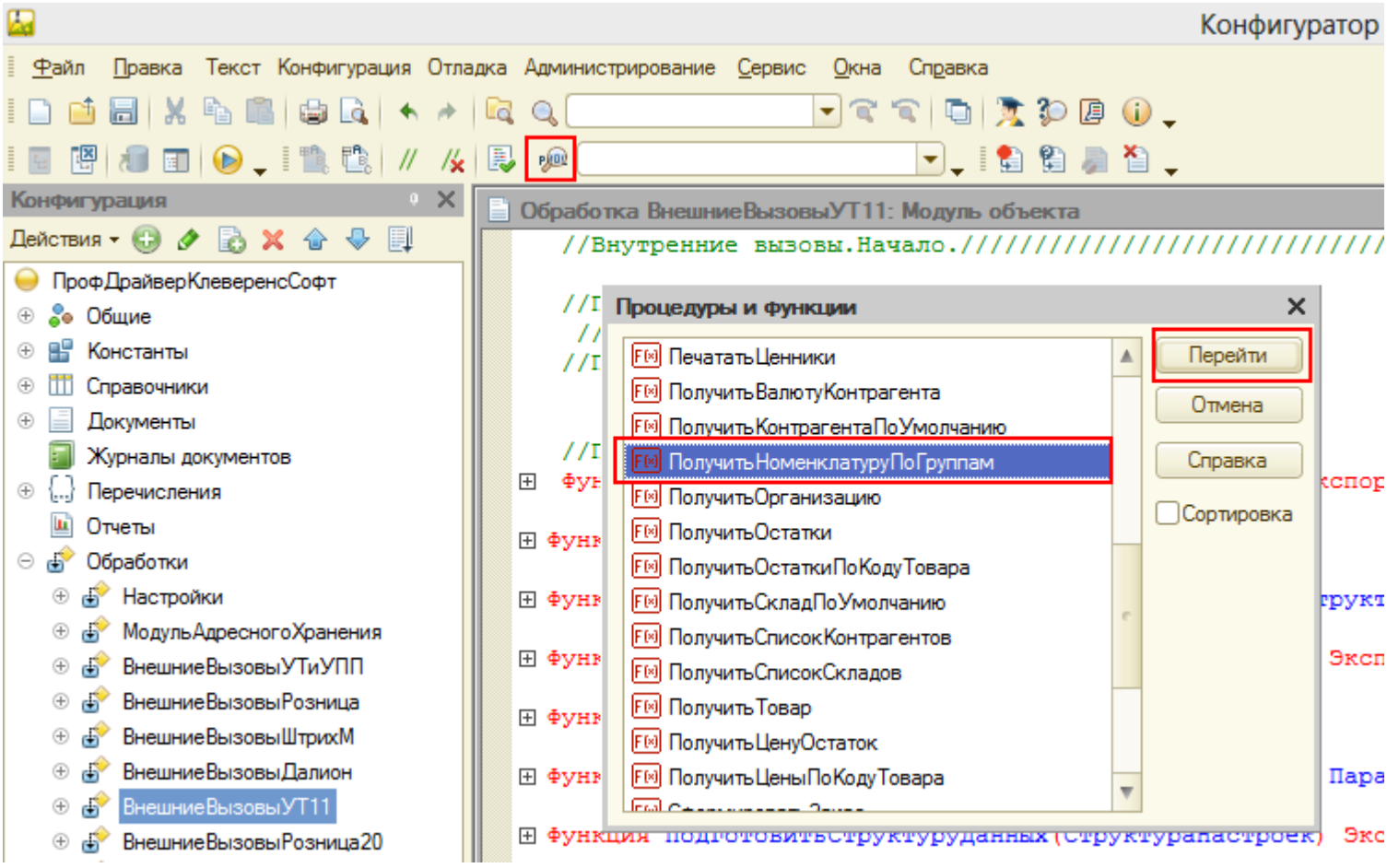


В данном примере подключена обработка «ВнешниеВызовыУТ11».

Теперь найдем данную процедуру в модуле обработки.



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

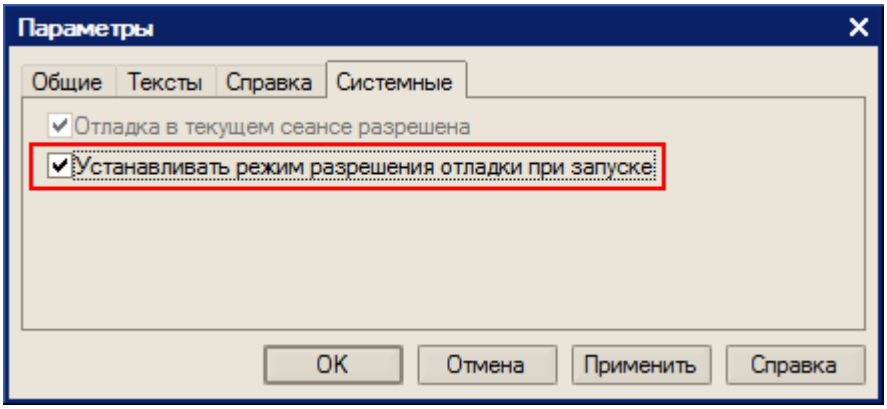


Обработка ВнешниеВызовыУТ11: Модуль объекта

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

    //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\».

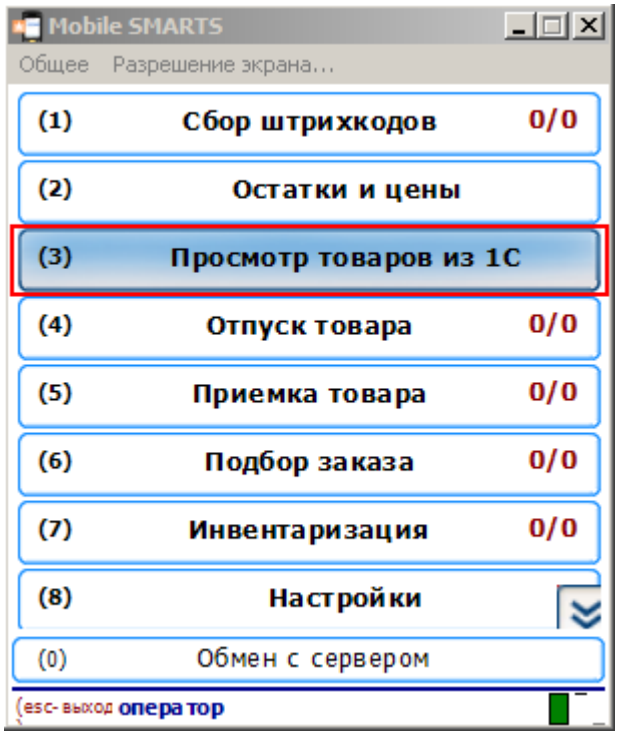
В файле прописано, что отладка разрешена и по умолчанию включается на локальной машине.

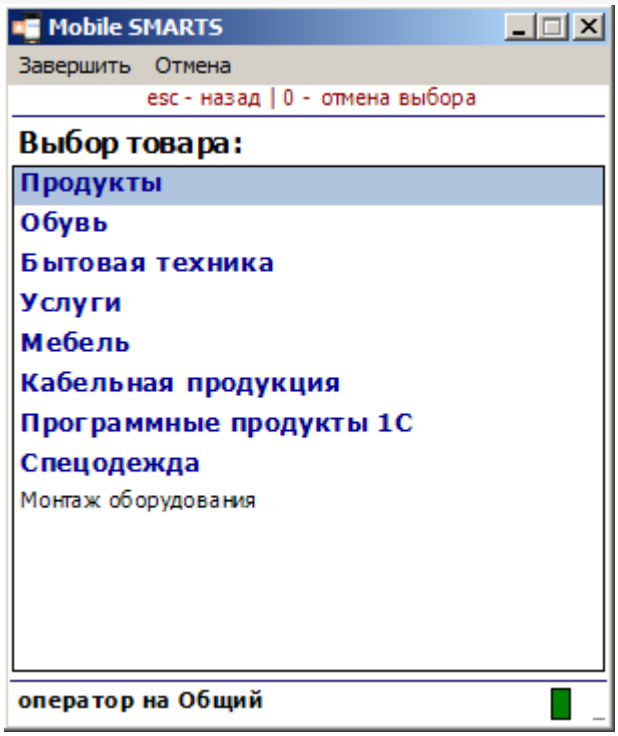
```
Файл  Правка  Вид  Кодировка  Справка
<config xmlns="http://v8.2c.ru/v8/comcntrcfg">
    <debugconfig debug="true"
    debuggerURL="tcp://localhost:1560"/>
</config>
```

Дальше поставим «точку останова».

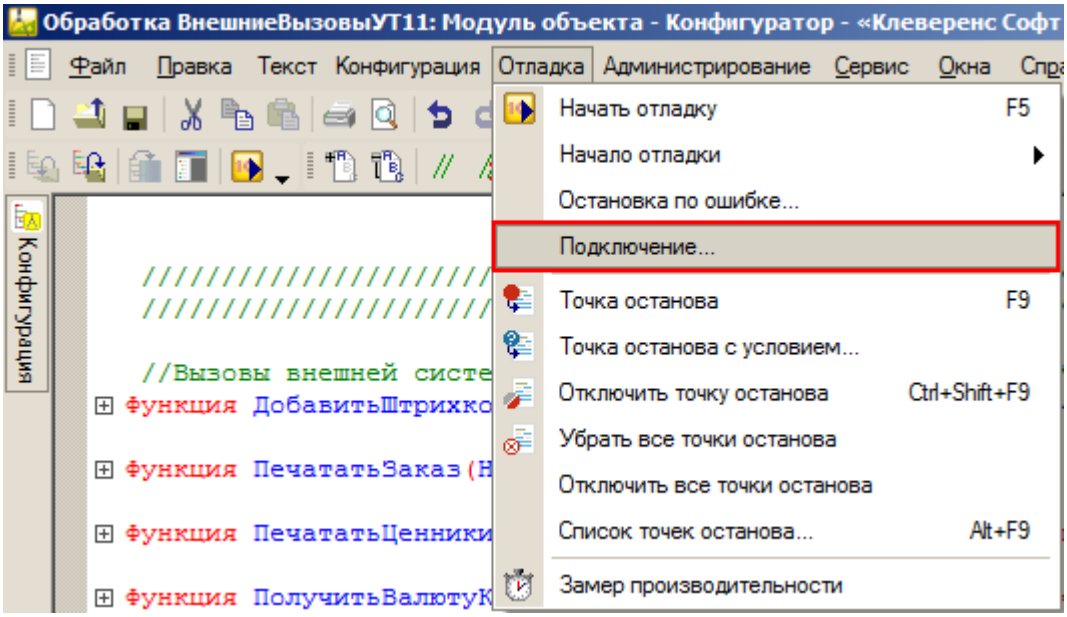
```
Функция ПолучитьНоменклатуруПоГруппам(UserId, КодГруппы, НаимХарактеристики, ПоказатьХарактеристики,Ото
    Возврат УниверсальныеМеханизмы.ПолучитьНоменклатуруПоГруппам(UserId, КодГруппы, НаимХарактеристики,
КонецФункции
```

Запустим вызов процедуры, выбрав операцию.

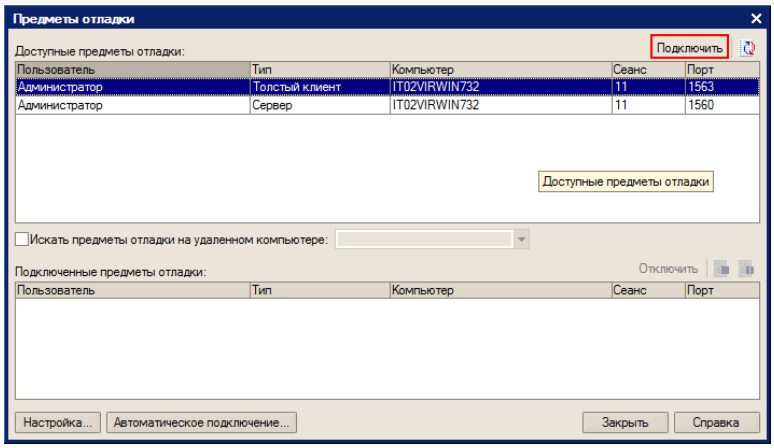


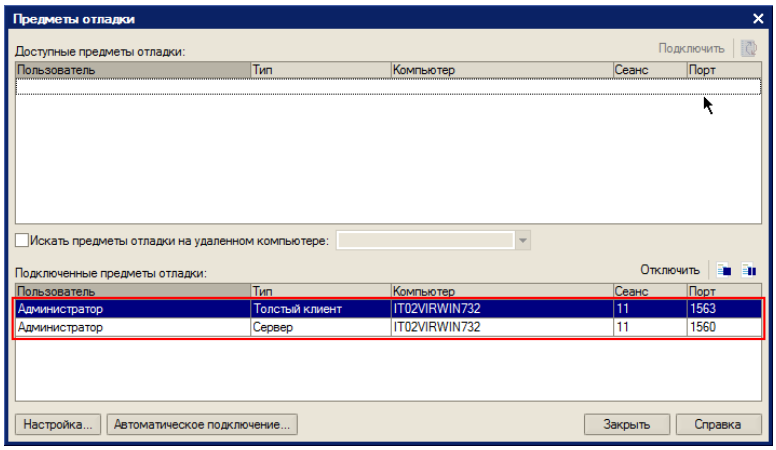


Создалось подключение сервера Mobile SMARTS к базе драйвера и теперь необходимо подключиться к той сессии, которая создалась. Выбираем Отладка -> Подключение.

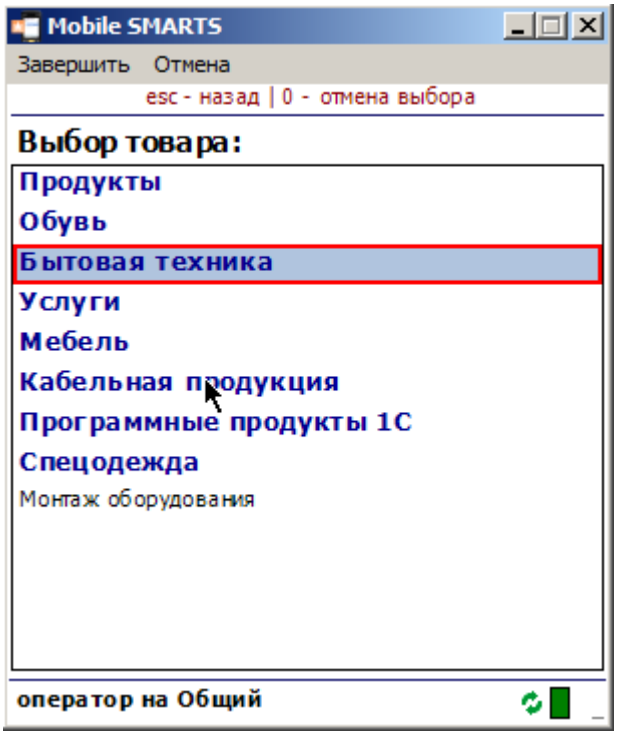


Производим подключение.





Вызовем нашу процедуру еще раз.



Система остановилась, на нашей «точке останова».

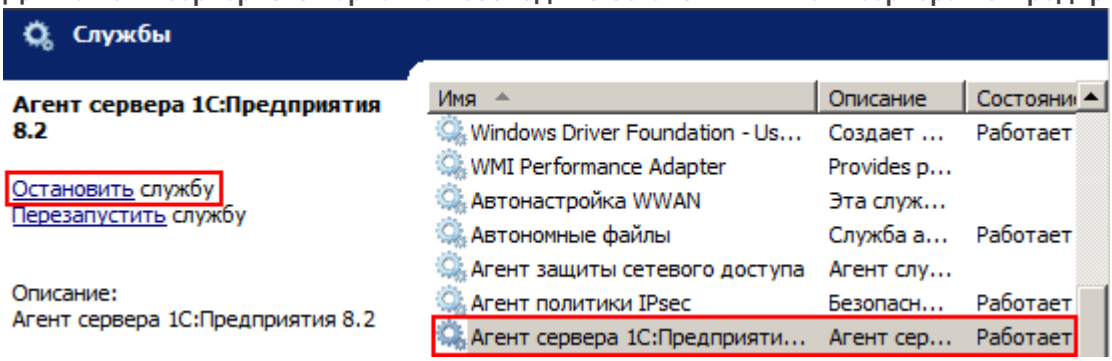
```
Функция ПолучитьНоменклатуруПоГруппам(UserId, КодГруппы, НаимХарактеристики, ПоказатьХарактеристики,Ото
    Возврат УниверсальныеМеханизмы.ПолучитьНоменклатуруПоГруппам(UserId, КодГруппы, НаимХарактеристики,
КонецФункции
```

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

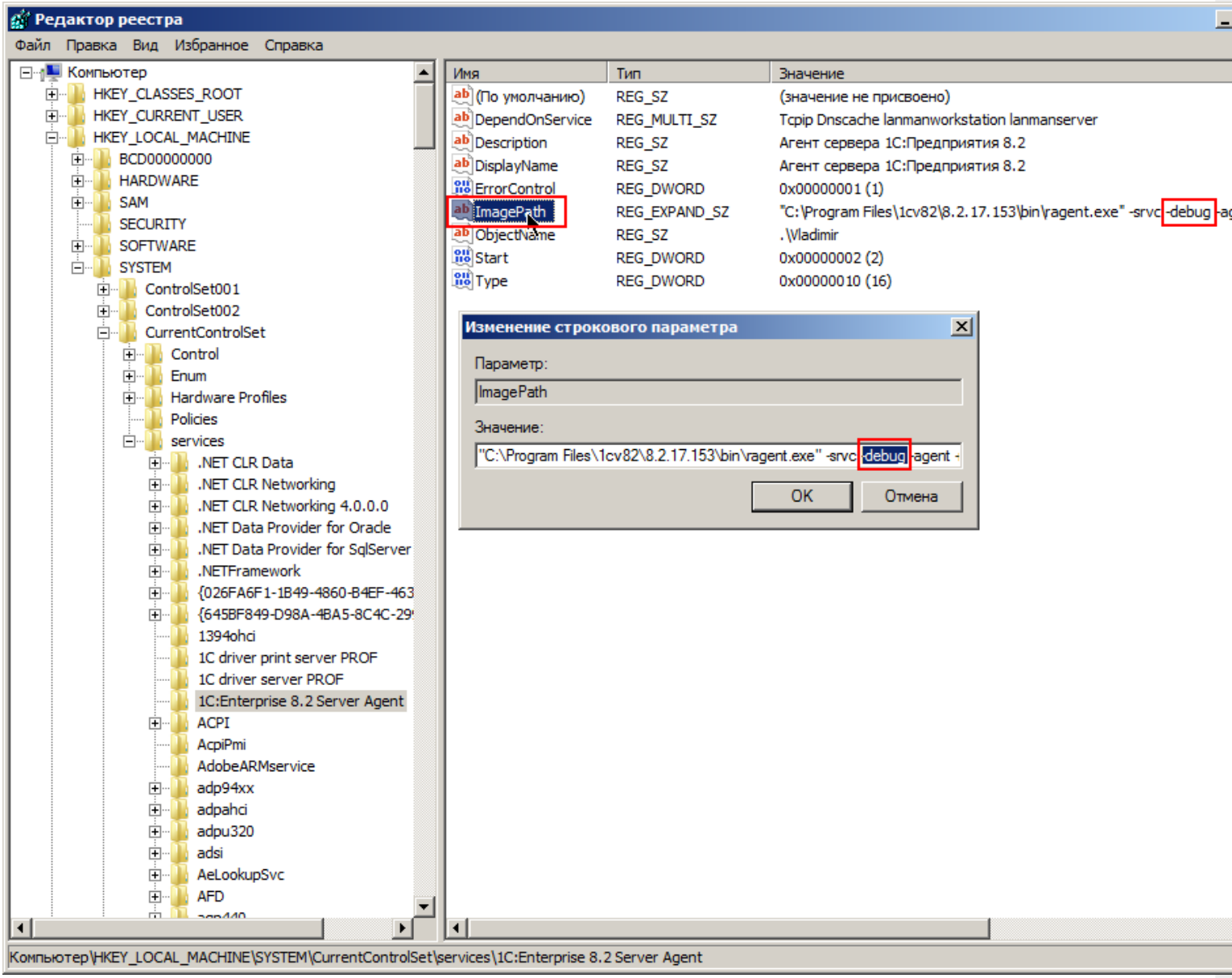
Клиент-серверный вариант работы

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

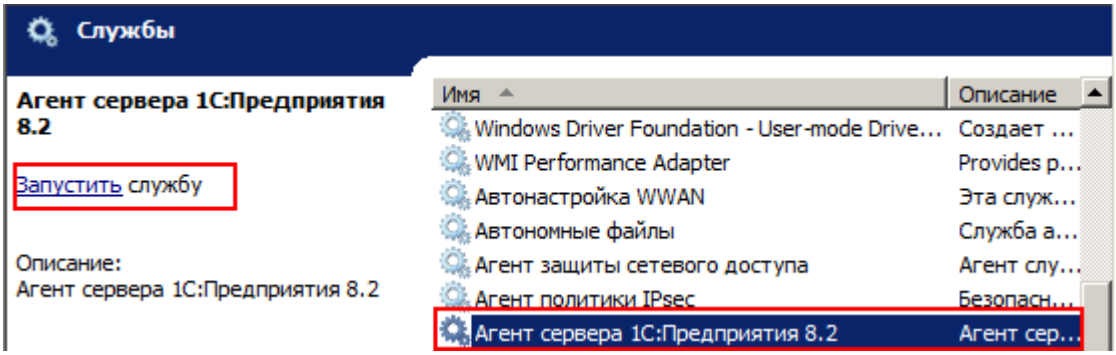
Для клиент-серверного варианта необходимо остановить «Агент сервера 1С:Предприятия».



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



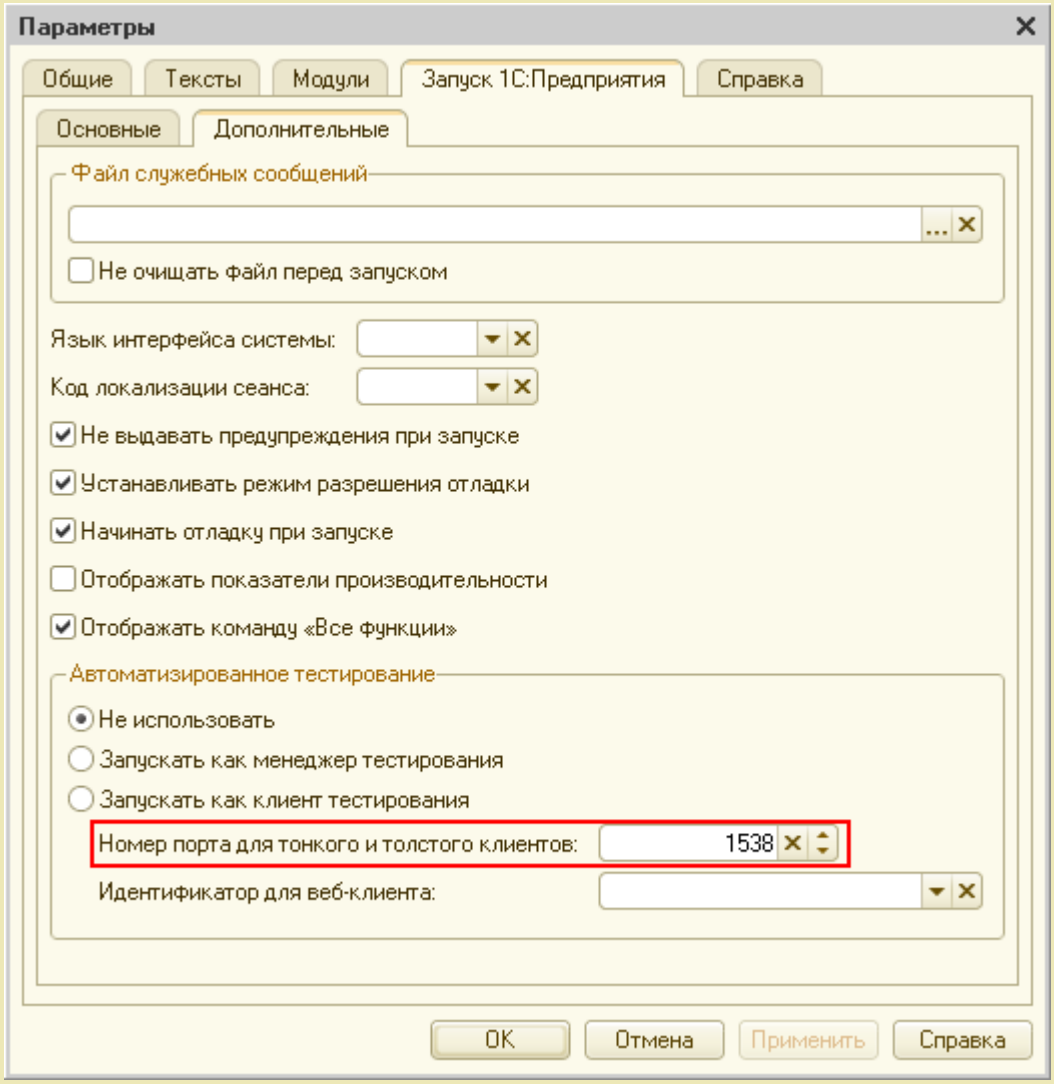
Запускаем службу.



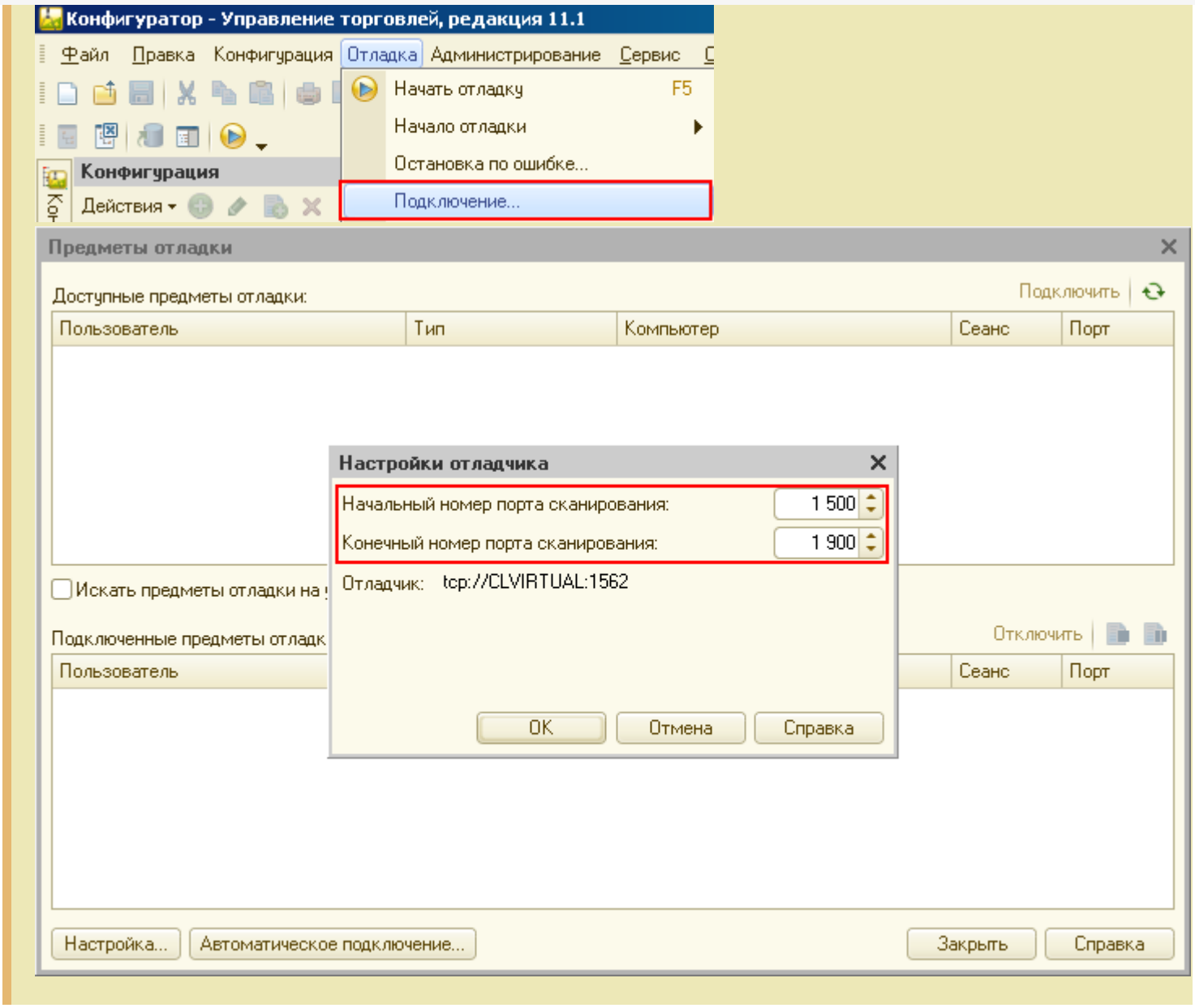
Запускаем конфигуратор базы драйвера.

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].

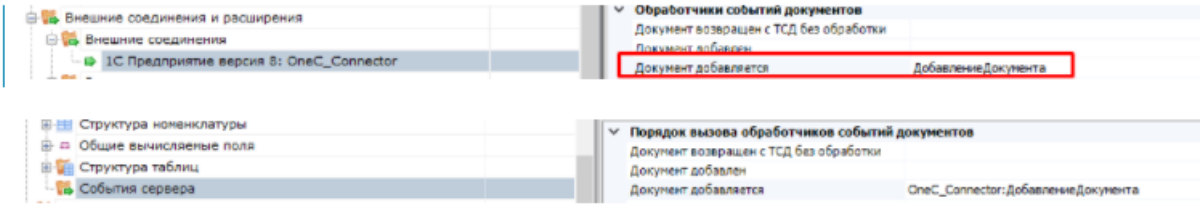
В случае обычных коннекторов, если обработчик события вернул null, добавление документа на сервер отменяется.

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).

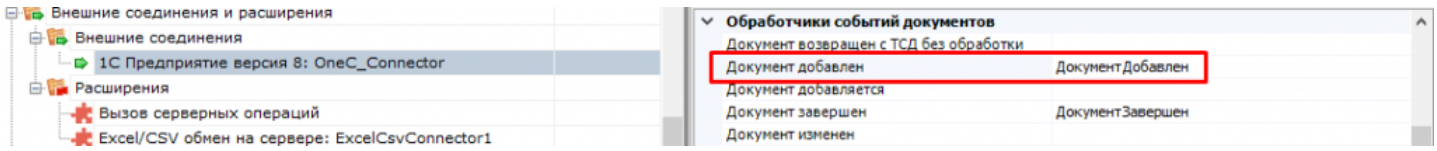
Обработчик может внести изменения в документ. После вызова обработчика документ сохраняется на сервере.

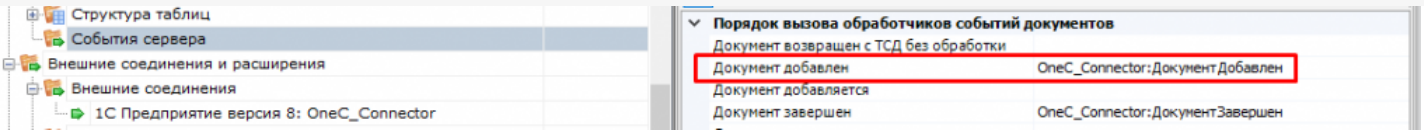
Описание в панели управления:

```
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].

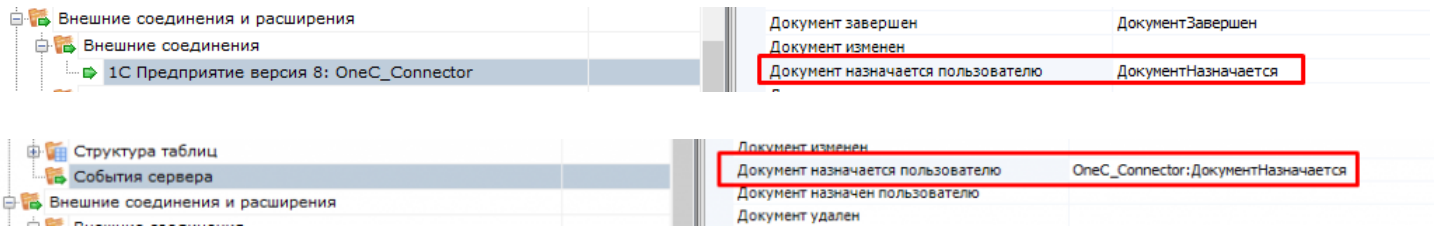
В случае обычных коннекторов, если обработчик события вернул null, назначение документа отменяется.

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).

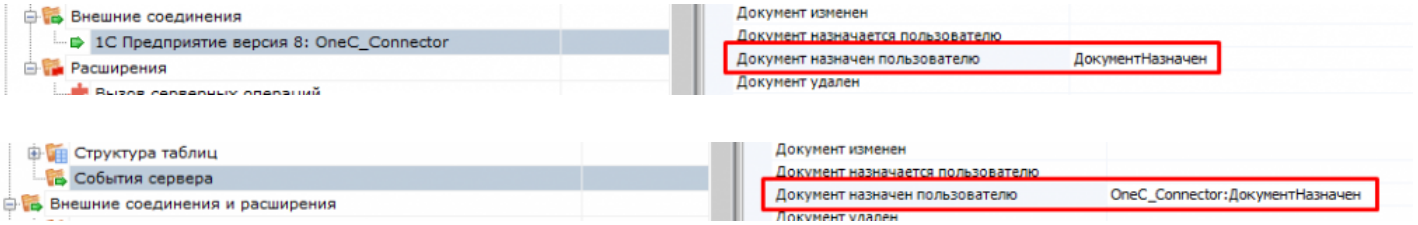
Обработчик может внести изменения в документ. После вызова обработчика документ сохраняется на сервере.

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».

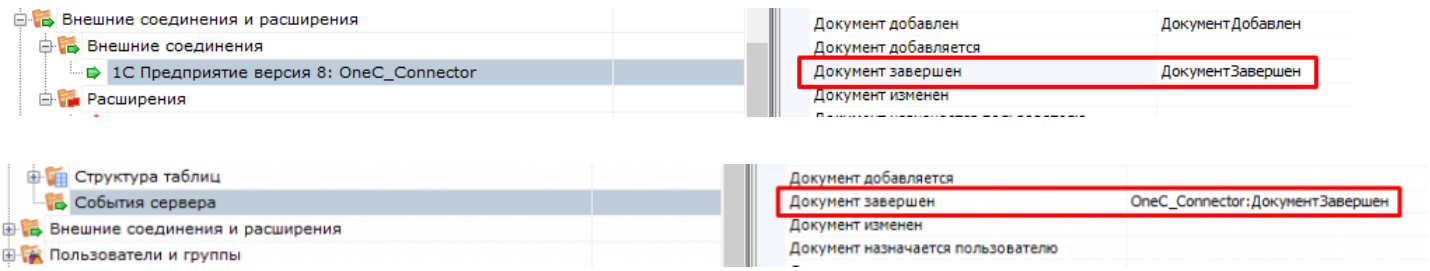
Событие может быть обработано модулем «Вызов серверных операций», в этом случае в серверную операцию передается сам документ (переменная сессии Document).

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».

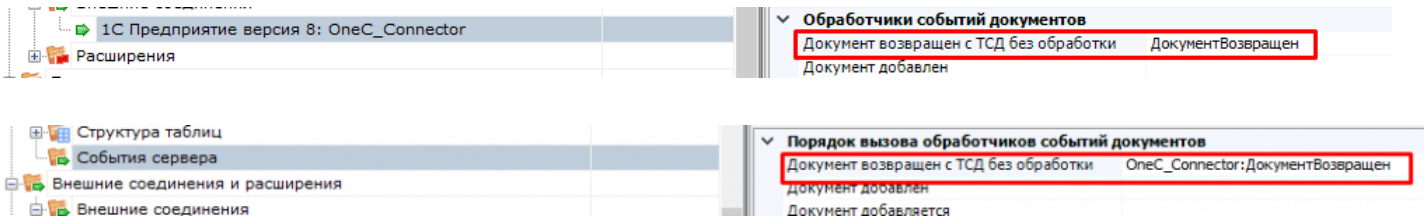
Событие может быть обработано модулем «Вызов серверных операций», в этом случае в серверную операцию передается сам документ (переменная сессии Document).

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».

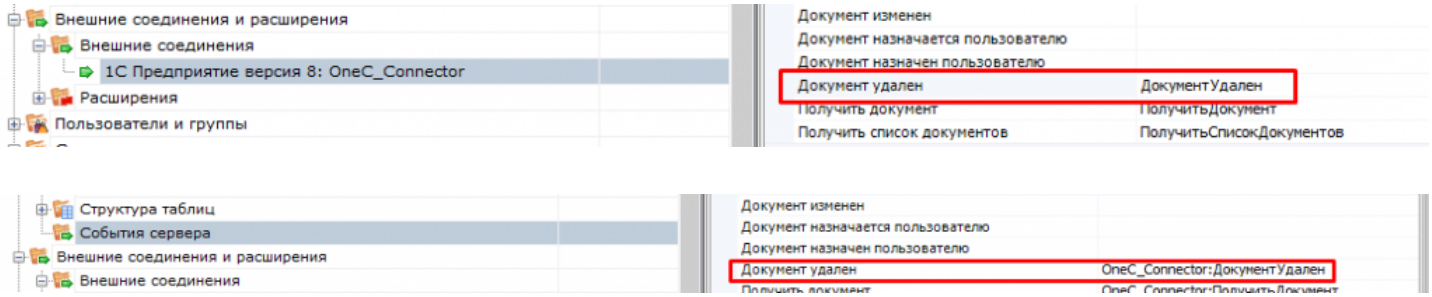
Событие может быть обработано модулем «Вызов серверных операций», в этом случае в серверную операцию передается сам документ (переменная сессии Document).

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.

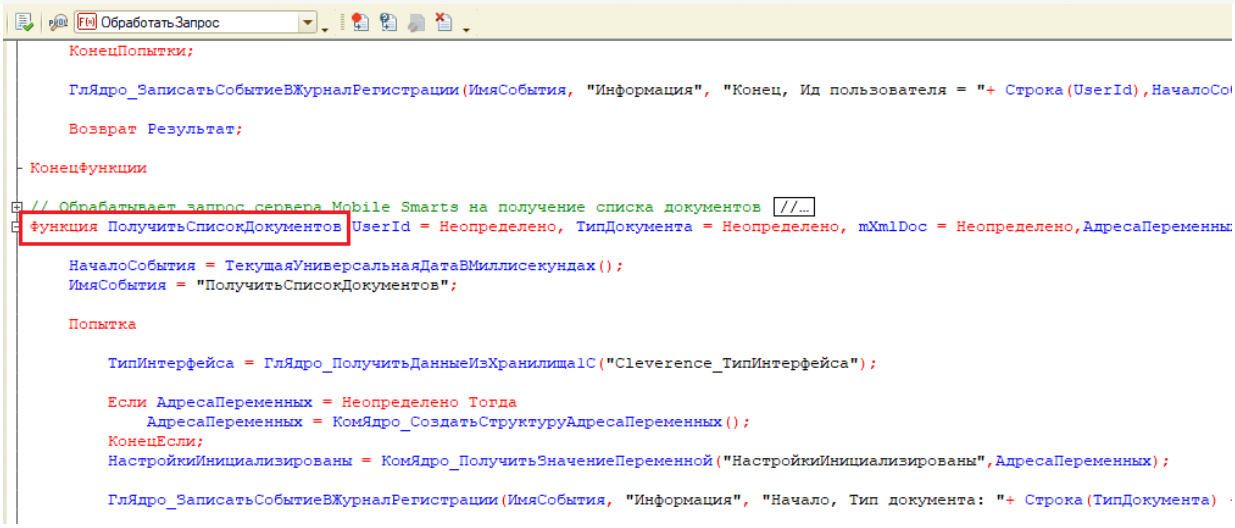
Например: OneC_Connector:ПолучитьСписокДокументов

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?

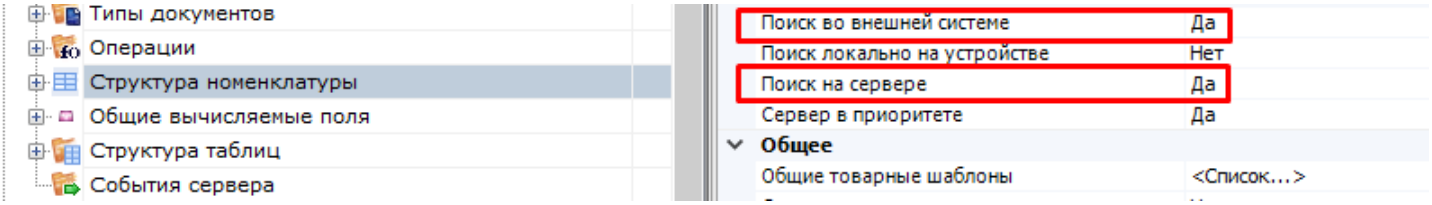
Задать вопрос в техническую поддержку

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:

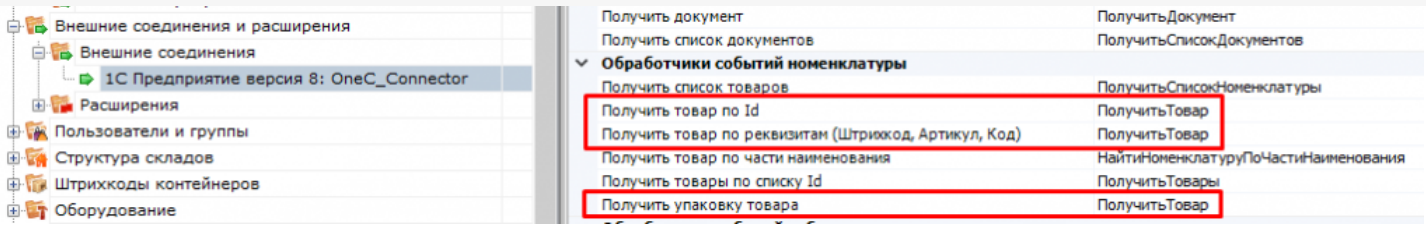
Требуется найти упаковку товара. Возвращаемое значение Cleverence.Warehouse.Packing в виде XML. Если товар не найден, необходимо вернуть null (неопределено).

Описание в панели управления:

```
C#

<ид. коннектора>:ПолучитьТовар
```

Ид. коннектора — задается в панели управления.

Например: OneC_Connector:ПолучитьТовар



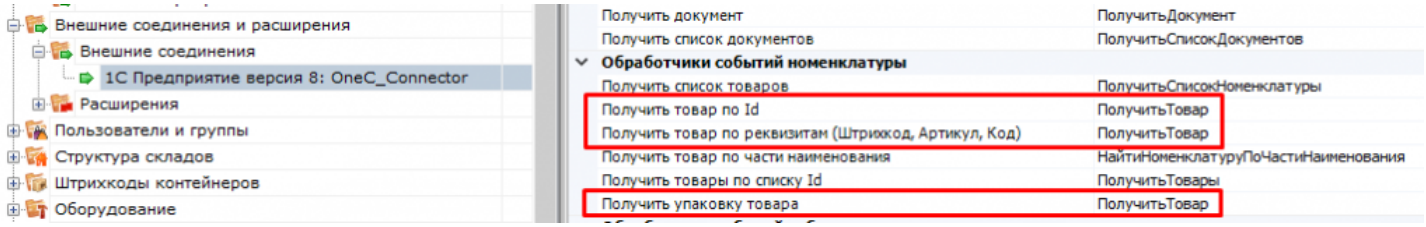
При использовании коннекторов, у которых имена обработчиков указываются в свойствах самого коннектора (производные от ConnectorTypical), есть возможность указать разные обработчики для разных режимов получения товара:

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
```

Не нашли что искали?

Задать вопрос в техническую поддержку

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»

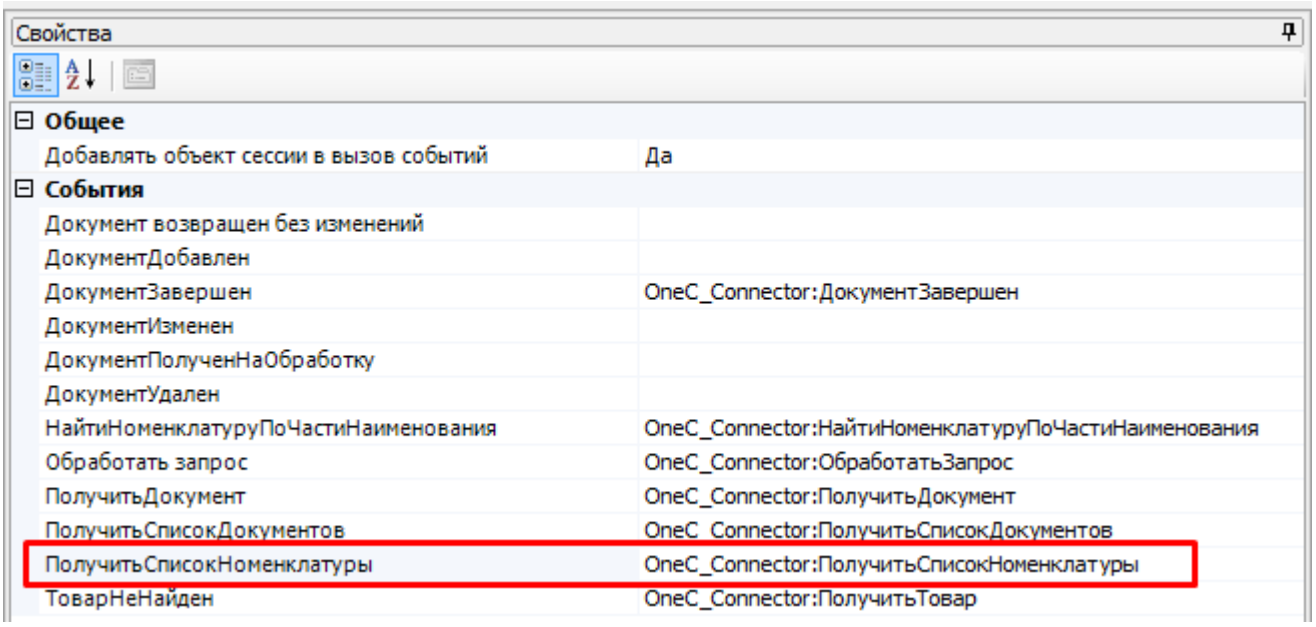
Последние изменения: 2024-03-26

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

1С

```
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?

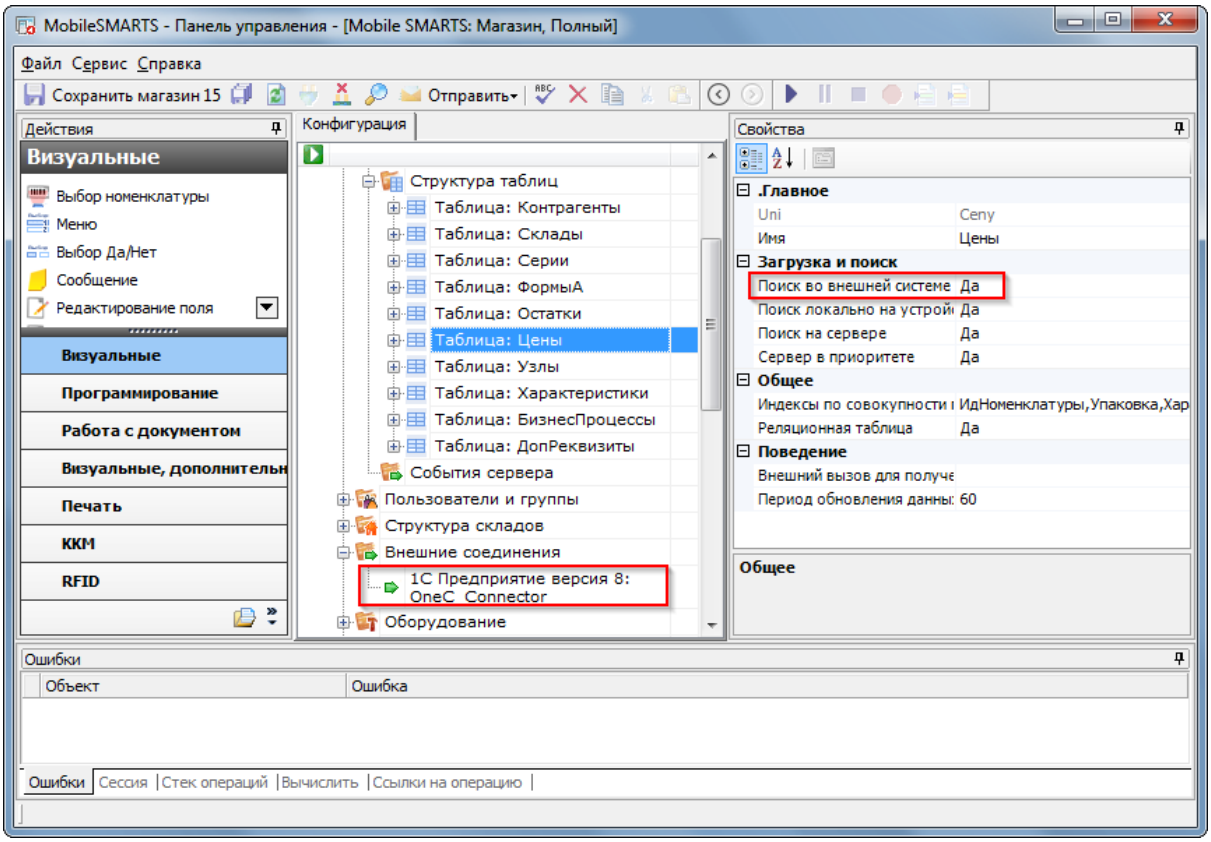
Задать вопрос в техническую поддержку

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.

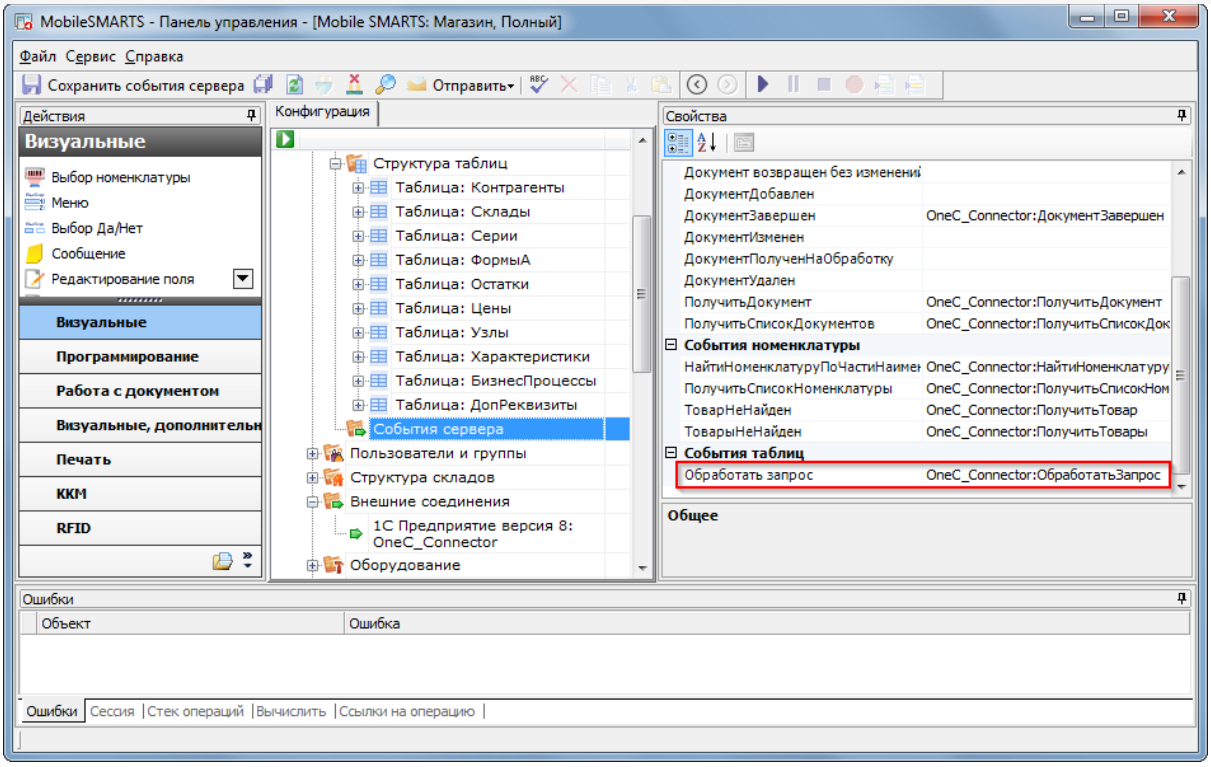
В случае 1С функция может возвращать таблицу значений. Наименования полей (как в случае строк Cleverence.Warehouse.Row, так и в случае колонок таблицы значений 1С) должны соответствовать наименованиям полей таблицы из конфигурации Mobile SMARTS.

Описание в панели управления:

```
1С

<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.

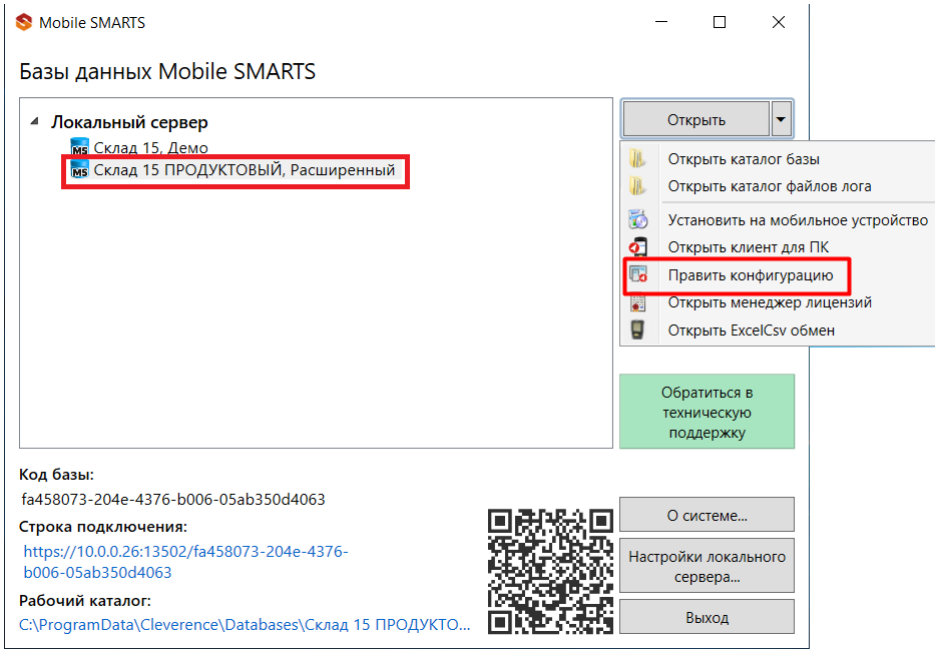
Начиная с версии 3.2 платформы Mobile SMARTS, в панели управления становится доступной функция ручной отправки основных событий для документов:

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

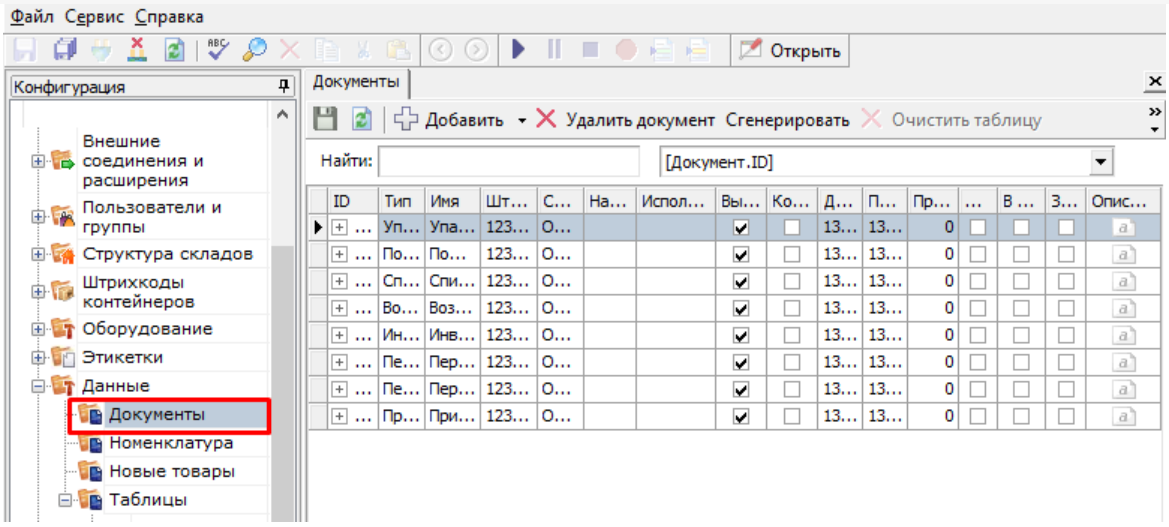
Это значит, что для вызова серверного события теперь не нужно производить какие-либо действия с документом на устройстве, достаточно просто нажать на одну кнопку в панели управления.

Для этого необходимо:

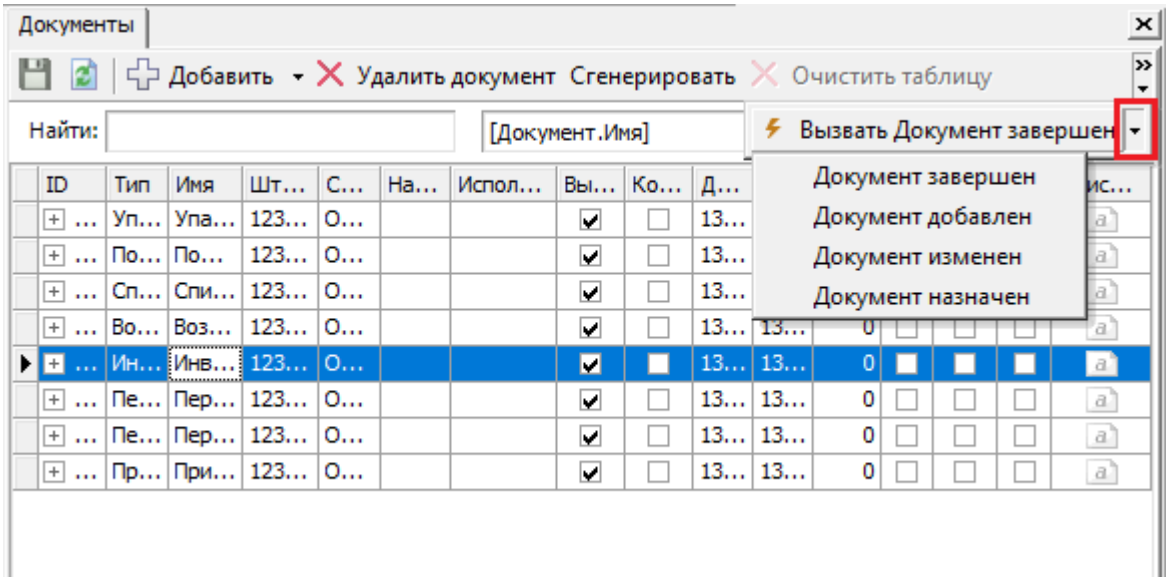
1. В менеджере баз Mobile SMARTS выбрать нужную базу и открыть для нее панель управления с помощью кнопки «Править конфигурацию».



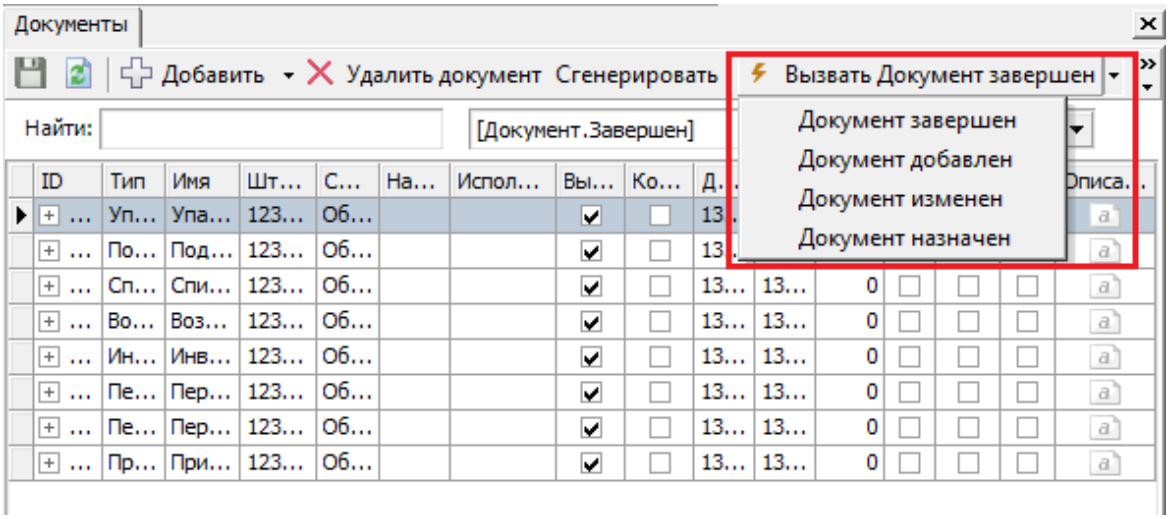
2. В панели управления открыть список документов данной базы.



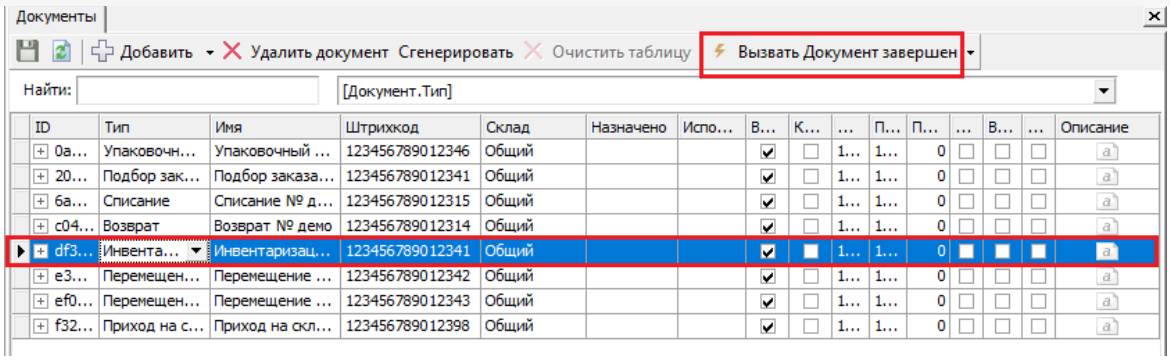
3. Добавить кнопку вызова события на панель инструментов с помощью кнопки . После этого выбрать нужное событие из выпадающего списка.



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.LicenseExternalSysten

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?

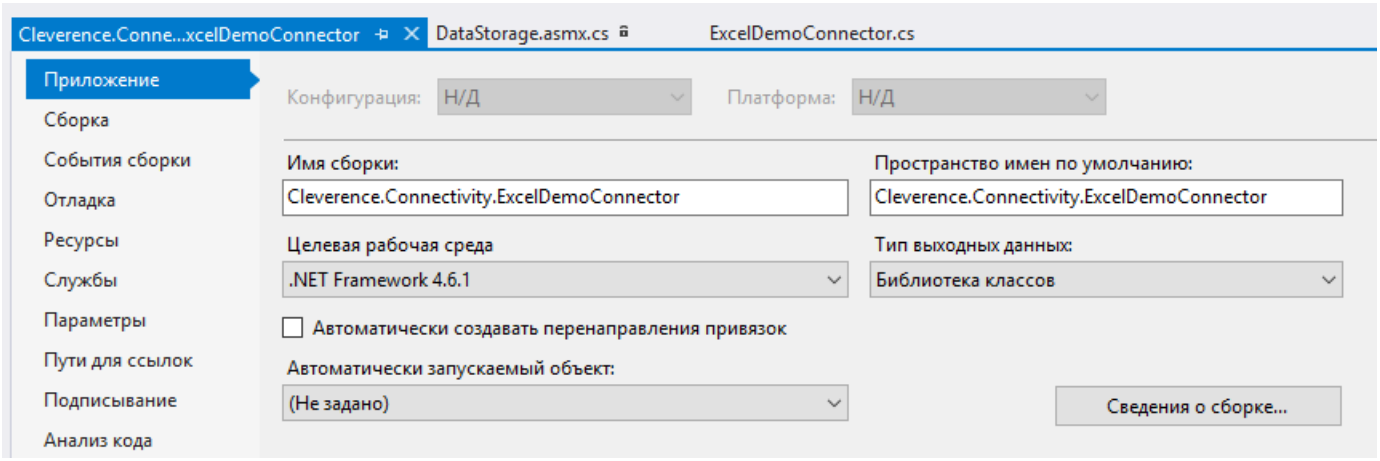
Задать вопрос в техническую поддержку

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):

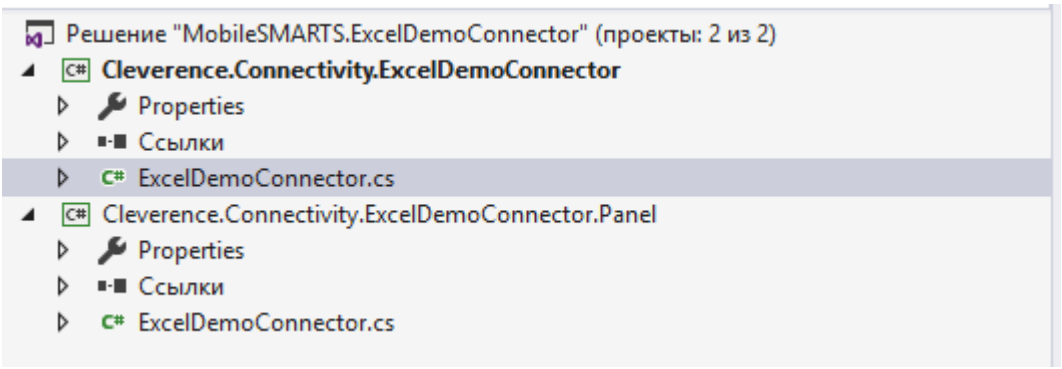
Cleverence.DataCollection.dll

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:

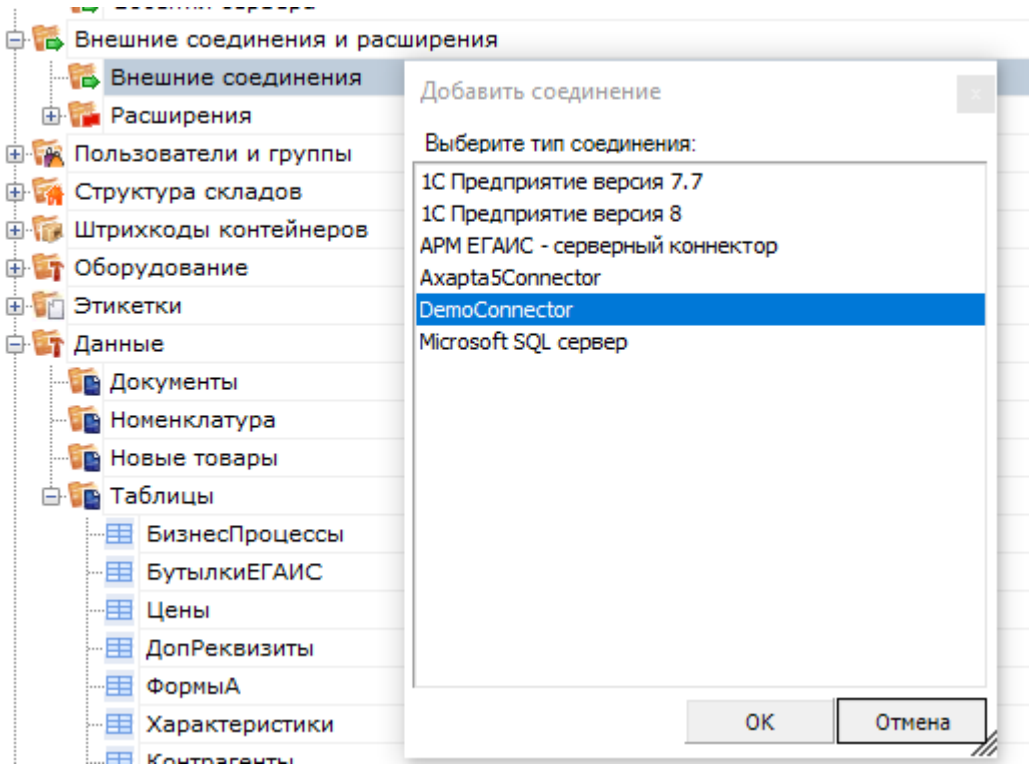
2019-08-05 10:08:08.6537|ERROR|NLogger.WriteNlogEvent| Коннекторы не загружены! Обнаружена неподписанная сборка: C:\ProgramData\Cleverence\Базы Mobile SMARTS\e3945857-308f-4829-92e2-720dc11d1bec\Server\DataService\bin\Cleverence.Connectivity.DemoConnector.dll

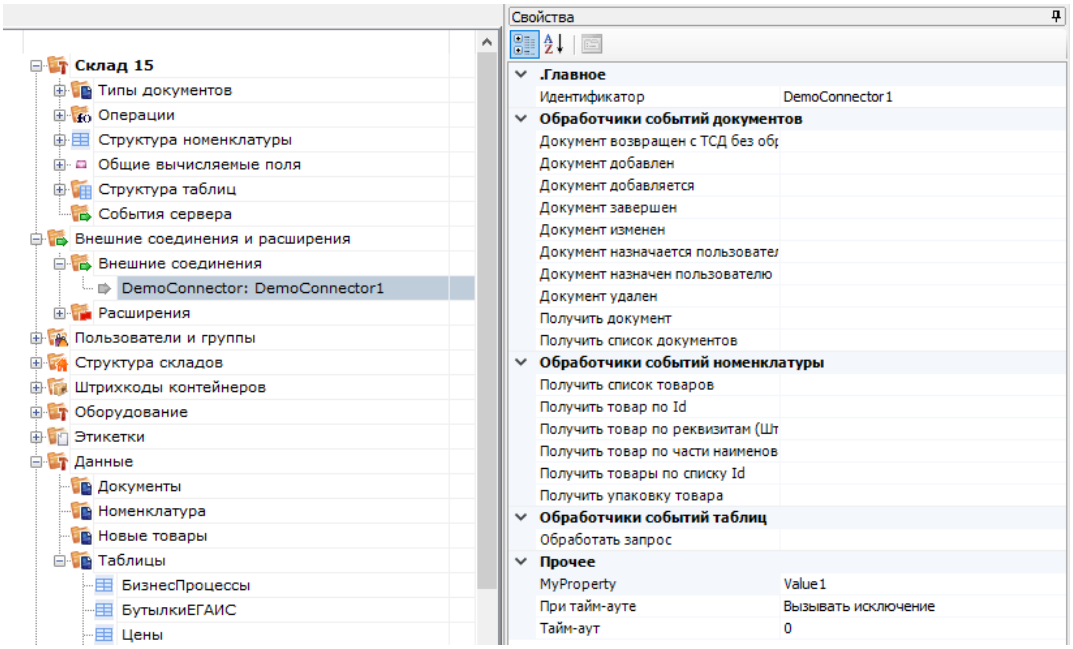
Видим, что dll коннектора не загрузилась. В некоторых случаях загрузка неподписанной сборки приводит

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):

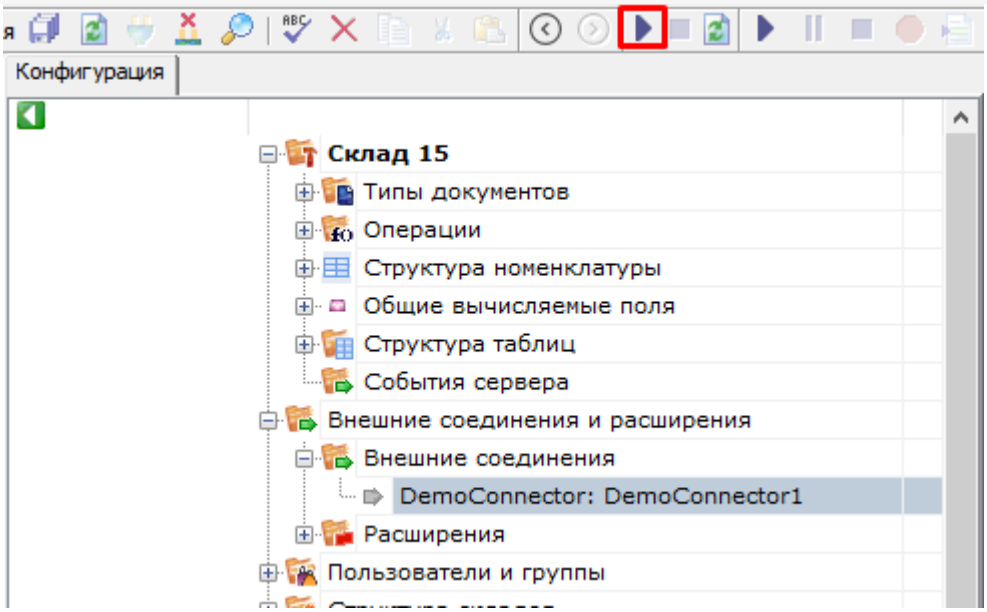
C:\Program Files (x86)\Cleverence Soft\Mobile SMARTS\Server\Cleverence.MobileSMARTS.Server.exe /debug

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

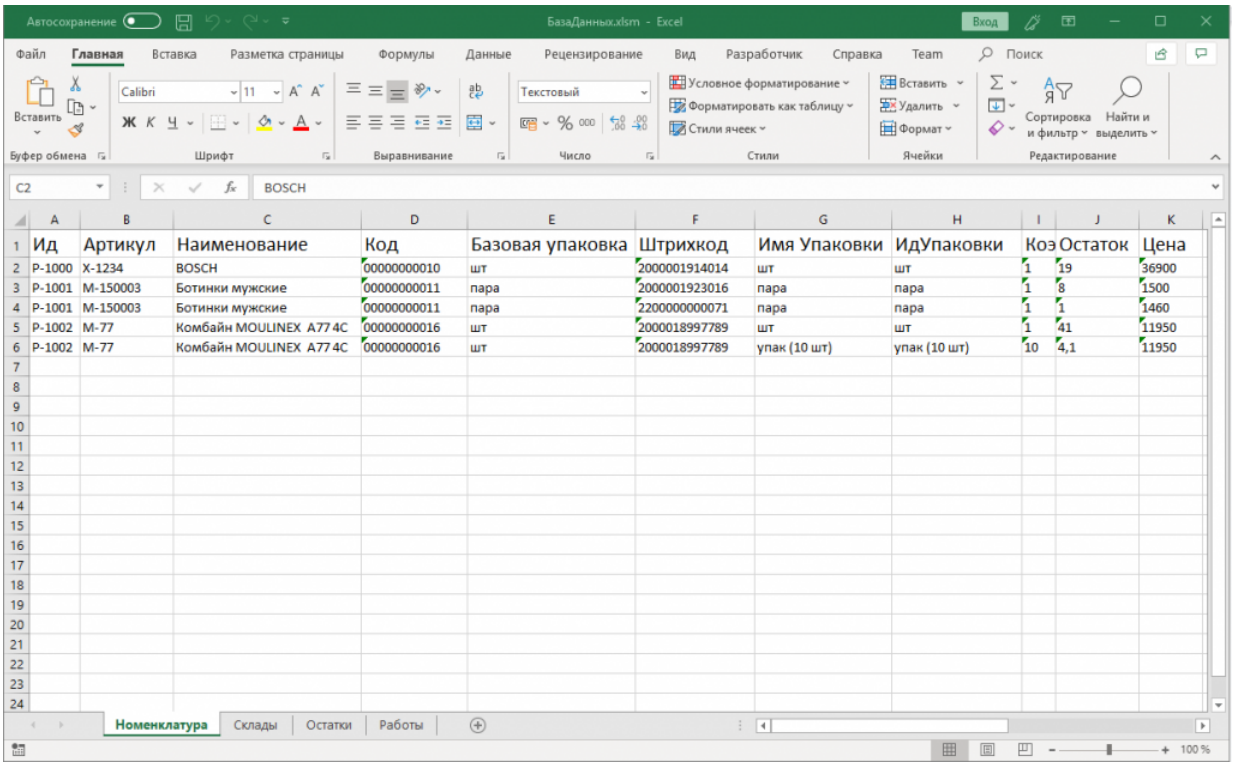
Для примера разработаем коннектор, который будет выполнять получение данных из файла Excel онлайн по запросам с ТСД. В примере будет рассмотрена, в том числе, перегрузка функций-обработчиков событий сервера для получения номенклатуры, документов и обработки запросов к таблицам.

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.



Ид	Артикул	Наименование	Код	Базовая упаковка	Штрихкод	Имя Упаковки	ИдУпаковки	Коэ	Остаток	Цена
P-1000	X-1234	BOSCH	00000000010	шт	2000001914014	шт	шт	1	19	36900
P-1001	M-150003	Ботинки мужские	00000000011	пара	2000001923016	пара	пара	1	8	1500
P-1001	M-150003	Ботинки мужские	00000000011	пара	2200000000071	пара	пара	1	1	1460
P-1002	M-77	Комбайн MOULINEX A77 4C	00000000016	шт	2000018997789	шт	шт	1	41	11950
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

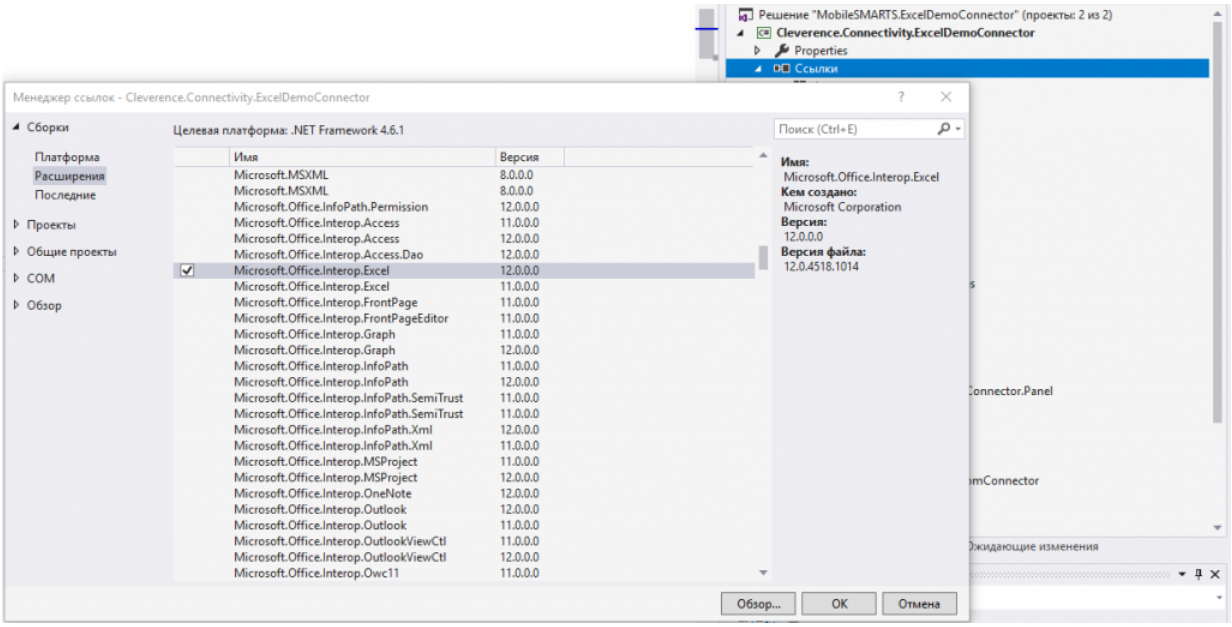
В Visual Studio создаем «Решение» (solution) и два проекта в нем:

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.

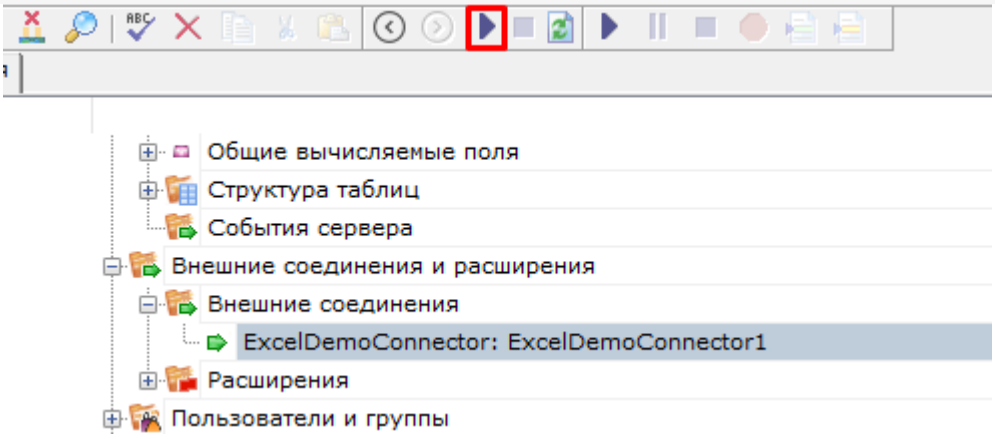
Деинициализация:

```
[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.

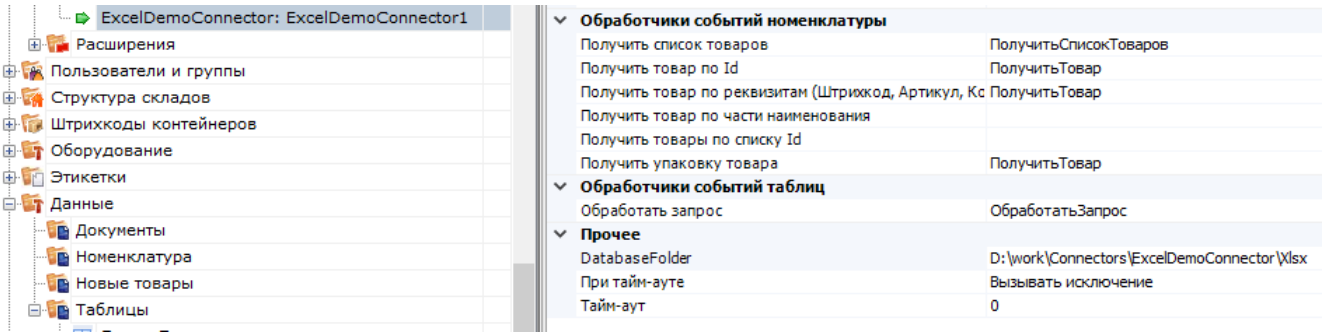
Событие «Получить список товаров» возникает, когда пользователь на ТСД заходит в список номенклатуры (выбор товара по 0 в действии «Выбор номенклатуры») и в базе нет выгруженного справочника. Перегрузим в коннекторе функцию GetProductsList:

```
[C#]

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

Функция возвращает PackedProductCollection со всем списком номенклатуры.

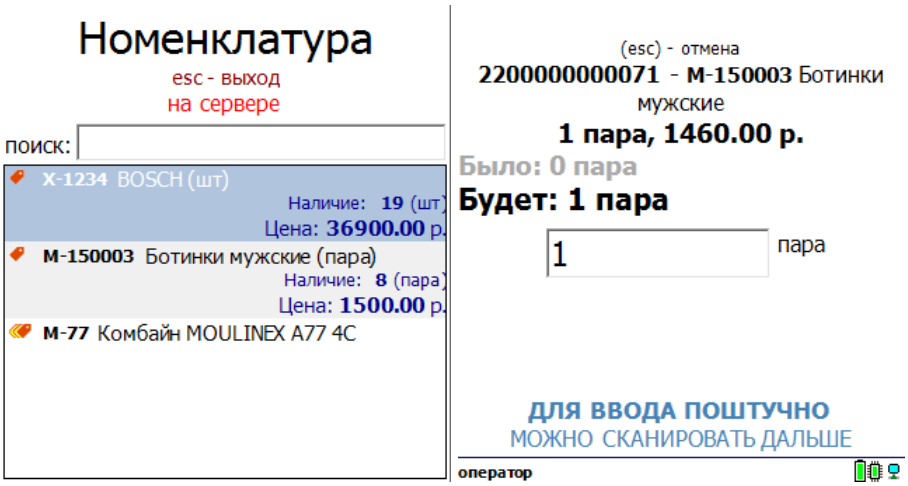
Проверим, как происходит получение номенклатуры. В настройках коннектора нужно указать путь к папке, в которой находится файл Excel со справочниками (DatabaseFolder). Нужно заполнить обработчики событий

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)
{
...
}
```

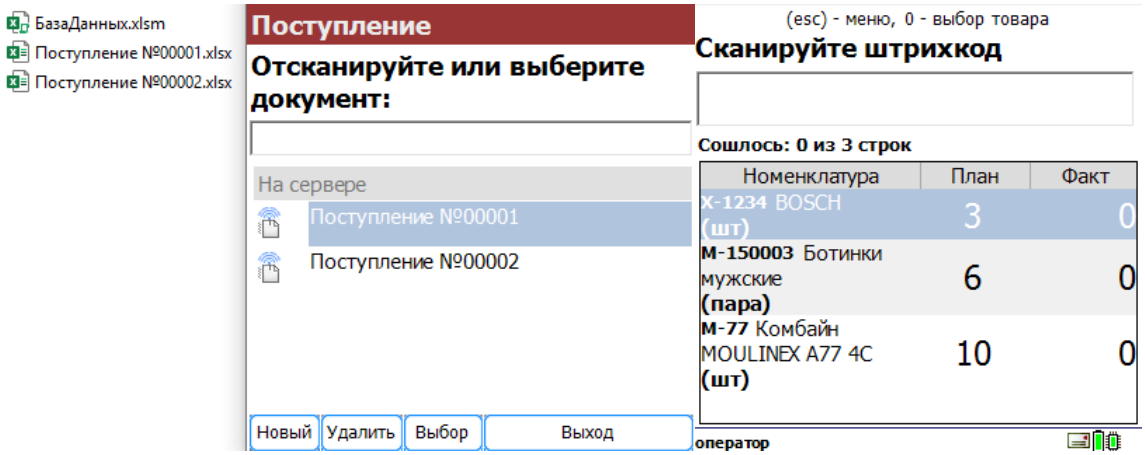
Функция возвращает DocumentDescriptionCollection со списком описаний документов (DocumentDescription), готовых для отдачи на ТСД. Список получаем на основе имен файлов xlsx в папке с данными.

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).

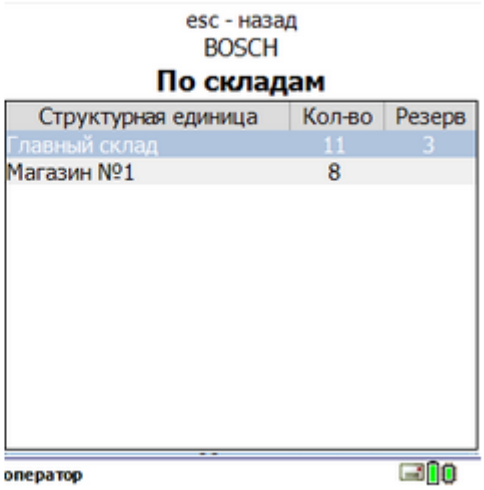
Реализуем получение данных онлайн при запросах к таблицам Mobile SMARTS. Например, в конфигурации «Магазин 15» есть таблицы «Склады», «Остатки» и др. На ТСД может выполняться получение списка всех складов, запрос остатков определенного товар и др. Если таблица хранится на сервере, в настройках таблицы 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:

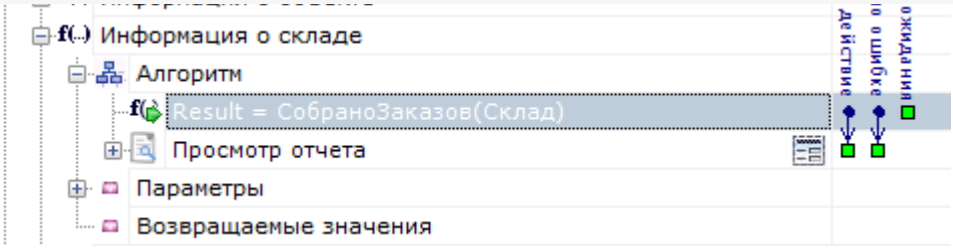
	A	B	C
1	Комплектовщик	Склад	Собрано заказов
2	Фахуртдинов	Главный склад	352
3	Петров	Главный склад	278
4	Коршунов	Магазин №1	94
5	Мухамедов	Магазин №2	65

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

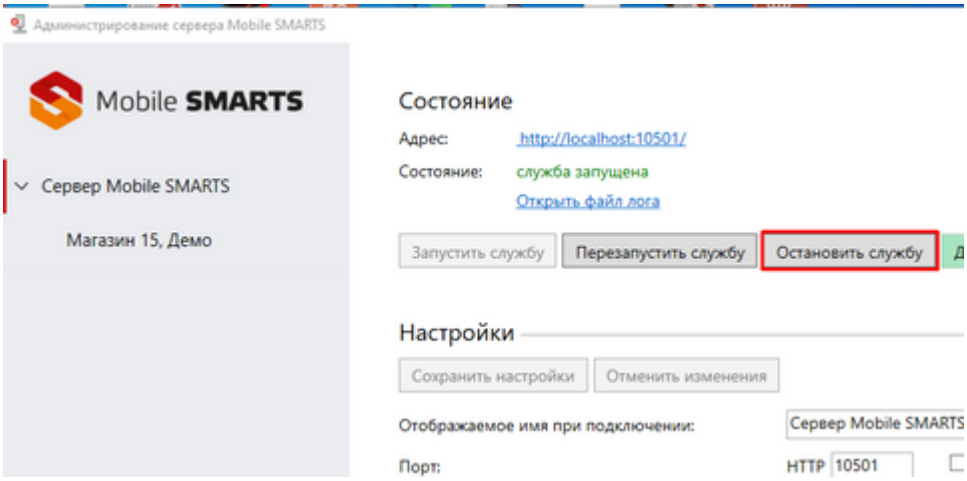
Последние изменения: 2024-03-26

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.

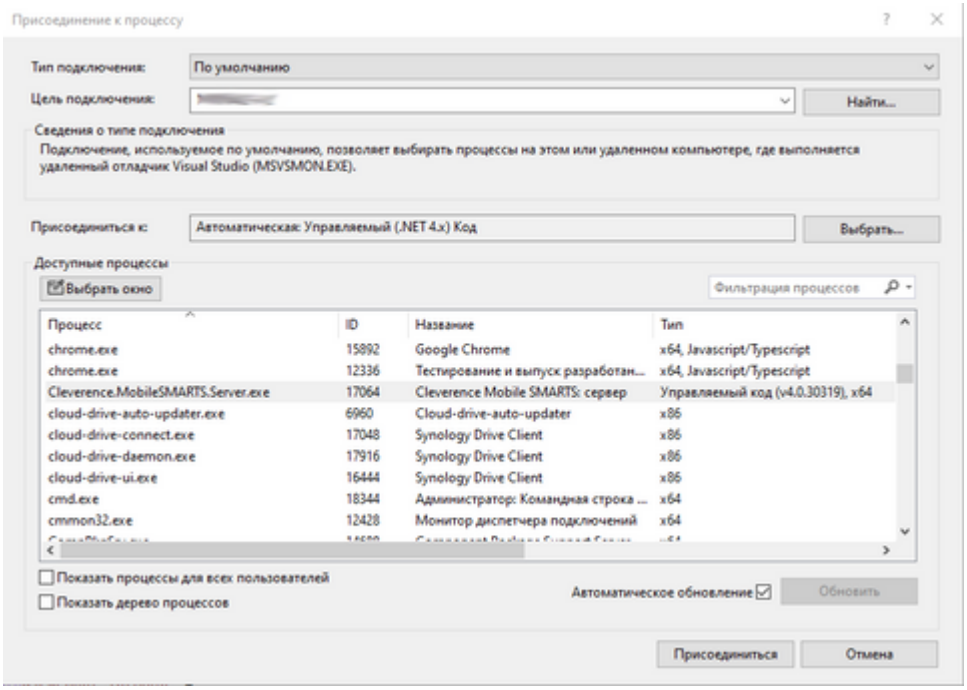
Для этого остановите сервис сервера:



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

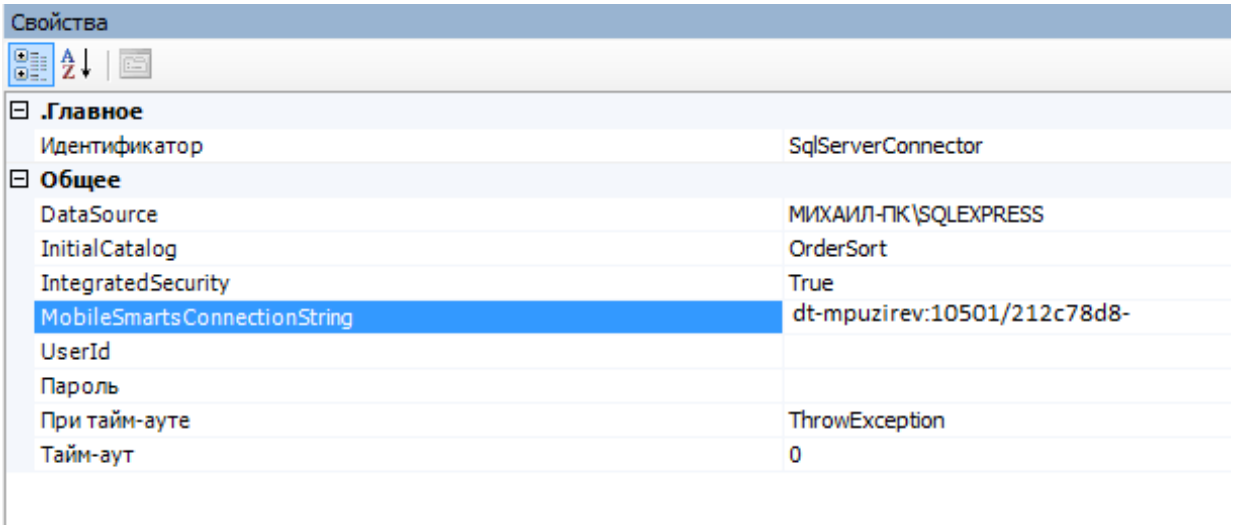
Пример запроса к базе MSSQL с помощью коннектора из Mobile SMARTS

Последние изменения: 2024-03-26

Кроме хранимых процедур, которые должны возвращать xml, можно использовать маппинг таблиц базы SQL на объекты Mobile SMARTS.

Скачать пример базы Mobile SMARTS с настроенным маппингом и базы SQL

Для работы с базой SQL Server в конфигурацию Mobile SMARTS нужно добавить коннектор к SQL Server и настроить подключение к базе.



Для получения данных из базы можно использовать действие «Вызов внешней системы». Также есть возможность с помощью коннектора обрабатывать события сервера Mobile SMARTS. Для получения номенклатуры онлайн по запросу с терминала используется событие «Получить товар».

После этого создайте в базе SQL хранимую процедуру:

```
CREATE PROCEDURE [dbo].[GetProduct]
 (
 @productId varchar(50),
 @packingId varchar(50),
 @userId varchar(50),
 @mode int,
 @resultXml xml OUTPUT
 )
 AS
 BEGIN
 ...
 END
```

В обработчике события «Получить товар» укажите SqlServerConnector:serverEvent_GetProduct (SqlServerConnector — ID коннектора, serverEvent_ — указание на обработчик серверного события). Процедура

должна возвращать через параметр @resultXml xml-файл особого вида, который представляет собой сериализованные объекты Mobile SMARTS. В случае режима 1 (поиск по ШК или другим атрибутам товара) это объекты Cleverence.Warehouse.PackedProduct (если найден один товар) или Cleverence.Warehouse.PackedProductCollection (несколько объектов), подробнее здесь. В процедуре нужно создать соответствующие COM-объекты, получить xml с помощью вызова ToXml объекта StorageConnector, можно также самим сформировать нужный xml, пример:

```
IF @mode = 1
  BEGIN
  SET @resultXml = '<?xml version="1.0" encoding="windows-1251"?>
  <PackedProduct expiredDate="0001-01-01T00:00:00" quantity="1" registrationDate="0001-01-
01T00:00:00"><Fields />
  <Product barcode="" basePackingId="шт" id="cbcf492a-55bc-11d9-848a-00112f43529a"
marking="X-1234" name="BOSCH">
  <Packings><Packing barcode="888999" id="шт" marking="" Int32_qty="0" Int32_price="27960"
currency="RUB"></Packing></Packings>
  </Product>
  <Packing barcode="888999" id="шт" marking="" Int32_qty="0" Int32_price="27960"
currency="RUB"></Packing>
  <Quantities />
  </PackedProduct>'
  END
```

При попадании завершенного документа с ТСД на сервер Mobile SMARTS вызывается событие «ДокументЗавершен» — с помощью обработчика этого события можно загрузить документ в базу SQL. Есть два варианта обработчика для SQL

1. SqlServerConnector:serverEvent_DocumentCompleted)
2. SqlServerConnector:serverEvent_DocumentCompletedXml

```
CREATE PROCEDURE [dbo].[DocumentCompleted]
(
 @documentId varchar(50),
 @resultXml xml OUTPUT
 )
 AS
 BEGIN
 /*
```

```
CREATE PROCEDURE [dbo].[DocumentCompletedXml]
(
@documentXml xml,
@result int OUTPUT
)
AS
BEGIN
SET @result = 1;
END
```

Нужно создать COM-объект Cleverence.Warehouse.StorageConnector, выполнить подключение к базе Mobile SMARTS с помощью SelectCurrentApp (<строка подключения к базе Mobile SMARTS или ID базы>), загрузить с сервера документ при помощи GetDocument (@documentId), обработать документ и, если загрузка выполнилась успешно, удалить с сервера документ при помощи RemoveDocument (@documentId).

```
  */
  SET @resultXml = '';
  END
```

documentXml содержит xml документа Mobile SMARTS, который можно разобрать внутри процедуры.

Если через result возвращается 1, коннектор удалит документ с сервера.

Для получения списка документов для работы на терминале и выгрузке из базы SQL используются события «ПолучитьСписокДокументов» и «ПолучитьДокумент».

Примеры процедур:

```
CREATE PROCEDURE [dbo].[GetDocumentsList]
  (
  @userId varchar(50),
  @documentTypeName varchar(50),
  @resultXml xml OUTPUT
  )
  AS
  BEGIN
  SET @resultXml = '<?xml version="1.0" encoding="windows-1251"?>
  <DocumentDescriptionCollection xmlns:clr="http://schemas.cleverence.ru/clr">
  <DocumentDescription barcode="123" createDate="0001-01-01T00:00:00"
distributeByBarcode="True" documentTypeName="Инвентаризация" id="123"
name="Инвентаризация 123" warehouseId="1"><Fields />
  </DocumentDescription>
  </DocumentDescriptionCollection>';
  END
```

```
CREATE PROCEDURE [dbo].[GetDocument]
(
@userId varchar(50),
@documentTypeName varchar(50),
@documentId varchar(50),
@mode int,
@resultXml xml OUTPUT
)
AS
BEGIN
SET @resultXml = '<?xml version="1.0" encoding="windows-1251"?>
<Document xmlns:clr="http://schemas.cleverence.ru/clr" barcode="00003" createDate="2014-03-
06T16:53:33" deviceId="" deviceIP="" deviceName="" documentTypeName="Инвентаризация"
id="123" name="Инвентаризация товаров на складе РТЦУТД00003 от 26.06.2008 11:22:45"
appointment="оператор" userId="оператор" userName="оператор" warehouseId="1">
<DeclaredItems>
<DocumentItem declaredQuantity="8" expiredDate="0001-01-01T00:00:00" packingId="пара"
productId="dee6e1d0-55bc-11d9-848a-00112f43529a" registeredDate="0001-01-01T00:00:00">
<Fields capacity="4"><FieldValue fieldName="price"><Value clr:Type="Int32">1424</Value>
</FieldValue></Fields></DocumentItem>
<DocumentItem declaredQuantity="20" expiredDate="0001-01-01T00:00:00" packingId="3"
productId="dee6e1d0-55bc-11d9-848a-00112f43529a" registeredDate="0001-01-01T00:00:00">
<Fields capacity="4"><FieldValue fieldName="descr"><Value clr:Type="String">6, 39,
Зеленый</Value></FieldValue><FieldValue fieldName="price"><Value
clr:Type="Int32">1424</Value></FieldValue></Fields></DocumentItem>
<DocumentItem declaredQuantity="9" expiredDate="0001-01-01T00:00:00" packingId="пара"
productId="dee6e1d3-55bc-11d9-848a-00112f43529a" registeredDate="0001-01-01T00:00:00">
<Fields capacity="4"><FieldValue fieldName="price"><Value clr:Type="Int32">1780</Value>
</FieldValue></Fields></DocumentItem>
<DocumentItem declaredQuantity="20" expiredDate="0001-01-01T00:00:00" packingId="3"
productId="dee6e1d3-55bc-11d9-848a-00112f43529a" registeredDate="0001-01-01T00:00:00">
<Fields capacity="4"><FieldValue fieldName="descr"><Value clr:Type="String">6, 39,
Белый</Value></FieldValue><FieldValue fieldName="price"><Value
clr:Type="Int32">1780</Value></FieldValue></Fields></DocumentItem>
</DeclaredItems><CurrentItems capacity="0" />
<ClassificatorIds capacity="0" />
<Classificators capacity="0" />
<ClassificatorUsings capacity="0" /><Errors capacity="0" />
<Fields capacity="4"><FieldValue fieldName="КонтрольКолва"><Value
clr:Type="Int32">0</Value></FieldValue><FieldValue fieldName="ПоЯчейкам"><Value
clr:Type="Int32">0</Value></FieldValue>
</Fields>
<Tables capacity="0" />
</Document>'
END
```

Также не обязательно формировать xml вручную, можно использовать COM-объекты Mobile SMARTS.

Не нашли что искали?

Задать вопрос в техническую поддержку