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);

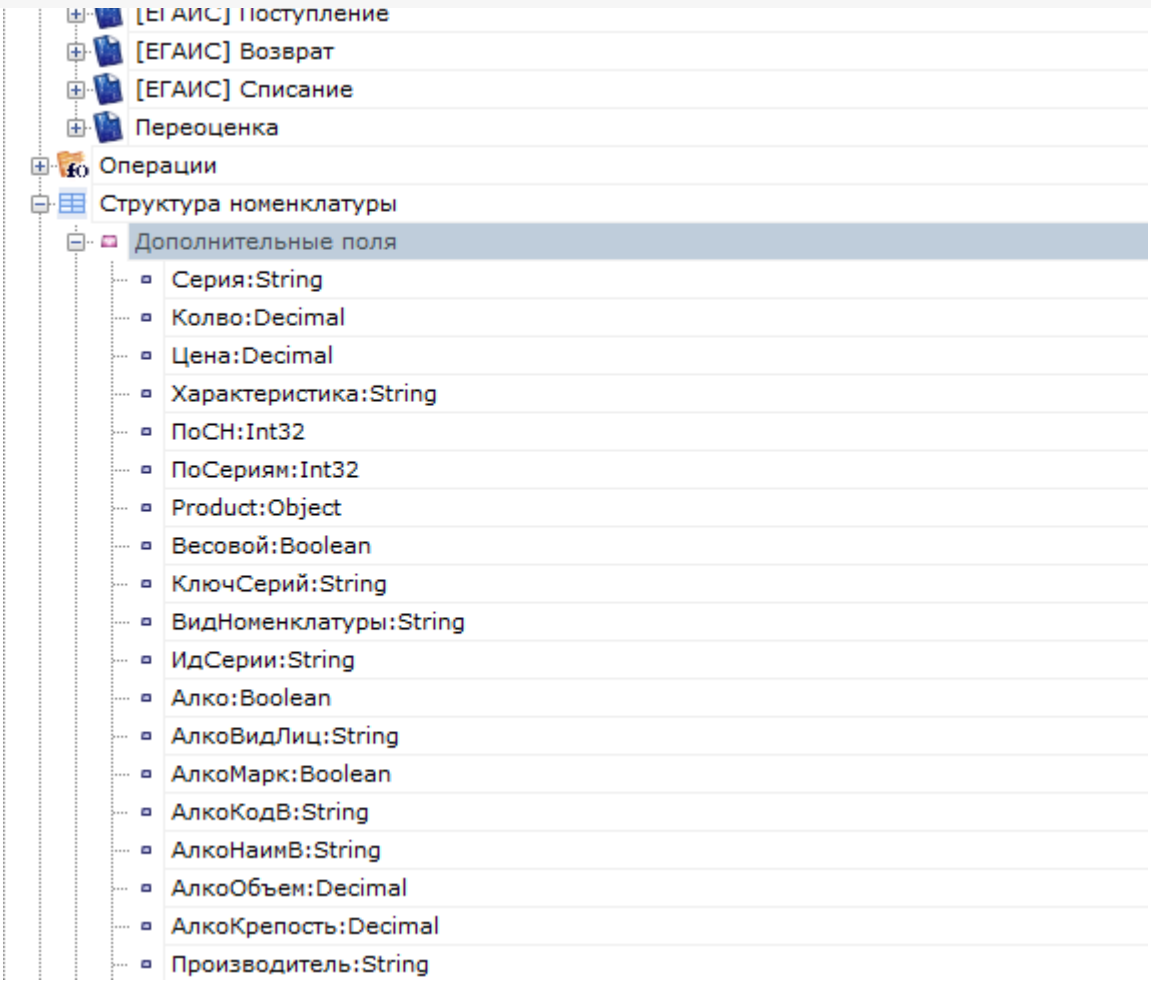
Packing [Упаковка]

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