# Документация WinShop COM-сервер

Оглавление

[Документация WinShop COM-сервер 1](#_Toc485197501)

[Навигация по товарам / разделам / группам / полям групп 4](#_Toc485197502)

[Описание основных полей товара / раздела 4](#_Toc485197503)

[Функция GetLastError 7](#_Toc485197504)

[Функция GetLastErrorStr 8](#_Toc485197505)

[Функция Connect 9](#_Toc485197506)

[Функция Disconnect 10](#_Toc485197507)

[Функция GetDatabaseInfo 10](#_Toc485197508)

[Функция UpdateShop 11](#_Toc485197509)

[Функция First 12](#_Toc485197510)

[Функция First2 13](#_Toc485197511)

[Функция Next 14](#_Toc485197512)

[Функция Prev 15](#_Toc485197513)

[Функция Last 16](#_Toc485197514)

[Функция Last2 16](#_Toc485197515)

[Функция FindById 17](#_Toc485197516)

[Функция FindById2 18](#_Toc485197517)

[Функция FindByField 19](#_Toc485197518)

[Функция FindByField2 20](#_Toc485197519)

[Функция GetIds 21](#_Toc485197520)

[Функция GetIds2 22](#_Toc485197521)

[Функция GetValue 23](#_Toc485197522)

[Функция GetValueStr 24](#_Toc485197523)

[Функция SetValue 25](#_Toc485197524)

[Функции SetProdValue 26](#_Toc485197525)

[Функции SetCategValue 28](#_Toc485197526)

[Функция AddItem 30](#_Toc485197527)

[Функция AddItem2 30](#_Toc485197528)

[Функция AddItems 31](#_Toc485197529)

[Функция DeleteItem 32](#_Toc485197530)

[Функция Refresh 33](#_Toc485197531)

[Поля для сортировки 34](#_Toc485197532)

[Функция Sort 35](#_Toc485197533)

[Функция SetQueue 37](#_Toc485197534)

[Функции GetCurrencyInfo, SetCurrencyInfo 38](#_Toc485197535)

[Функция FirstGroup 39](#_Toc485197536)

[Функция NextGroup 40](#_Toc485197537)

[Функция PrevGroup 41](#_Toc485197538)

[Функция LastGroup 42](#_Toc485197539)

[Функция BOFGroups 42](#_Toc485197540)

[Connect, PrevGroup, Навигация по товарам / разделам / группам / полям групп. 43](#_Toc485197541)

[Функция EOFGroups 43](#_Toc485197542)

[Connect, Навигация по товарам / разделам / группам / полям групп. 44](#_Toc485197543)

[Функция FindGroupById 44](#_Toc485197544)

[Функция FindGroupByCode 45](#_Toc485197545)

[Функция FindGroupByName 46](#_Toc485197546)

[Функция GetGroupId 46](#_Toc485197547)

[Функция GetGroupCode 47](#_Toc485197548)

[Функция GetGroupName 48](#_Toc485197549)

[Функция SetGroupCode 49](#_Toc485197550)

[Функция SetGroupName 50](#_Toc485197551)

[Функция AddGroup 50](#_Toc485197552)

[Функция DeleteGroup 51](#_Toc485197553)

[Функция RefreshGroups 52](#_Toc485197554)

[Функция FirstField 53](#_Toc485197555)

[Функция NextField 54](#_Toc485197556)

[Функция PrevField 54](#_Toc485197557)

[Функция LastField 55](#_Toc485197558)

[Функция BOFFields 56](#_Toc485197559)

[Connect, PrevField, Навигация по товарам / разделам / группам / полям групп. 57](#_Toc485197560)

[Функция EOFFields 57](#_Toc485197561)

[Connect, Навигация по товарам / разделам / группам / полям групп. 58](#_Toc485197562)

[Функция FindFieldById 58](#_Toc485197563)

[Функция FindFieldByName 59](#_Toc485197564)

[Функция FindFieldByCaption 59](#_Toc485197565)

[Функция FindFieldByType 60](#_Toc485197566)

[Функция FindFieldByPurpose 61](#_Toc485197567)

[Функция GetFieldId 62](#_Toc485197568)

[Функция GetFieldName 63](#_Toc485197569)

[Функция GetFieldCaption 64](#_Toc485197570)

[Функция GetFieldType 65](#_Toc485197571)

[Функция GetFieldPurpose 66](#_Toc485197572)

[Функция SetFieldName 66](#_Toc485197573)

[Функция SetFieldCaption 67](#_Toc485197574)

[Функция SetFieldType 68](#_Toc485197575)

[Функция SetFieldPurpose 69](#_Toc485197576)

[Функция AddField 70](#_Toc485197577)

[Функция DeleteField 72](#_Toc485197578)

[Функция RefreshFields 72](#_Toc485197579)

[Функция SetFieldQueue 73](#_Toc485197580)

[Пример для Delphi 7 74](#_Toc485197581)

[Пример для 1C: Предприятие 7.7 80](#_Toc485197582)

[Пример для 1C: Предприятие 8.2 97](#_Toc485197583)

# Навигация по товарам / разделам / группам / полям групп

Принцип навигации состоит в том, что устанавливается курсор (указатель) на записи, с которыми будут производиться операции чтения, редактирования, удаления.

Навигация по товарам / разделам может осуществляться двумя способами:

- "по ключу" – курсор устанавливается на один товар / раздел;

- "по значению поля" – курсор устанавливается на несколько товаров / разделов.

Функции для навигации по товарам / разделам:

- "по ключу" - **FindById**, **FindById2**, **First**, **First2**, **Next**, **Prev**, **Last**, **Last2**, **AddItem**, **AddItem2**, **AddItems**;

- "по значению поля" - **FindByField**, **FindByField2**.

Навигацию "по ключу" рекомендуется использовать для таких задач, как синхронизация содержимого магазина с базами данных других приложений (бухгалтерских, складских и проч.). Навигация "по значению полю" удобна, если, например, необходимо удалить все товары со статусом "новинка".

Функции для навигации по группам: **FirstGroup**, **NextGroup**, **PrevGroup**, **LastGroup**, **FindGroupById**, **FindGroupByCode**, **FindGroupByName**, **AddGroup**.

Функции для навигации по полям группы: **FirstField**, **NextField**, **PrevField**, **LastField**, **FindFieldByName**, **FindFieldByCaption**, **FindFieldByType**, **FindFieldByPurpose**, **AddField**.

# Описание основных полей товара / раздела

|  |  |  |
| --- | --- | --- |
| **Имя поля** | **Тип поля** | **Описание поля** |
| item\_id | целое число | Идентификатор товара / раздела.  Доступен только для чтения. |
| item\_code | Строка | Код товара / раздела.  Максимальный размер – 20 символов. |
| url | Строка | Адрес страницы товара / раздела.  Максимальный размер – 100 символов. |
| name | Строка | Краткое название товара / раздела.  Максимальный размер – 100 символов. |
| headline | Строка | Полное название товара / раздела.  Максимальный размер – 200 символов. |
| abstract | Текст | Краткое описание товара / раздела.  Максимальный размер – 200 символов. |
| info | Текст | Полное описание товара / раздела.  Максимальный размер – 200 символов. |
| image | Строка | Имя файла картинки товара / раздела.  Максимальный размер – 255 символов. |
| icon | Строка | Имя файла иконки товара / раздела.  Максимальный размер – 255 символов. |
| \_\_image1 | Строка | Имя файла иконки2 товара.  Максимальный размер – 255 символов. |
| \_\_image2 | Строка | Имя файла картинки2 товара.  Максимальный размер – 255 символов. |
| \_\_image3 | Строка | Имя файла иконки3 товара.  Максимальный размер – 255 символов. |
| \_\_image4 | Строка | Имя файла картинки3 товара.  Максимальный размер – 255 символов. |
| price | дробное число | Цена товара.  Разделитель дробной части – точка. |
| catalog\_price | дробное число | Цена товара до скидки.  Разделитель дробной части – точка. |
| currency\_id | Целое число или строка | Валюты, в которой указаны «Цена» и «Цена до скидки» товара.  Допустимые значения:  0 или BYR – Белорусский рубль  1 или RUB – Российский рубль  2 или USD – Доллар США  3 или EUR – Евро |
| status | целое число | Статус товара / раздела.  Значение от 0 до 3 (товар).  0 – “В резерве”  1 – “Доступен”  2 – “Нет на складе”  3 – “Новинка”  Значение от 0 до 2 (раздел)  0 – “В резерве”  1 – “Активен”  2 – “Не активен”  Для задания товару статуса "Под заказ" необходимо, во первых, чтобыт это товар имел статус - "Доступен" , во-вторых, указать количество дней, которое потребуется для доставки товара. Например, для назначения товару статуса "Под заказ в течение 7 дней", полю "status" присваиваем значение равное "1", полю "order\_delay" - "7". |
| primer\_id | целое число | Идентификатор основного раздела, в котором размещен товар (идентификатор родительского раздела, в котором размещен раздел).  Доступен только для чтения. Идентификатор главной страницы равен 1. |
| section\_ids | Строка | Идентификаторы разделов, в которых размещен товар / идентификатор родительского раздела, в котором находится раздел.  Максимальный размер – 255 символов.  Для того чтобы разместить товар (товары) в разделах, необходимо в качестве значения поля, указать через запятую идентификаторы разделов, в которых будет размещен товар (товары). Первый указанный раздел будет назначен основным для товара (товаров). Текущее размещение товара (товаров) в разделах, идентификаторы которых не были указаны, будет отменено. Чтобы к текущему размещению товара (товаров) добавить еще одно, указываемые идентификаторы разделов, в которых будет размещен товар, необходимо сопроводить символом “+”. Для отмены размещения товара (товаров) в разделах, необходимо в качестве значения поля указать через запятую идентификаторы разделов, размещение в которых нужно отменить (например: -27,52). Должен существовать хотя бы один раздел с указанным идентификатором.  Чтобы переместить раздел, необходимо в качестве значения поля, указать идентификатор раздела, в который будет перемещен раздел. Не допускается указывать идентификатор перемещаемого раздела, идентификатор раздела для которого перемещаемый раздел является родительским. |
| related\_ids | Cтрока | Идентификаторы разделов и товаров, которые являются сопутствующими для товара.  Максимальный размер – 255 символов.  Чтобы назначить товару (товарам) сопутствующие товары и разделы, необходимо в качестве значения поля указать через запятую идентификаторы товаров и разделов, которые будут являться сопутствующими. Ранее назначенные товару (товарам) сопутствующие товары и разделы, идентификаторы которых не были указаны, будут сброшены. Чтобы назначаемые сопутствующие товары и разделы были добавлены к уже назначенным, указываемые идентификаторы сопутствующих товаров и разделов необходимо сопроводить символом "+". Чтобы для товара (товаров) сбросить сопутствующие товары и разделы, необходимо в качестве значения поля указать через запятую идентификаторы сопутствующих товаров и разделов, которые будут сброшены, сопроводив их символом "-" (например: -27,52). Должен существовать хотя бы один сопутствующий товар / раздел с указанным идентификатором. При указании в качестве значения поля пустой строки, для товара (товаров) будут сброшены все назначенные сопутствующие товары и разделы. |
| type\_id | целое число | Идентификатор типа классификатора товара.  Идентификатор незаклассифицированного товара должен быть равным 15. |
| group\_id | целое число | Идентификатор группы товара. |
| seo.title | Строка | Текст заголовка страницы товара / раздела.  Максимальная длина – 255 символов. |
| seo.description | Текст | Краткое содержание страницы товара / раздела.  Максимальная длина – 65535 символов. |
| seo.keywords | Текст | Перечень ключевых слов, по которым будет осуществляться индексация страницы товара / раздела поисковыми системами.  Максимальная длина – 65535. |
| order\_delay | целое число | Срок доставки товара (дней).  Значение должно быть больше 0 |

# Функция GetLastError

Функция позволяет определить, было ли получено сообщение об ошибке.

Синтаксис

function GetLastError: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

Если сообщение об ошибке было получено, функция возвращает значение 0, иначе значение -1.

Заметки

Функция должна вызываться сразу после вызова функции, которая завершилась с ошибкой.

# Функция GetLastErrorStr

Функция используется для получения текста последней ошибки.

Синтаксис

function GetLastErrorStr: OleVariant;

Параметры

Функция не имеет параметров.

Возвращаемые значения

Функция возвращает текст последней ошибки. Если текст последней ошибки получить не удалось, функция возвращает пустую строку.

Заметки

Функция должна вызываться сразу после вызова функции, которая завершилась с ошибкой.

См. также

**GetLastError**.

# Функция Connect

Функция используется для открытия соединения (сессии) с WinShop-сервером.

Синтаксис

function Connect**(***Host*: OleVariant; *Port*: Integer; *UserName*: OleVariant; *Password*: OleVariant; *DatabaseName*: OleVariant): Integer;

Параметры

Host [in]

Имя или IP-адрес компьютера, на котором запущен WinShop-сервер. Значение можно оставить пустым, для подключения к WinShop-серверу, расположенному на локальном компьютере.

Port [in]

Номер порта. Значение можно оставить равным 0, тогда подключение будет производиться по принятому по умолчанию порту для WinShop равному 2006.

UserName [in]

Имя пользователя.

Password [in]

Пароль пользователя.

DatabaseName [in]

Имя магазина.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Для параметров *UserName* и *Password* необходимо использовать имя пользователя и пароль существующей в WinShop учетной записи пользователя (меню “Сервис” -> “Параметры” -> закладка “Пользователи”). Для параметра *DatabaseName* используйте имя магазина существующего в WinShop (меню “Сервис” -> “Параметры” -> закладка “Магазины”).

# Функция Disconnect

Функция используется для закрытия соединения (сессии) с WinShop-сервером.

Синтаксис

function Disconnect: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

# Функция GetDatabaseInfo

Функция возвращает информацию о магазине по его порядковому номеру в списке магазинов, управляемых WinShop-сервером.

Синтаксис

function GetDatabaseInfo(*Index*: Integer; out *DatabaseName*: OleVariant; out *DatabasePath*: OleVariant): Integer;

Параметры

Index [in]

Порядковый номер магазина.

DatabaseName [out]

Имя магазина.

DatabasePath [out]

Путь к базе данных магазина (на компьютере, где запущен WinShop-сервер).

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу. Подключаясь необязательно указывать имя магазина.

Если на сервере установлено три магазина, то для параметра *Index* = 0 будет возвращена информация по первому магазину, для параметра *Index* = 1 – второму магазину и так далее. Для параметра *Index* = 3 функция вернет код ошибки. Таким образом, чтобы получить полный список магазинов, необходимо вызывать данную функцию с увеличением параметра *Index* до тех пор, пока она не вернет код ошибки.

См. также

**Connect**.

# Функция UpdateShop

Функция используется для обновления данных в интернет-магазине.

Синтаксис

function UpdateShop(*UpdateMode*: Integer): Integer;

Параметры

UpdateMode [in]

Режим обновления интернет-магазина:

0 – полное;

1 – частичное.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу с использованием имени пользователя, имеющего права доступа "Администратор".

См. также

**Connect**.

# Функция First

Функция устанавливает курсор на первый товар.

Синтаксис

function First: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

В случае успешного завершения функция возвращает идентификатор первого товара. Если товары в магазине отсутствуют, то функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

Функция задает способ навигации “по ключу”.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция First2

Функция устанавливает курсор на первый товар / раздел.

Синтаксис

function First2(*Where*: Integer): Integer;

Параметры

Where [in]

Перемещаемся к первому:

1 – товару;

2 – разделу;

3 – товару или разделу.

Возвращаемые значения

В случае успешного завершения функция возвращает идентификатор первого товара / раздела. Если товары / разделы в магазине отсутствуют, то функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

Функция задает способ навигации “по ключу”.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**, **First**.

# Функция Next

Функция устанавливает курсор на следующий товар / раздел.

Синтаксис

function Next: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

В случае успешного завершения функция возвращает идентификатор следующего товара / раздела. Если товары / разделы в магазине отсутствуют или достигнут конец набора товаров / разделов, то функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

Функция может быть использована только в том случае, если задан способ навигации “по ключу”.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция Prev

Функция устанавливает курсор на предыдущий товар / раздел.

Синтаксис

function Prev: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

В случае успешного завершения функция возвращает идентификатор предыдущего товара / раздела. Если товары / разделы в магазине отсутствуют или достигнуто начало набора товаров / разделов, то функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

Функция может быть использована только в том случае, если задан способ навигации “по ключу”.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция Last

Функция устанавливает курсор на последний товар.

Синтаксис

function Last: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

В случае успешного завершения функция возвращает идентификатор последнего товара. Если товары в магазине отсутствуют, то функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

Функция задает способ навигации “по ключу”.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция Last2

Функция устанавливает курсор на последний товар / раздел.

Синтаксис

function Last2(*Where*: Integer): Integer;

Параметры

Where [in]

Перемещаемся к последнему:

1 – товару;

2 – разделу;

3 – товару или разделу.

Возвращаемые значения

В случае успешного завершения функция возвращает идентификатор последнего товара / раздела. Если товары / разделы в магазине отсутствуют, то функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

Функция задает способ навигации “по ключу”.

См. также

**Connect**, **Навигация по товарам / разделам**, **Выборка товаров / разделов**, **Last**.

# Функция FindById

Функция ищет товар по идентификатору и устанавливает на него курсор.

Синтаксис

function FindById(*ItemId*: Integer): Integer;

Параметры

ItemId [in]

Идентификатор товара, по которому будет происходить поиск.

Возвращаемые значения

Если товар найден, то функция возвращает значение 1. Если товар не найден, то функция возвращает 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

Функция задает способ навигации “по ключу".

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция FindById2

Функция ищет товар / раздел по идентификатору и устанавливает на него курсор.

Синтаксис

function FindById2(*ItemId*, *Where*: Integer): Integer;

Параметры

ItemId [in]

Идентификатор товара / раздела, по которому будет происходить поиск.

Where [in]

Ищем:

1 – товар;

2 – раздел;

3 – товар или раздел.

Возвращаемые значения

Если товар / раздел найден, то функция возвращает значение 1. Если товар / раздел не найден, то функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

Функция задает способ навигации “по ключу”.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**, **FindById**.

# Функция FindByField

Функция ищет товары по значению основного поля и устанавливает на них курсор.

Синтаксис

function FindByField(*FieldName*: OleVariant; *FieldValue*: OleVariant): Integer;

Параметры

FieldName [in]

Имя основного поля товара, по которому будет происходить поиск.

FieldValue [in]

Значение основного поля товара, по которому будет происходить поиск.

Возвращаемые значения

В случае успешного завершения функция возвращает количество найденных товаров. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

Функция задает способ навигации “по значению полю”.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция FindByField2

Функция ищет товары / разделы по значению основного поля и устанавливает на них курсор.

Синтаксис

function FindByField2(*FieldName*: OleVariant; *FieldValue*: OleVariant; *Where*: Integer): Integer;

Параметры

FieldName [in]

Имя основного поля товара / раздела, по которому будет происходить поиск.

FieldValue [in]

Значение основного поля товара / раздела, по которому будет происходить поиск.

Where [in]

Ищем:

1 – товары;

2 – разделы;

3 – товары и разделы.

Возвращаемые значения

В случае успешного завершения функция возвращает количество найденных товаров / разделов. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

Функция задает способ навигации “по значению поля”.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**, **FindByField**.

# Функция GetIds

Функция используется для получения идентификаторов товаров, на которые установлен курсор.

Синтаксис

function GetIds(out *Ids*: OleVariant):Integer;

Параметры

Ids [out]

Разделенные запятой, идентификаторы товаров, которые будут возвращены.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

Connect, Навигация по товарам / разделам / группам / полям групп.

# Функция GetIds2

Функция используется для получения идентификаторов товаров / разделов, на которые установлен курсор.

Синтаксис

function GetIds2(out *Ids*: OleVariant; Where: Integer): Integer;

Параметры

Ids [out]

Разделенные запятой, идентификаторы товаров / разделов, которые будут возвращены.

Where [in]

Получаем идентификаторы:

1 – товаров;

2 – разделов;

3 – товаров и разделов.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

Connect, Навигация по товарам / разделам / группам / полям групп, GetIds.

# Функция GetValue

Функция используется для получения значений основных полей и полей группы товара / раздела, на который установлен курсор.

Синтаксис

function GetValue(FieldName: OleVariant; out FieldValue: OleVariant): Integer;

Параметры

FieldName [in]

Имя поля товара /раздела, значение которого необходимо получить.

FieldValue [out]

Значение поля товара / раздела, которое будет возвращено.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

Функции неважно, какой способ навигации задан, главное, чтобы в выборке был только один товар / раздел.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция GetValueStr

Функция используется для получения значений основных полей и полей группы товара / раздела на который установлен курсор.

Синтаксис

function GetValueStr(FieldName: OleVariant): OleVariant;

Параметры

FieldName [in]

Имя поля товара /раздела, значение которого необходимо получить.

Возвращаемые значения

В случае успешного завершения функция возвращает полученное значение поля. В случае ошибки функция возвращает пустую строку. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

Функции неважно, какой способ навигации задан, главное, чтобы в выборке был только один товар / раздел.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция SetValue

Функция используется для установки значений основным полям и полям групп товаров / разделов, на которые установлен курсор.

Синтаксис

function SetValue(*FieldName*, *FieldValue*: OleVariant): Integer;

Параметры

FieldName [in]

Имя поля товара / раздела, для которого устанавливаем значение.

FieldValue [in]

Значение поля, которое будет назначено товару / разделу.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Основные поля товара / раздела**, **Навигация по товарам / разделам / группам / полям групп**, **Connect**.

# Функции SetProdValue

Функции используется для установки значений полям выбранного товара.

Синтаксис

function SetProdValue(*FieldName*, *FieldValue*: OleVariant): Integer;

Параметры

FieldName [in]

Имя поля.

FieldValue [in]

Значение поля.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Для использования функции требуется:

1. Подключение к магазину (см. **Connect**). Если подключение к магазину будет отсутствовать, функция завершиться с ошибкой - “Нет подключения к магазину”.
2. Товар, выбранный “по ключу” (см. **Навигация по товарам**). Если товар выбран “по полю” или выбран раздел - функция вернет ошибку – “Не выбран товар”.

Список полей товара, доступных для редактирования:

* name – краткое название
* headline – полное название
* type\_id – тип
* price – цена
* catalog\_price – цена до скидки
* currency\_id – валюта
* status – статус
* order\_delay – под заказ
* item\_code – код
* abstract – краткое описание
* info – полное описание
* icon – иконка
* image – картинка
* \_\_image1 – иконка 2
* \_\_image2 – картинка 2
* \_\_image3 – иконка 3
* \_\_image4 – картинка 3
* group\_id - группа
* url – адрес страницы
* seo.title – заголовок (title)
* seo.description – краткое содержание (description)
* seo.keywords – ключевые слова (keywords)

+ поля группы “Товар” и вложенных в нее групп.

Функция является альтернативой для медленно работающей функции SetValue по части работы с товарами.

Функция не поддерживает работу с полями section\_ids и related\_ids.

Символ-разделитель для полей типа “дробное число” – “.” (точка).

При установке значения списковому полю (типы: “список с выбором одного значения” и “список с выбором нескольких значений”), если устанавливаемое значение не из списка значений этого поля, оно будет добавлено в список значений и назначено. Символ-разделитель для списковых полей типа “список с выбором нескольких значений” – “|” (вертикальная черта).

См. также

**Основные поля товара / раздела**, **Навигация по товарам / разделам / группам / полям групп**, **Connect**.

# Функции SetCategValue

Функции используется для установки значений полям выбранного раздела.

Синтаксис

function SetCategValue(*FieldName*, *FieldValue*: OleVariant): Integer;

Параметры

FieldName [in]

Имя поля.

FieldValue [in]

Значение поля.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Для использования функции требуется:

1. Подключение к магазину (см. **Connect**). Если подключение к магазину будет отсутствовать, функция завершиться с ошибкой - “Нет подключения к магазину”.
2. Раздел, выбранный по “ключу” (см. **Навигация по разделам**). Если раздел выбран “по полю” или выбран товар - функция вернет ошибку – “Не выбран раздел”.

Список полей раздела, доступных для редактирования:

* name – краткое название
* headline – полное название
* status – статус (опционально)
* item\_code – код
* abstract – краткое описание
* info – полное описание
* icon – иконка
* image – картинка
* url – адрес страницы (опционально)
* seo.title – заголовок (title)
* seo.description – краткое содержание (description)
* seo.keywords – ключевые слова (keywords)

+ поля группы “Раздел”

Функция является альтернативой для медленно работающей функции SetValue по части работы с разделами.

Функция не поддерживает работу с полями section\_ids и related\_ids.

Для главного (основного) раздела недоступны поля “status” и “url”. При попытке установить одному из этих полей значение, функция завершиться с ошибкой – “Поле “FieldName” не найдено”.

Символ-разделитель для полей типа “дробное число” – “.” (точка).

При установке значения списковому полю (типы: “список с выбором одного значения” и “список с выбором нескольких значений”), если устанавливаемое значение не из списка значений этого поля, оно будет добавлено в список значений и назначено. Символ-разделитель для списковых полей типа “список с выбором нескольких значений” – “|” (вертикальная черта).

См. также

**Основные поля товара / раздела**, **Навигация по товарам / разделам / группам / полям групп**, **Connect**.

# Функция AddItem

Функция добавляет новый товар в указанный раздел и устанавливает на него курсор.

Синтаксис

function AddItem(*SectionId*: Integer): Integer;

Параметры

SectionId [in]

Идентификатор раздела, в который будет добавлен новый товар. Раздел должен существовать.

Возвращаемые значения

В случае успешного завершения функция возвращает идентификатор добавленного товара. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

Функция задает способ навигации “по ключу”.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**, **AddItems**.

# Функция AddItem2

Функция добавляет новый товар / раздел в указанный раздел и устанавливает на него курсор.

Синтаксис

function AddItem2(*SectionId*: Integer; *GroupId*: Integer): Integer;

Параметры

SectionId [in]

Идентификатор раздела, в который будет добавлен новый товар / раздел. Раздел должен существовать.

GroupId [in]

Что добавляем:

Идентификатор группы <> 0 – товар. Группа должна существовать.

0 – раздел.

Возвращаемые значения

В случае успешного завершения функция возвращает идентификатор добавленного товара или созданного подраздела. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

Функция задает способ навигации по “по ключу”.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**, **AddItem**, **AddItems**.

# Функция AddItems

Функция добавляет новый товар в указанные разделы и устанавливает на него курсор.

Синтаксис

function AddItems(*SectionIds*: OleVariant): Integer;

Параметры

SectionIds [in]

Разделенные запятой, идентификаторы разделов, в которые будет добавлен новый товар. Должен существовать хотя бы один раздел. Первый указанный раздел будет назначен основным.

Возвращаемые значения

В случае успешного завершения функция возвращает идентификатор добавленного товара. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**, **AddItem**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

Функция задает способ навигации “по ключу”.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция DeleteItem

Функция используется для удаления товаров / разделов, на которые установлен курсор.

Синтаксис

function DeleteItem: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп.**

# Функция Refresh

Функция обновляет данные о товарах / разделах магазина.

Синтаксис

function Refresh: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**.

# Поля для сортировки

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя поля** | **Краткое описание поля** | **Возможность сортировки** | |
| **Товары** | **Разделы** |
| item\_code | Код товара / раздела |  |  |
| url | Адрес страницы товара / раздела |  |  |
| name | Краткое название товара / раздела |  |  |
| headline | Полное название товара / раздела |  |  |
| image | Имя файла картинки товара / раздела |  |  |
| icon | Имя файла иконки товара / раздела |  |  |
| price | Цена товара |  |  |
| catalog\_price | Цена товара до скидки |  |  |
| currency\_id | Валюта |  |  |
| web\_price | Цена товара в основной валюте |  |  |
| web\_catalog\_price | Цена товара до скидки в основной валюте |  |  |
| status | Статус товара / раздела |  |  |
| seo.title | Текст заголовка страницы товара / раздела |  |  |
| seo.description | Краткое содержание страницы товара / раздела |  |  |
| seo.keywords | Перечень ключевых слов, по которым будет осуществляться индексация страницы товара / раздела поисковыми системами |  |  |

Также можно указывать имена полей группы "Товар" - для товаров, группы "Раздел" - для разделов за исключением полей с типом "список с выбором одного значения" или "список с выбором нескольких значений".

# Функция Sort

Функция сортирует товары / разделы по нескольким полям, используя встроенный механизм сортировки - “quicksort”.

Синтаксис

function Sort(*SectionId*: Integer; *FieldName1*: OleVariant; *Asc1*: WordBool; *FieldName2*: OleVariant; *Asc2*: WordBool; *FieldName3*: OleVariant; *Asc3*: WordBool; *Where*: Integer): Integer;

Параметры

SectionId [in]

Идентификатор раздела, товары / разделы которого будут отсортированы. Раздел должен существовать.

*FieldName1* [in]

Имя первого поля для сортировки.

*Asc1* [in]

Порядок сортировки по первому полю:

True – по возрастанию;

False – по убыванию.

*FieldName2* [in]

Имя второго поля для сортировки.

*Asc2* [in]

Порядок сортировки товаров по второму полю.

*FieldName3* [in]

Имя третьего поля для сортировки.

*Asc3* [in]

Порядок сортировки по третьему полю.

*Where* [in]

Что сортируем:

1 – товары;

2 – разделы.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Поля для сортировки**, **Список основных полей товара / раздела**

# Функция SetQueue

Функция позволяет указать позицию товара / раздела относительно других сортируемых товаров / разделов при написании собственной сортировки.

Синтаксис

function SetQueue(*ItemId*: Integer; *Where*: Integer; *Queue*: Integer): Integer;

Параметры

*Item*Id [in]

Идентификатор сортируемого товара / раздела. Товар / раздел должен существовать.

*Where* [in]

Что сортируем:

1 – товар;

2 – раздел;

*Queue* [in]

Позиция товара / раздела по отношению к другим сортируемым товарам / раздела. Позиция должна задаваться с определенными промежутками. Например, первый товар – Queue = 0, второй товар – Queue = 16384, и так далее.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

К примеру, мы получили и отсортировали у себя в программе товары раздела "Одежда". Теперь, чтобы в WinShop товары отображались в той же последовательности, нужно установить курсор на раздел, товары которого были отсортированы и поочередно передать в функцию их идентификаторы, смещая для каждого позицию.

Функция может использоваться при сортировке сопутствующих товаров. Для этого необходимо установить курсор на товар, которому сопутствуют отсортированные товары, и передать в функцию их идентификаторы, указав для каждого позицию.

См. также

**Connect**, **Поля для сортировки**, **Список основных полей товара / раздела**

# Функции GetCurrencyInfo, SetCurrencyInfo

Функция возвращает параметры указанной валюты.

Синтаксис

function GetCurrencyInfo(CurrencyCode: OleVariant; out Main: WordBool; out Value: OleVariant; out Round: WordBool; out RoundValue: OleVariant; out RoundUp: WordBool): Integer;

function SetCurrencyInfo(CurrencyCode: OleVariant; Main: WordBool; Value: OleVariant; Round: WordBool; RoundValue: OleVariant; RoundUp: WordBool): Integer;

Параметры

CurrencyCode [in]

Код валюты, значения параметров которой необходимо получить. В качестве значения необходимо указать один из поддерживаемых кодов валюты:

BYR – Белорусский рубль;

RUB – Российский рубль;

USD – Доллар США;

EUR – Евро;

Main [out]

Значение показывает, является указанная валюта основной или нет.

Value [out]

Значение курса валюты в белорусских рублях. Разделитель дробной части – точка.

Для BYR значение всегда должно быть равным 1.

Round [out]

Значение показывает, будет ли производиться округление цен, при конвертировании в данную валюту (конвертирование производится при обновлении магазина цен всех товаров в основную валюту).

RoundValue [out]

Параметр кратности, используемый при округлении цен (т.е. полученная цена будет кратна данному значению). Разделитель дробной части – точка.

RoundUp [out]

Значение показывает, как будет производиться округление: в большую сторону (Правда) или математически (Ложь).

Возвращаемые значения

В случае успешного завершения функция возвращает 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

# Функция FirstGroup

Функция устанавливает курсор на первую группу.

Синтаксис

function FirstGroup: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

В случае успешного завершения функция возвращает идентификатор первой группы. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция NextGroup

Функция устанавливает курсор на следующую группу.

Синтаксис

function NextGroup: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

В случае успешного завершения функция возвращает идентификатор следующей группы. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция PrevGroup

Функция устанавливает курсор на предыдущую группу.

Синтаксис

function PrevGroup: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

В случае успешного завершения функция возвращает идентификатор предыдущей группы. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция LastGroup

Функция устанавливает курсор на последнюю группу.

Синтаксис

function LastGroup: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

В случае успешного завершения функция возвращает идентификатор последней группы. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция BOFGroups

Функция позволяет определить, расположен ли курсор в начале набора групп.

Синтаксис

function BOFGroups: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

Если курсор находится в начале набора групп (была вызвана функция FirstGroup, была вызвана функция PrevGroup, когда курсор находился на первой группе), функция возвращает значение 1. Если курсор находится в любом другом месте, функция возвращает значение 0. В случае ошибки функция возвращает -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

Функция обычно используется совместно с функцией PrevGroup.

См. также

# Connect, PrevGroup, Навигация по товарам / разделам / группам / полям групп.

# Функция EOFGroups

Функция позволяет определить, расположен ли курсор в конце набора групп.

Синтаксис

function EOFGroups: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

Если курсор находится в конце набора групп (была вызвана функция LastGroup, была вызвана функция NextGroup, когда курсор находился на последней группе), функция возвращает значение 1. Если курсор находится в любом другом месте, функция возвращает значение 0. В случае ошибки функция возвращает -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

Функция обычно используется совместно с функцией NextGroup.

См. также

# Connect, Навигация по товарам / разделам / группам / полям групп.

# Функция FindGroupById

Функция ищет группу по идентификатору и устанавливает на нее курсор.

Синтаксис

function FindGroupById(*Id*: Integer): Integer;

Параметры

Id [in]

Идентификатор группы, по которому будет происходить поиск.

Возвращаемые значения

Если группа найдена, функция возвращает значение 1. Если группа не найдена, то функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция FindGroupByCode

Функция ищет группу по коду и устанавливает курсор на первую найденную.

Синтаксис

function FindGroupByCode(*Code*: OleVariant): Integer;

Параметры

Code [in]

Код группы, по которому будет происходить поиск.

Возвращаемые значения

Если группа найдена, то функция возвращает значение 1. Если группа не найдена функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция FindGroupByName

Функция ищет группу по названию и устанавливает курсор не первую найденную.

Синтаксис

function FindGroupByName(*Name*: OleVariant): Integer;

Параметры

Name [in]

Название группы, по которому будет происходить поиск.

Возвращаемые значения

Если группа найдена, то функция возвращает значение 1. Если группа не найдена, то функция возвращает 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция GetGroupId

Функция используется для получения идентификатора группы, на которую установлен курсор.

Синтаксис

function GetGroupId: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

В случае успешного завершения функция возвращает идентификатор группы. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция GetGroupCode

Функция используется для получения кода группы, на которую установлен курсор.

Синтаксис

function GetGroupCode(out Code: OleVariant): Integer;

Параметры

Code [out]

Код группы, который будет возвращен.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция GetGroupName

Функция используется для получения названия группы, на которую установлен курсор.

Синтаксис

function GetGroupName(out Name: OleVariant): Integer;

Параметры

Code [out]

Название группы, которое будет возвращено.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция SetGroupCode

Функция назначает код группе, на которую установлен курсор.

Синтаксис

function SetGroupCode(Code: OleVariant): Integer;

Параметры

Code [in]

Код, который будет назначен группе. Значение не должно быть пустым. Допустимыми символами являются: буквы латинского алфавита, цифры и знаки подчеркивания. Максимальный размер – 20 символов.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция SetGroupName

Функция название группе, на которую установлен курсор.

Синтаксис

function SetGroupName(Name: OleVariant): Integer;

Параметры

Name [in]

Название, которое будет назначено группе. Максимальный размер – 200 символов.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция AddGroup

Функция добавляет новую группу и устанавливает на нее курсор.

Синтаксис

function AddGroup(*Code*: OleVariant; *Name*: OleVariant): Integer;

Параметры

Code [in]

Код группы. Значение можно оставить пустым, тогда код будет сформирован автоматически. Допустимыми символами являются: буквы латинского алфавита, цифры и знаки подчеркивания. Максимальный размер – 20 символов.

Name [in]

Название группы. Максимальный размер – 200 символов.

Возвращаемые значения

В случае успешного завершения функция возвращает идентификатор добавленной группы. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция DeleteGroup

Функция используется для удаления группы, на которую установлен курсор.

Синтаксис

function DeleteGroup: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**.

# Функция RefreshGroups

Функция обновляет данные о группах.

Синтаксис

function RefreshGroups: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**.

# Функция FirstField

Функция устанавливает в текущей (на которой установлен курсор) группе курсор на первое поле.

Синтаксис

function FirstField: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

В случае успешного завершения функция возвращает идентификатор первого поля. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция NextField

Функция устанавливает в текущей (на которой установлен курсор) группе курсор на следующее поле.

Синтаксис

function NextField: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

В случае успешного завершения функция возвращает идентификатор следующего поля. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция PrevField

Функция устанавливает в текущей (на которой установлен курсор) группе курсор на предыдущее поле.

Синтаксис

function PrevField: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

В случае успешного завершения функция возвращает идентификатор предыдущего поля. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция LastField

Функция устанавливает в текущей (на которой установлен курсор) группе курсор на последнее поле.

Синтаксис

function LastField: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

В случае успешного завершения функция возвращает идентификатор последнего поля. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция BOFFields

Функция позволяет определить, расположен ли курсор в начале набора полей текущей (на которой установлен курсор) группы.

Синтаксис

function BOFFields: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

Если курсор находится в начале набора полей (в группе отсутствуют поля, была вызвана функция FirstField, была вызвана функция PrevField, когда курсор находился на первом поле), функция возвращает значение 1. Если курсор находится в любом другом месте, функция возвращает значение 0. В случае ошибки функция возвращает -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

Функция обычно используется совместно с функцией PrevField.

См. также

# Connect, PrevField, Навигация по товарам / разделам / группам / полям групп.

# Функция EOFFields

Функция позволяет определить, расположен ли курсор в конце набора полей текущей (на которой установлен курсор) группы.

Синтаксис

function EOFFields: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

Если курсор находится в конце набора полей (в группе отсутствуют поля, была вызвана функция LastField, была вызвана функция NextField, когда курсор находился на последнем поле), функция возвращает значение 1. Если курсор находится в любом другом месте, функция возвращает значение 0. В случае ошибки функция возвращает -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

Функция обычно используется совместно с функцией NextField.

См. также

# Connect, Навигация по товарам / разделам / группам / полям групп.

# Функция FindFieldById

Функция ищет в текущей (на которую установлен курсор) группе поле по идентификатору, и устанавливает на него курсор.

Синтаксис

function FindFieldById(*Id*: Integer): Integer;

Параметры

Id [in]

Идентификатор поля, по которому будет происходить поиск.

Возвращаемые значения

Если поле найдено, функция возвращает значение 1. Если поле не найдено, то функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция FindFieldByName

Функция ищет в текущей (на которую установлен курсор) группе поле по имени, и устанавливает на него курсор.

Синтаксис

function FindFieldByName(*Name*: OleVariant): Integer;

Параметры

Name [in]

Имя поля, по которому будет происходить поиск.

Возвращаемые значения

Если поле найдено, функция возвращает значение 1. Если поле не найдено, то функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция FindFieldByCaption

Функция ищет в текущей (на которую установлен курсор) группе поле по названию, и устанавливает курсор на первое найденное.

Синтаксис

function FindFieldByCaption(*Caption*: OleVariant): Integer;

Параметры

Caption [in]

Название поля, по которому будет происходить поиск.

Возвращаемые значения

Если поле найдено, функция возвращает значение 1. Если поле не найдено, то функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция FindFieldByType

Функция ищет в текущей (на которой установлен курсор) группе поле по типу, и устанавливает курсор на первое найденное.

Синтаксис

function FindFieldByType(*Type\_*: Integer): Integer;

Параметры

Type\_ [in]

Тип поля, по которому будет происходить поиск:

0 – строка;

1 – текст;

3 – целое число;

4 – дробное число;

5 – файл;

6 – картинка;

7 – список с выбором одного значения;

8 – список с выбором нескольких значений;

9 – интернет адрес (url).

Возвращаемые значения

Если поле найдено, функция возвращает значение 1. Если поле не найдено, то функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция FindFieldByPurpose

Функция ищет в текущей (на которой установлен курсор) группе поле по назначению, и устанавливает курсор на первое найденное.

Синтаксис

function FindFieldByPurpose(*Purpose*: Integer): Integer;

Параметры

Purpose[in]

Назначение поля, по которому будет происходить поиск:

0 – Поиск и сравнение;

1 – Назначение 1;

2 – Назначение 2;

3 – Не передается на интернет-сервер.

Возвращаемые значения

Если поле найдено, функция возвращает значение 1. Если поле не найдено, то функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция GetFieldId

Функция используется для получения идентификатора поля, на которое установлен курсор в текущей (на которую установлен курсор) группе.

Синтаксис

function GetFieldId: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

В случае успешного завершения функция возвращает идентификатор поля. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция GetFieldName

Функция используется для получения имени поля, на которое установлен курсор в текущей (на которую установлен курсор) группе.

Синтаксис

function GetFieldName(out *Name*: OleVariant): Integer;

Параметры

Name [out]

Имя поля, которое будет возвращено.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция GetFieldCaption

Функция используется для получения названия поля, на которое установлен курсор в текущей (на которую установлен курсор) группе.

Синтаксис

function GetFieldCaption(out *Caption*: OleVariant): Integer;

Параметры

Caption [out]

Название поля, которое будет возвращено.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция GetFieldType

Функция используется для получения типа поля, на которое установлен курсор в текущей (на которую установлен курсор) группе.

Синтаксис

function GetFieldType(out *Type\_*: Integer): Integer;

Параметры

Type\_ [out]

Тип поля, который будет возвращен.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция GetFieldPurpose

Функция используется для получения назначения поля, на которое установлен курсор в текущей (на которую установлен курсор) группе.

Синтаксис

function GetFieldPurpose(out *Purpose*: Integer): Integer;

Параметры

Purpose [out]

Назначение поля, которое будет возвращено.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция SetFieldName

Функция назначает имя полю, на которое установлен курсор в текущей (на которую установлен курсор) группе.

Синтаксис

function SetFieldName(*Name*: OleVariant): Integer;

Параметры

Name [in]

Имя, которое будет назначено полю. Допустимыми символами являются: буквы латинского алфавита, цифры и знаки подчеркивания. Первым символом должна быть буква латинского алфавита. Значение не должно быть пустым. Максимальный размер – 20 символов. Назначаемое имя не должно совпадать с именами полей в группе “Товар”.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция SetFieldCaption

Функция назначает название полю, на которое установлен курсор в текущей (на которую установлен курсор) группе.

Синтаксис

function SetFieldCaption(*Caption*: OleVariant): Integer;

Параметры

Caption [in]

Название поля, которое будет назначено. Значение не должно быть пустым. Максимальный размер – 200 символов.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция SetFieldType

Функция назначает тип полю, на которое установлен курсор в текущей (на которую установлен курсор) группе.

Синтаксис

function SetFieldType(*Type\_*: Integer): Integer;

Параметры

Type\_ [in]

Тип поля, который будет назначен:

0 – строка;

1 – текст;

3 – целое число;

4 – дробное число;

5 – файл;

6 – картинка;

7 – список с выбором одного значения;

8 – список с выбором нескольких значений;

9 – интернет адрес (url).

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция SetFieldPurpose

Функция назначает назначение полю, на которое установлен курсор в текущей группе (на которую установлен курсор).

Синтаксис

function SetFieldPurpose(*Purpose*: Integer): Integer;

Параметры

Purpose [in]

Назначение поля, которое будет назначено:

0 – Поиск и сравнение;

1 – Назначение 1;

2 – Назначение 2;

3 – Не передается на интернет-сервер.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция AddField

Функция добавляет в текущую (на которую установлен курсор) группу новое поле и устанавливает на него курсор.

Синтаксис

function AddField(*Name*: OleVariant; *Caption*: OleVariant; *Type\_*: Integer; *Purpose*: Integer): Integer;

Параметры

Name [in]

Имя добавляемого поля. Допустимыми символами являются: буквы латинского алфавита, цифры и знаки подчеркивания. Первым символом должна быть буква латинского алфавита. Значение не должно быть пустым. Максимальный размер – 20 символов. Назначаемое имя не должно совпадать с именами полей в группе “Товар”.

Caption [in]

Название добавляемого поля. Значение не должно быть пустым. Максимальный размер – 200 символов.

Type\_ [in]

Тип добавляемого поля:

0 – строка;

1 – текст;

3 – целое число;

4 – дробное число;

5 – файл;

6 – картинка;

7 – список с выбором одного значения;

8 – список с выбором нескольких значений;

9 – интернет адрес (url).

Purpose [in]

Назначение добавляемого поля:

0 – Поиск и сравнение;

1 – Назначение 1;

2 – Назначение 2;

3 – Не передается на интернет-сервер.

Возвращаемые значения

В случае успешного завершения функция возвращает идентификатор добавленного поля. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

На момент вызова функции магазин не должен использоваться другими пользователями.

См. также

**Connect**, **Навигация по товарам / разделам / группам / полям групп**.

# Функция DeleteField

Функция удаляет в текущей (на которую установлен курсор) группе поле, на которое установлен курсор.

Синтаксис

function DeleteField: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**.

# Функция RefreshFields

Функция обновляет данные о полях в текущей (на которой установлен курсор) группе.

Синтаксис

function RefreshFields: Integer;

Параметры

Функция не имеет параметров.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**.

# Функция SetFieldQueue

Функция позволяет указать позицию поля в группе относительно других сортируемых полей группы при написании собственной сортировки.

Синтаксис

function SetFieldQueue(*FieldId*: Integer; *Queue*: Integer): Integer;

Параметры

*FieldId*[in]

Идентификатор сортируемого поля группы. Поле группы должно существовать.

*Queue*[in]

Позиция поля группы по отношению к другим сортируемым полям группы. Позиция должна задаваться с определенными промежутками. Например, первое поле группы – Queue = 0, второе поле группы – Queue = 16384, и так далее.

Возвращаемые значения

В случае успешного завершения функция возвращает значение 0. В случае ошибки функция возвращает значение -1. Для получения информации об ошибке используйте функции **GetLastError**, **GetLastErrorStr**.

Заметки

Чтобы использовать функцию, необходимо быть подключенным к WinShop-серверу.

См. также

**Connect**.

# Пример для Delphi 7

Для начала работы необходимо, чтобы на компьютере был установлен WinShop (Тип установки “Главный компьютер” или “Дополнительный” значения не имеет).

Запускаем среду разработки Delphi 7 и импортируем библиотеку типов. Для этого выбираем в главном меню команду “Project” -> “Import Type Library” (Рис. 01).

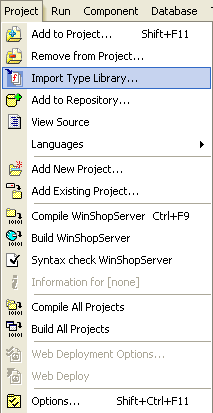


Рис. 01

В открывшемся окне, в списке находим и выделяем строку “WinShopCOM (Version 1.0)” и нажимаем кнопку “Install” (Рис. 02).

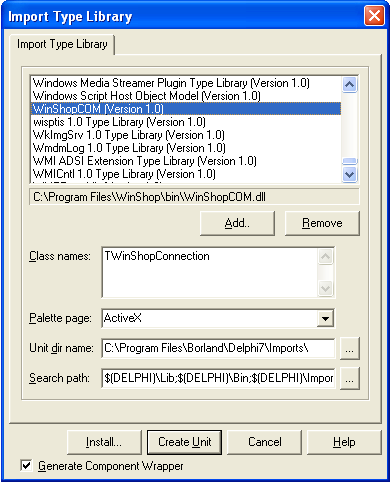


Рис. 02

После этого на закладке компонентов “ActiveX” появится компонент с именем WinShopConnection (Рис. 03).

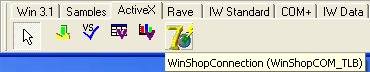


Рис. 03

Создаем новый проект. На форму помещаем компонент WinShopConnection и 6 компонентов-кнопок с именами (Name): “ConnectButton”, “DisconnectButton”, “NumberOfProductsButton”, “ClearPriceForAllProductsButton”, “NumberOfFieldsInGroupsButton”, “SortProducts”, “CustomSortProducts”. В качестве заголовков (Caption) соответственно укажем: “Connect”, “Disconnect”, “Number of products”, “Clear price for all products”, “Number of fields of groups, “Sort products”, “Custom sort products”.

Для кнопок создадим обработчики события “OnClick” (исходный код приведен ниже). По нажатию на кнопку “Connect” будет выполнено подключение к магазину “Магазин-пример”. Нажав кнопку “Disconnect” произойдет отключение от магазина. После нажатия на кнопку “Number of products” будет подсчитано общее количество товаров в магазине, а также количество товаром, имеющих статус “В резерве”. Нажав кнопку “Clear price for all products” для всех товаров магазина, имеющих статус “В резерве” будет очищена цена (установлена равной 0). По нажатию кнопки “Number of fields in groups”, будет выведено количество полей в каждой группе магазина. После нажатия на кнопку “Sort products” товары главной страницы будут отсортированы по полям: “цена” – по убыванию, “краткое название” – по возрастанию, “статус” – по возрастанию с использованием встроенной сортировки. Нажав кнопку “Custom sort products” будет выполнена пользовательская сортировка товаров по цене.

unit ExampleUnit;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, OleServer, WinShopCOM\_TLB, StdCtrls;

type

TExampleForm = class(TForm)

ClearPriceForAllProductsButton: TButton;

ConnectButton: TButton;

DisconnectButton: TButton;

NumberOfFieldsInGroupsButton: TButton;

NumberOfProductsButton: TButton;

WinShopConnection1: TWinShopConnection;

SortProducts: TButton;

CustomSortProducts: TButton;

procedure ClearPriceForAllProductsButtonClick(Sender: TObject);

procedure ConnectButtonClick(Sender: TObject);

procedure CustomSortProductsClick(Sender: TObject);

procedure DisconnectButtonClick(Sender: TObject);

procedure NumberOfFieldsInGroupsButtonClick(Sender: TObject);

procedure NumberOfProductsButtonClick(Sender: TObject);

procedure SortProductsClick(Sender: TObject);

private

function CheckResult(Value: Integer): Integer;

public

{ Public declarations }

end;

var

ExampleForm: TExampleForm;

implementation

{$R \*.dfm}

function TExampleForm.CheckResult(Value: Integer): Integer;

begin

Result := Value;

if Result = -1 then

raise Exception.Create(WinShopConnection1.GetLastErrorStr);

end;

procedure TExampleForm.ClearPriceForAllProductsButtonClick(Sender: TObject);

var

FoundProducts, ModifiedProducts: Integer;

begin

FoundProducts := CheckResult(WinShopConnection1.FindByField('status', 0));

ModifiedProducts := 0;

if FoundProducts > 0 then

ModifiedProducts := CheckResult(WinShopConnection1.SetValue('price', 0));

ShowMessageFmt('Изменено товаров: %d', [ModifiedProducts]);

end;

procedure TExampleForm.ConnectButtonClick(Sender: TObject);

begin

CheckResult(WinShopConnection1.Connect1('', 0, 'Administrator', '',

'Магазин-пример'));

end;

procedure TExampleForm.CustomSortProductsClick(Sender: TObject);

type

TProduct = record

Id: Integer;

Price: Currency;

end;

TProducts = array of TProduct;

var

Product: TProduct;

ProductId, I, J: Integer;

Products: TProducts;

begin

SetLength(Products, 0);

ProductId := CheckResult(WinShopConnection1.First);

if ProductId > 0 then

while ProductId > 0 do

begin

if WinShopConnection1.GetValueStr('primer\_id') = 2 then

begin

Product.Id := ProductId;

Product.Price :=

StrToFloat(StringReplace(WinShopConnection1.GetValueStr('price'), '.', ',', []));

SetLength(Products, Length(Products) + 1);

Products[Length(Products) - 1] := Product;

end;

ProductId := CheckResult(WinShopConnection1.Next);

end;

for I := 0 to Length(Products) - 2 do

for J := 0 to Length(Products) - 2 - I do

if Products[J].Price > Products[J + 1].Price then

begin

Product := Products[J];

Products[J] := Products[J + 1];

Products[J + 1] := Product;

end;

CheckResult(WinShopConnection1.FindById2(2, 2));

for I := 0 to Length(Products) - 2 do

CheckResult(WinShopConnection1.SetQueue(Products[I].Id, 1, I \* 16384));

end;

procedure TExampleForm.DisconnectButtonClick(Sender: TObject);

begin

CheckResult(WinShopConnection1.Disconnect1);

end;

procedure TExampleForm.NumberOfFieldsInGroupsButtonClick(Sender: TObject);

var

GroupCode, GroupName: OleVariant;

NumberOfFields: Integer;

begin

CheckResult(WinShopConnection1.FirstGroup);

while CheckResult(WinShopConnection1.EOFGroups) <> 1 do

begin

CheckResult(WinShopConnection1.GetGroupCode(GroupCode));

CheckResult(WinShopConnection1.GetGroupName(GroupName));

NumberOfFields := 0;

CheckResult(WinShopConnection1.FirstField);

while CheckResult(WinShopConnection1.EOFFields) <> 1 do

begin

Inc(NumberOfFields);

CheckResult(WinShopConnection1.NextField);

end;

ShowMessageFmt('Код группы: %s; Имя группы: %s; Количество полей: %d',

[GroupCode, GroupName, NumberOfFields]);

CheckResult(WinShopConnection1.NextGroup);

end;

end;

procedure TExampleForm.NumberOfProductsButtonClick(Sender: TObject);

var

ProductId, AllProducts, ProductsInReserve: Integer;

begin

AllProducts := 0;

ProductsInReserve := 0;

ProductId := CheckResult(WinShopConnection1.First);

while ProductId > 0 do

begin

Inc(AllProducts);

if CheckResult(WinShopConnection1.GetValueStr('status')) = 0 then

Inc(ProductsInReserve);

ProductId := CheckResult(WinShopConnection1.Next);

end;

ShowMessageFmt('Всего товаров: %d (в резерве: %d)',

[AllProducts, ProductsInReserve]);

end;

procedure TExampleForm.SortProductsClick(Sender: TObject);

begin

CheckResult(WinShopConnection1.Sort(1, 'price', False, 'name', True, 'status', True, 1));

end;

end.

# Пример для 1C: Предприятие 7.7

Рассмотрим использование функций COM-сервера на примере синхронизации товаров магазина WinShop со справочником товаров 1C: Предприятие 7.7.

Запускаем программу 1C: Предприятие 7.7. В окне “Запуск 1C:Предприятия” из выпадающего списка “В режиме:” выбираем "Конфигуратор", нажимаем кнопку “OK” (Рис. 01).

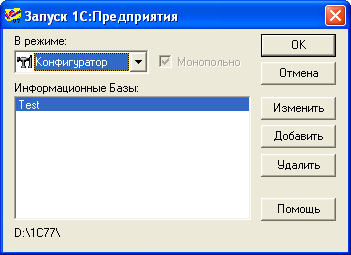


Рис. 01

Для создания справочника товаров в окне “Конфигуратор – задача” в меню “Конструкторы” выбираем пункт “Новый справочник...” (Рис. 02) или нажимаем кнопку “Новый справочник” на панели инструментов “Конструкторы” (Рис. 03).

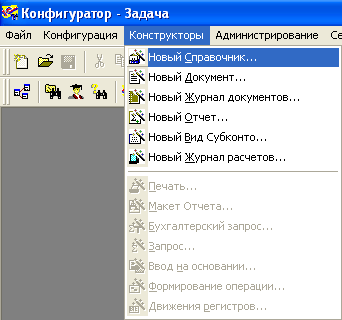


Рис. 02



Рис. 03

В окне “Конструктор справочника” в поле “Идентификатор” укажем - "Товары", нажимаем кнопку “Далее” (Рис. 04).

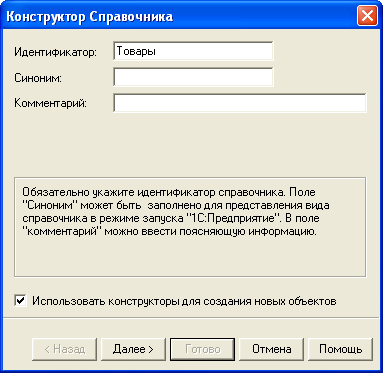


Рис. 04

Нажимаем кнопку “Далее” (Рис. 05).

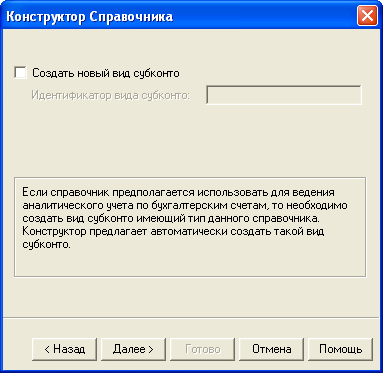


Рис. 05

Нажимаем кнопку “Готово” (Рис. 06).

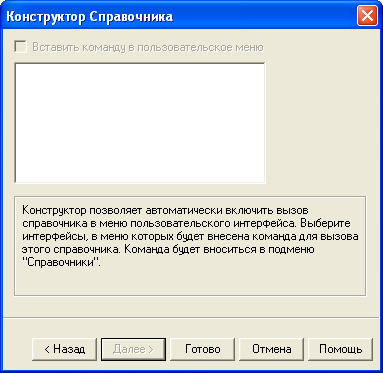


Рис. 06

Для добавления реквизита “ЦенаСНДС” в список “Реквизиты” в окне “Справочник товары” нажимаем кнопку “Новый” (Рис. 07).

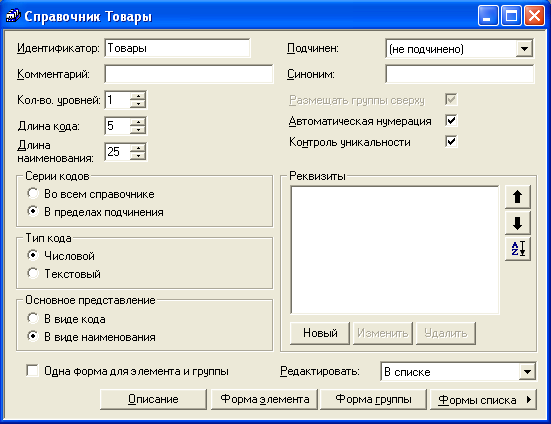


Рис. 07

В окне “Свойства реквизита” на вкладке “Общие” в поле “Идентификатор” укажем - "ЦенаСНДС", из выпадающего списка “Тип значения” выберем “<<Число>>”, нажимаем кнопку “OK” (Рис. 08).

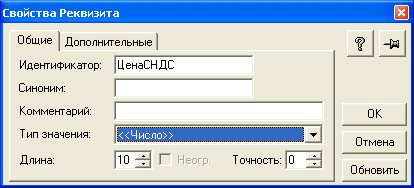


Рис. 08

Таким же образом добавляем реквизиты “ОптНадб” и “Количество” (Рис. 09, Рис. 10).

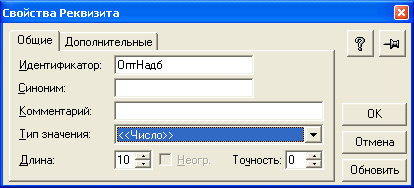


Рис. 09

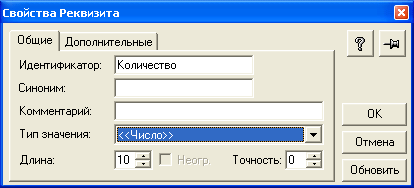


Рис. 10

В окне “Размещение реквизитов справочника” нажимаем кнопку “Вставить” (Рис. 11).

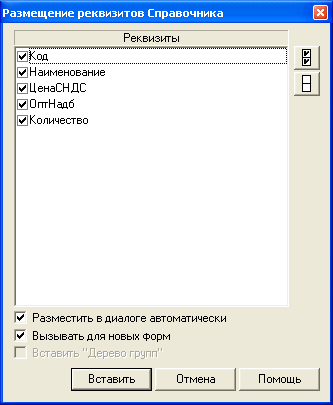


Рис. 11

Из панели инструментов “Элементы диалога” (Рис. 12) поместим в окно “Форма-Справочник.Товары.ФормаСписка.ФормаСписка” элемент диалога “Текст”. В окне “Свойства текста” в поле “Заголовок” укажем - "Имя компьютера", нажимаем. кнопку “OK” (Рис. 13).



Рис. 12

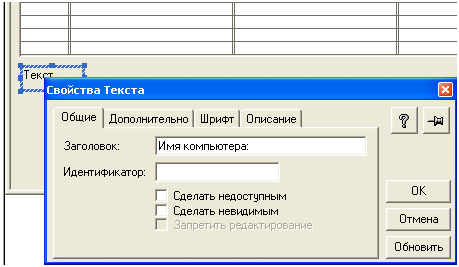


Рис. 13

Таким же образом добавляем 4 элемента диалога “Текст” с заголовками "Номер порта" (Рис. 14), "Имя пользователя" (Рис. 15), "Пароль" (Рис. 16) и "Название магазина" (Рис. 17).

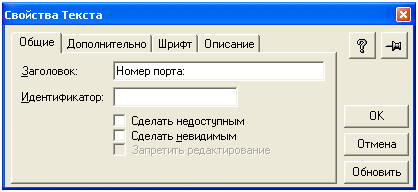


Рис. 14

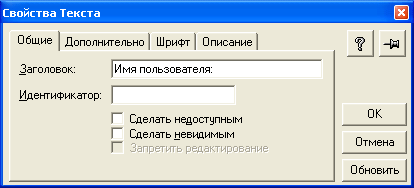


Рис. 15

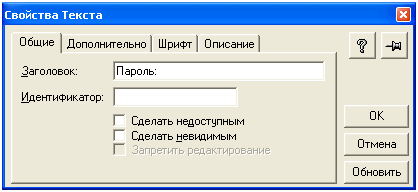


Рис. 16

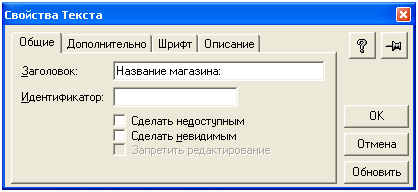


Рис. 17

Из панели инструментов “Элементы диалога” (Рис. 12) поместим в окно “Форма-Справочник.Товары.ФормаСписка.ФормаСписка” элемент диалога “Реквизит диалога”. В окне “Свойства поля ввода” на вкладке “Общие” в поле “Идентификатор” укажем - "ИмяКомпьютера" (Рис. 18), на вкладке “Тип” из выпадающего списка “Тип” выберем “<<Строка>>”, в поле “Длина” укажем - 255, нажимаем кнопку “OK” (Рис. 19).

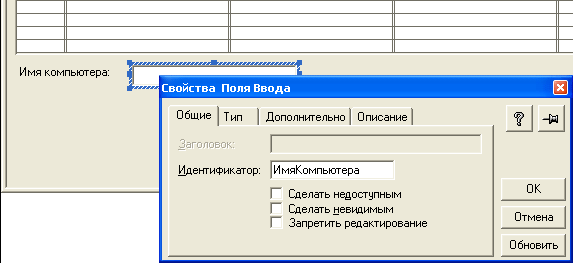


Рис. 18

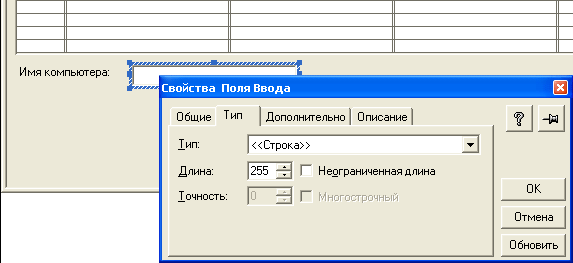


Рис. 19

Таким же образом добавляем 4 элемента диалога “Реквизит диалога” с идентификаторами “НомерПорта” (Рис. 20, Рис. 21), “ИмяПользователя” (Рис. 22, Рис. 23), “Пароль” (Рис. 24, Рис. 25) и “НазваниеМагазина” (Рис. 26, Рис. 27).

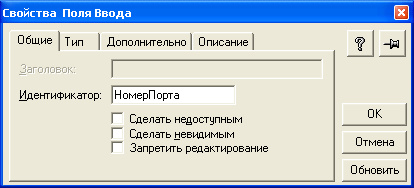


Рис. 20

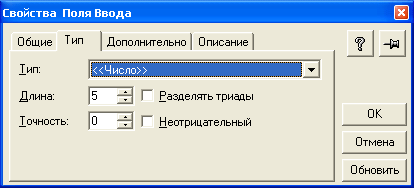


Рис. 21

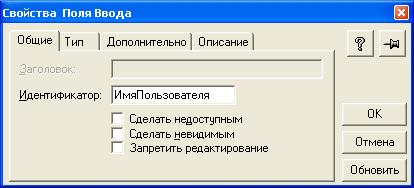


Рис. 22

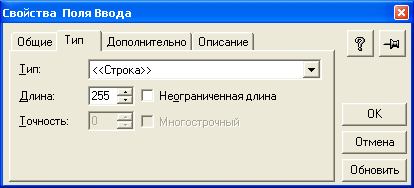


Рис. 23

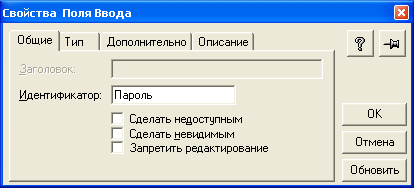


Рис. 24

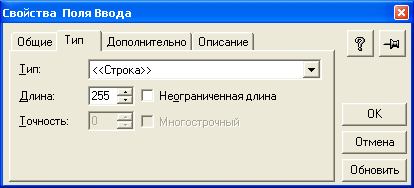


Рис. 25

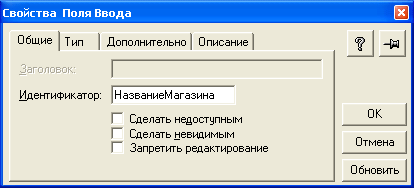


Рис. 26

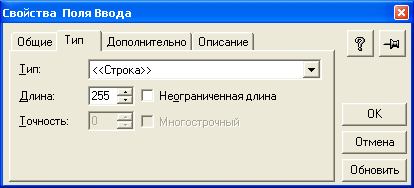


Рис. 27

Из панели инструментов “Элементы диалога” (Рис. 12) поместим в окно “Форма-Справочник.Товары.ФормаСписка.ФормаСписка” элемент диалога “Кнопка”. В окне “Свойства кнопки” на вкладке “Общие” в поле “Заголовок” укажем - "СинхрСВинШоп" (Рис. 28), на вкладке “Дополнительно” в поле “Формула” зададим - "СинхрСВинШоп()", нажимаем кнопку “OK” (Рис. 29).

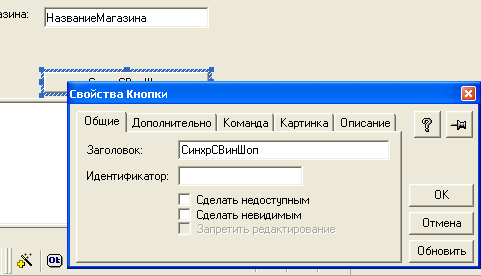


Рис. 28

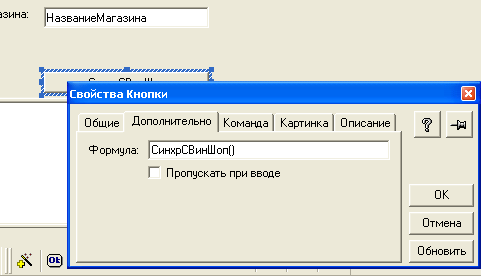


Рис. 29

В окне “Форма-Справочник.Товары.ФормаСписка.ФормаСписка” переходим на вкладку “Модуль”, вставляем нижеследующий исходный код (Рис. 30);

Процедура СинхрСВинШоп();

Товары = СоздатьОбъект("Справочник.Товары");

ВинШоп = СоздатьОбъект("WinShopCOM.WinShopConnection");

Результат = ВинШоп.Connect(СокрЛП(ИмяКомпьютера), НомерПорта, СокрЛП(ИмяПользователя),

СокрЛп(Пароль), СокрЛП(НазваниеМагазина));

Если Результат = -1 Тогда

Сообщить("Не удалось подключиться к ВинШоп");

Возврат;

КонецЕсли;

Товары.ВыбратьЭлементы();

Пока Товары.ПолучитьЭлемент() = 1 Цикл

Результат = ВинШоп.FindByField("Name", Товары.Наименование);

Если Результат <> -1 Тогда

Если Результат = 0 Тогда

Результат = ВинШоп.AddItem(1);

Если Результат = -1 Тогда

Сообщить("Не удалось добавить товар в ВинШоп");

Продолжить;

КонецЕсли;

КонецЕсли;

ВинШоп.SetValue("Name", Товары.Наименование);

ВинШоп.SetValue("Price", Товары.ЦенаСНДС);

Если Товары.Количество <> 0 Тогда

ВинШоп.SetValue("Status", 1);

Иначе

ВинШоп.SetValue("Status", 2);

КонецЕсли;

КонецЕсли;

КонецЦикла;

ВинШоп.Disconnect();

КонецПроцедуры

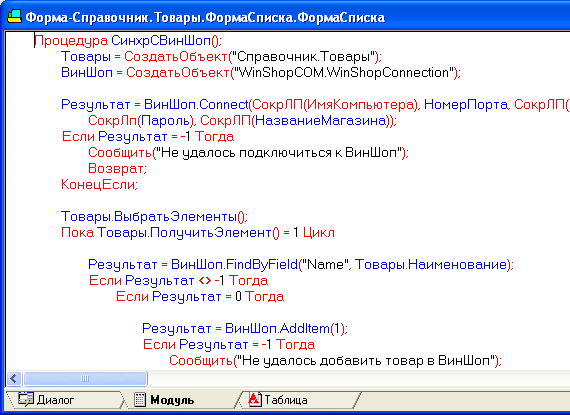


Рис. 30

Для сохранения метаданных в окне “Конфигуратор – задача” в меню “Файл” выбираем пункт “Сохранить” (Рис. 31) или нажимаем кнопку “Сохранить” на панели инструментов ”Стандартная” (Рис. 32).

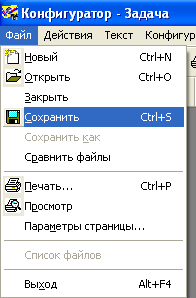


Рис. 31



Рис.32.

В окне сообщения “Конфигуратор” нажимаем кнопку “Да” (Рис. 33).

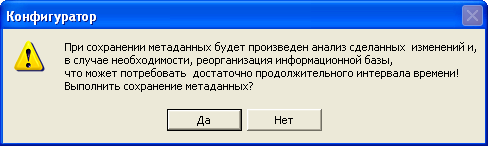


Рис. 33

В окне “Реорганизация информации” нажимаем кнопку “Принять” (Рис. 34)

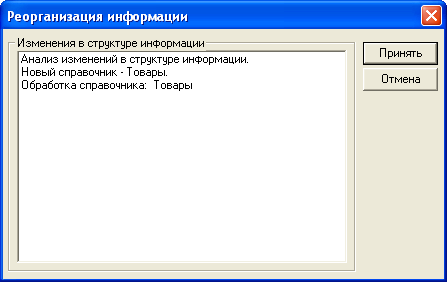


Рис. 34

В окне сообщения “Конфигуратор” нажимаем кнопку “OK” (Рис. 35).

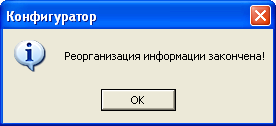


Рис. 35

Для перехода в режим “1C: Предприятие” в окне “Конфигуратор – задача” в меню “Сервис” выбираем пункт “1C: Предприятие” (Рис. 36) или нажимаем кнопку “1C: Предприятие” на панели инструментов “Конфигурация” (Рис. 37).

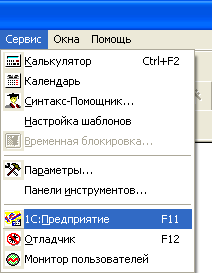


Рис. 36



Рис. 37

В окне “1C: Предприятие – задача” в меню “Операции” выбираем пункт “Справочники…” (Рис. 38).

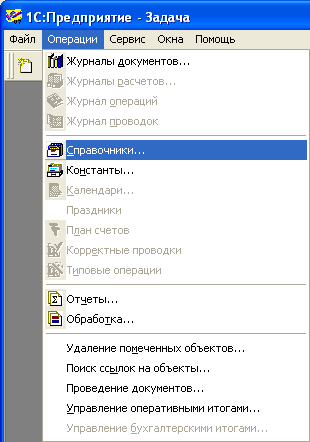


Рис. 38

В окне “Выбор справочника” выбираем – “Товары” и нажимаем кнопку “OK” (Рис. 39).

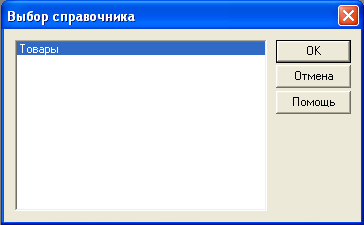


Рис. 39

Заполняем справочник “Товары”, указываем магазин, с которым будем синхронизироваться, нажимаем кнопку “СинхрСВинШоп” (Рис. 40).

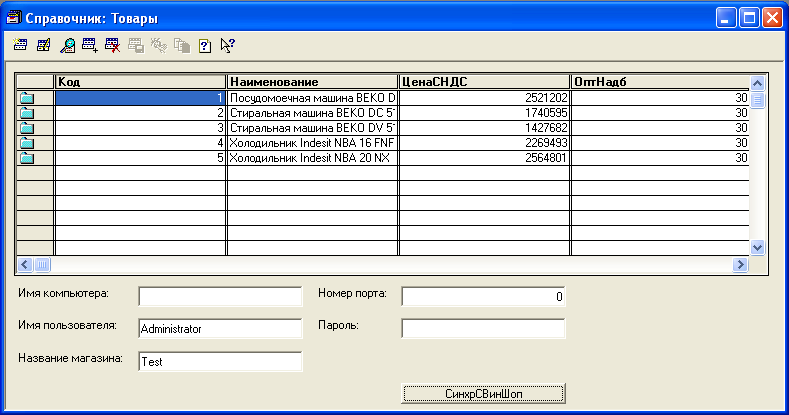


Рис. 40

# Пример для 1C: Предприятие 8.2

Рассмотрим использование WinShop COM-сервера на примере импорта товаров из справочника 1С: Предприятие 8.2 в магазин WinShop.

Для начала, создадим справочник, в котором будут храниться товары. Запускаем информационную базу в режиме конфигуратора (рис. 1).

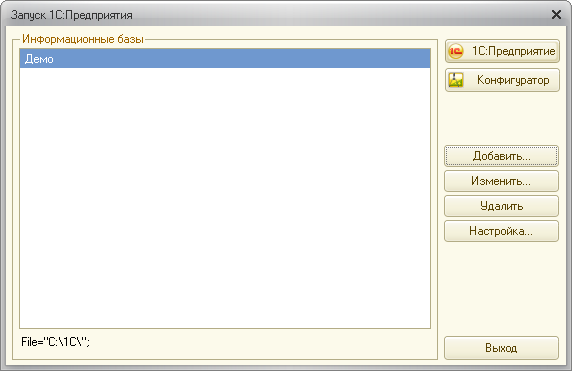


Рис. 1

Открываем дерево конфигурации (рис. 2).

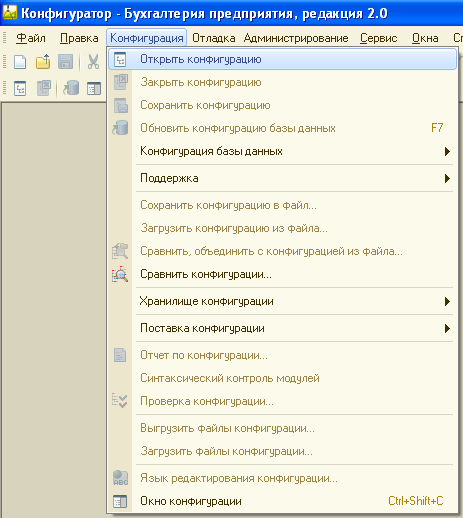


Рис. 2

Добавляем новый справочник (рис. 3).

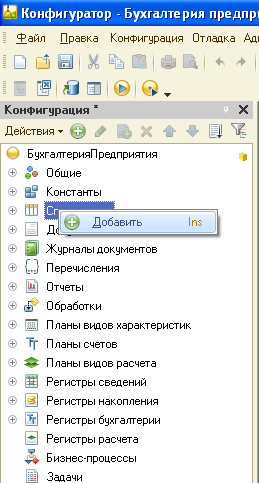


Рис. 3

На вкладке **Основные** укажем: **Имя** — Товары, **Синоним** — Товары (рис. 4).

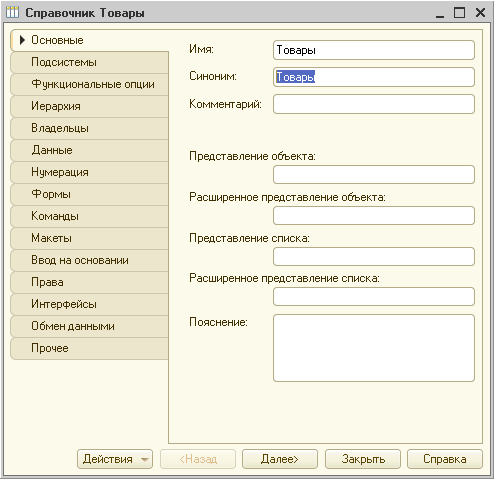


Рис. 4

Переходим на вкладку **Данные** и задаем: **Длина наименования** — 150, **Основное представление** — **В виде кода** (рис. 5).

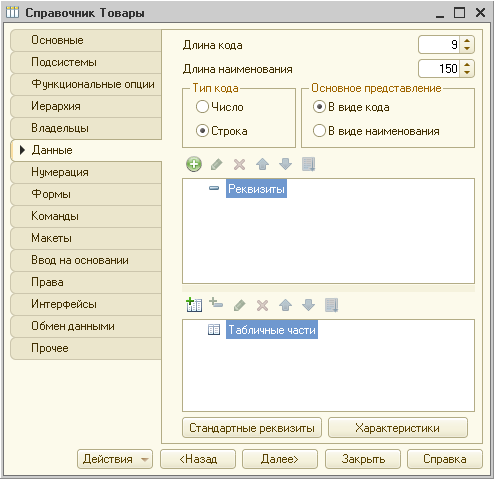


Рис. 5

Для стандартного реквизита **Код** установим: **Синоним** — Артикул, **Проверка заполнения** — **Выдавать ошибку** (рис. 6).

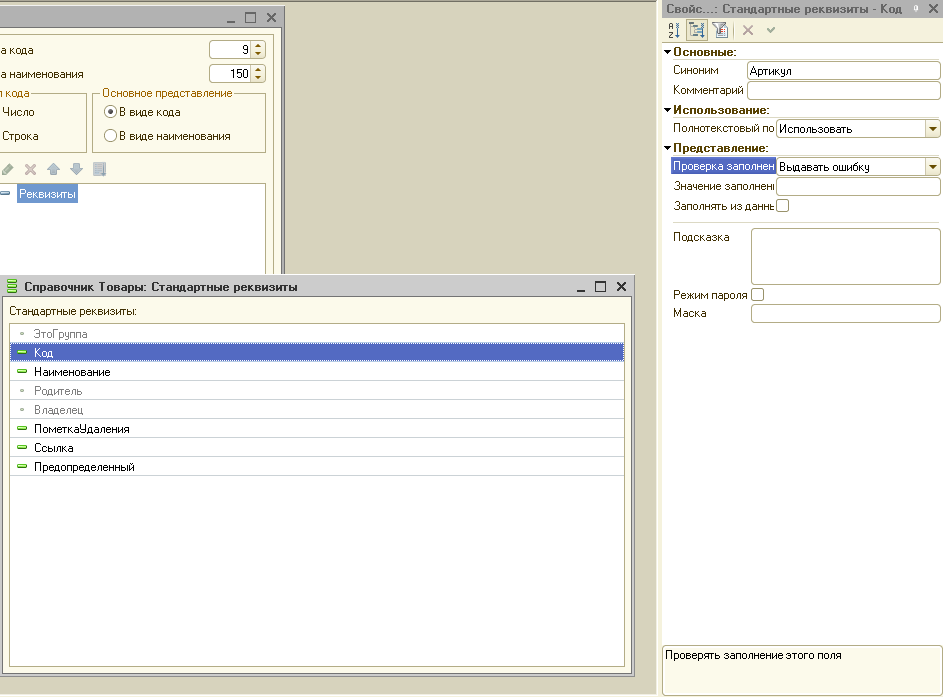


Рис. 6

Для стандартного реквизита **Наименование** укажем:**Синоним** — Название (рис. 7).

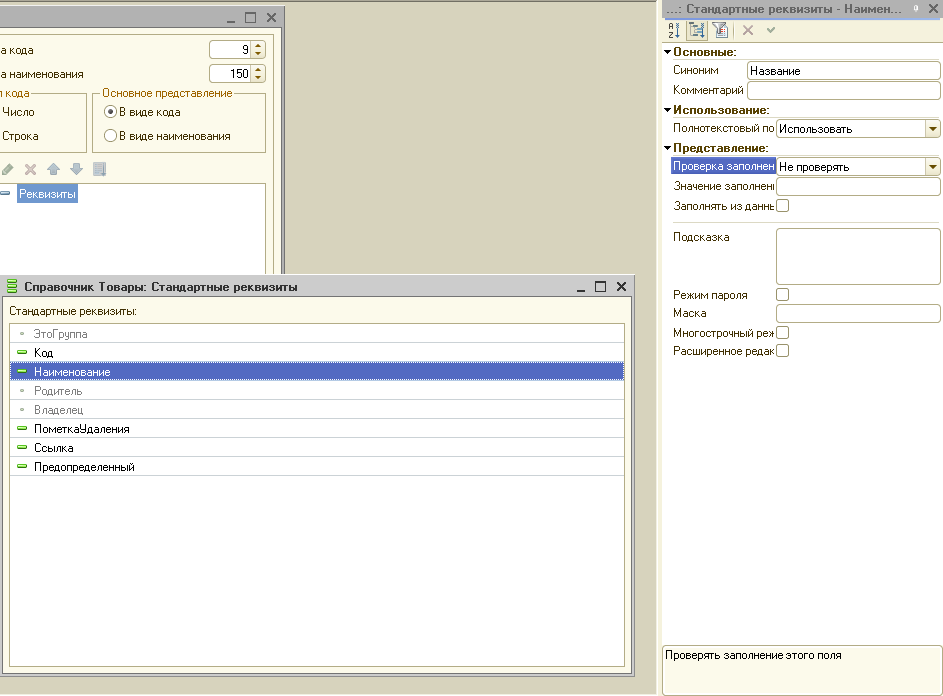


Рис. 7

Добавляем новый реквизит: **Имя** — Доступность, **Синоним** — Доступность, **Тип** — **Строка**, **Длина** — 25 (рис. 8).

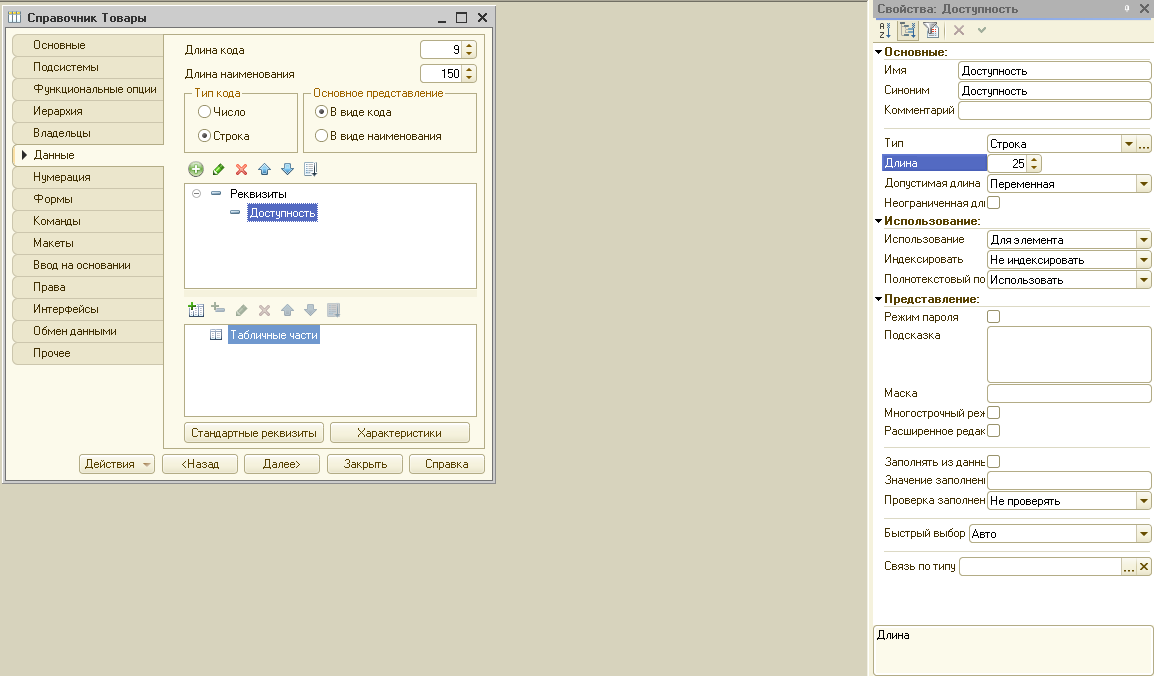


Рис. 8

Добавляем новый реквизит: **Имя** — Цена, **Синоним** — Цена, **Тип** — **Число**, **Формат** — **ЧРД=,** **Формат редактирования** —**ЧРД=,** (рис. 9).

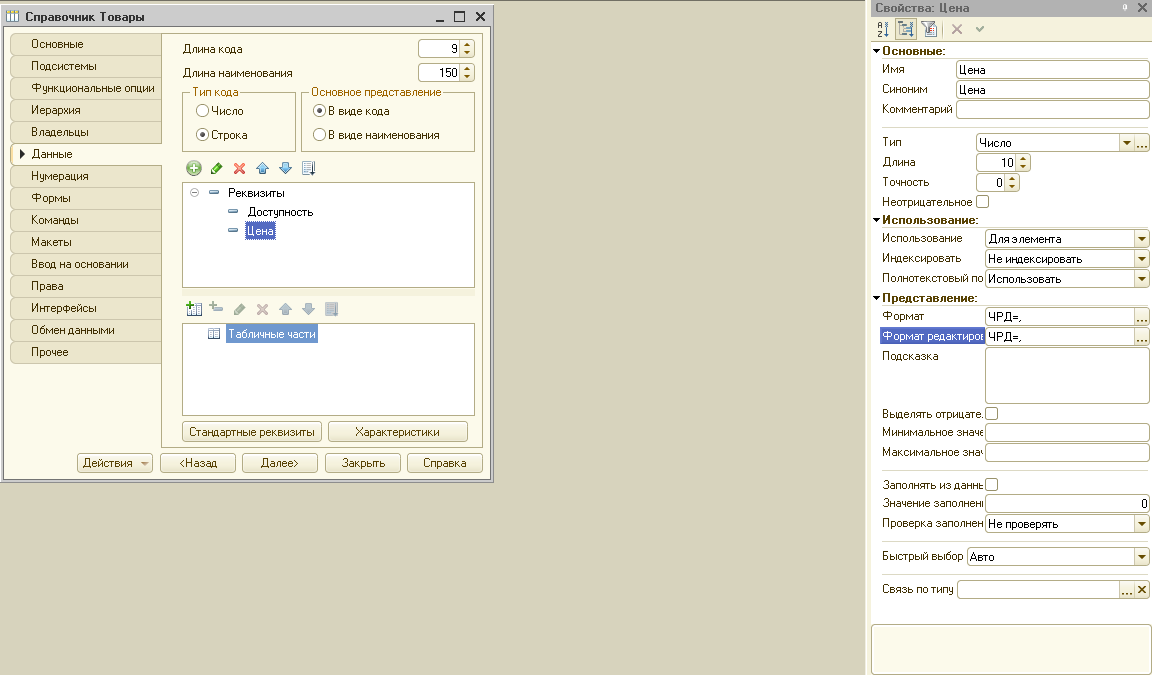


Рис. 9

Добавляем новый реквизит: **Имя** *—* АдресСтраницы, **Синоним** — Адрес страницы, **Тип** — **Строка**, **Длина** — 255 (рис. 10).

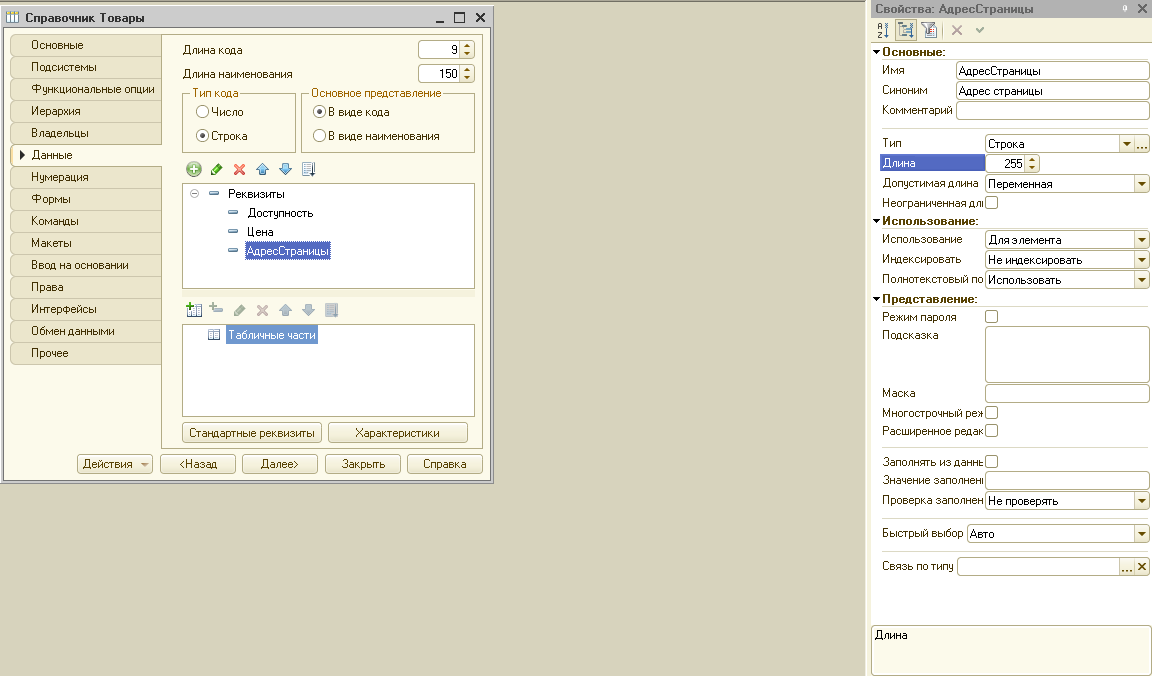


Рис. 10

Переходим на вкладку **Формы** и добавляем новую форму: **Выберите тип формы** — **Форма списка справочника** (рис. 11).

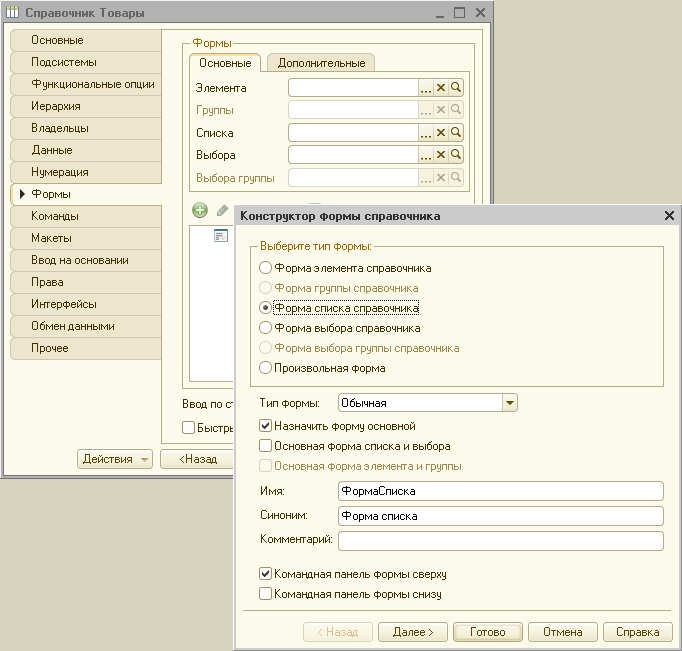


Рис. 11

Отмечаем реквизиты: **Доступность**, **Цена**, **АдресСтраницы** (рис. 12).

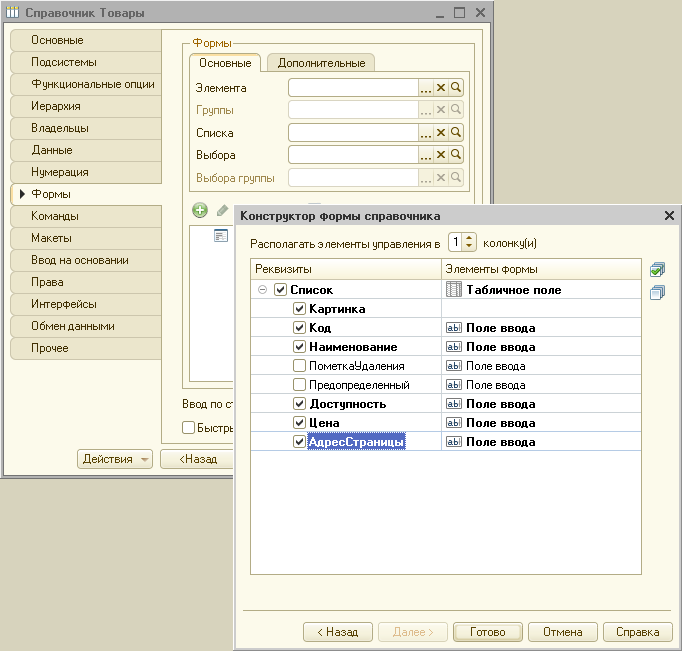


Рис. 12

Добавляем на форму надпись: **Имя** — НадписьНазваниеМагазина, **Заголовок** — Название магазина: (рис. 13).

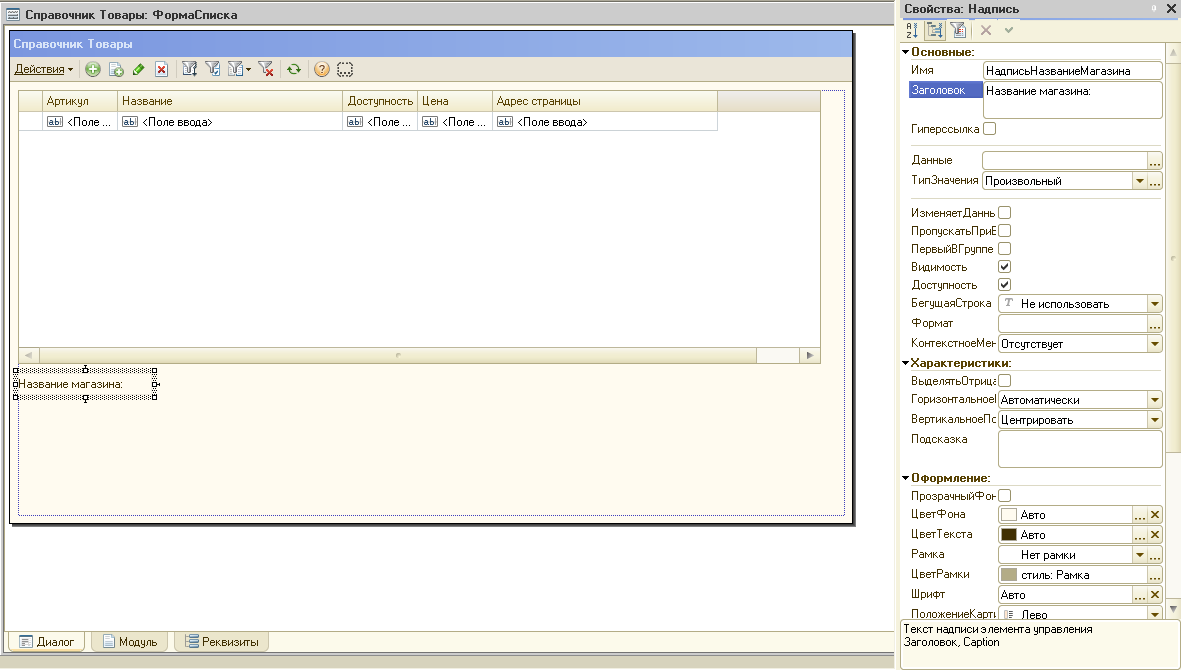


Рис. 13

Добавляем поле ввода: **Имя** — ПолеВводаНазваниеМагазина (рис. 14).

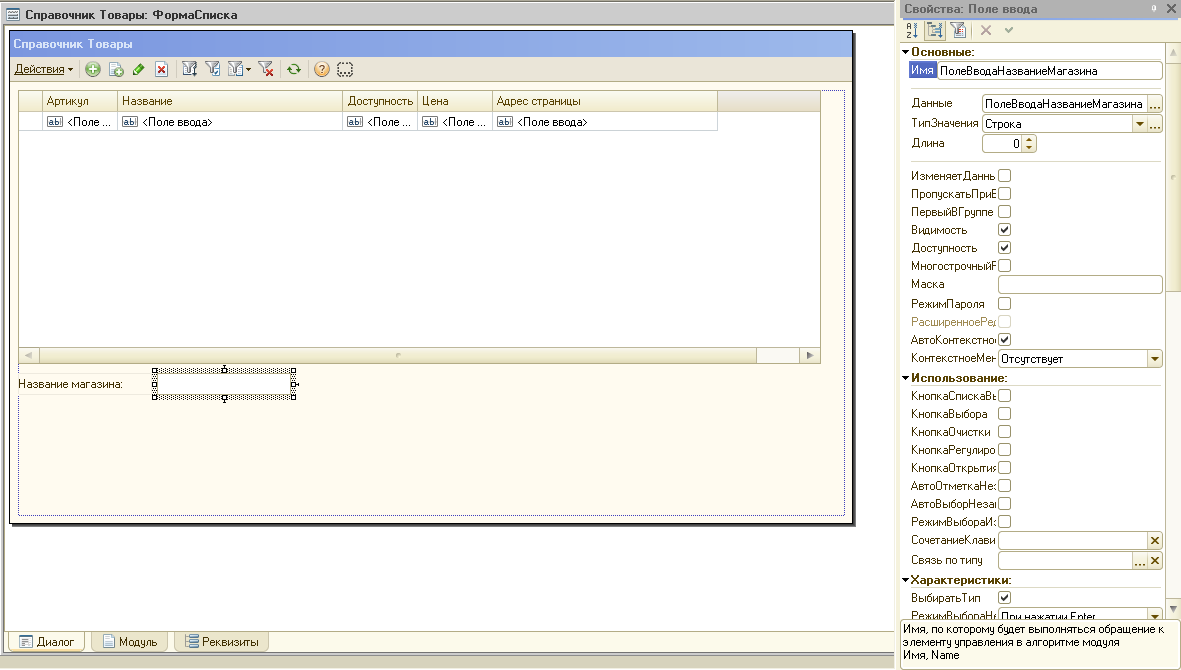


Рис. 14

Добавляем надпись: **Имя** — НадписьИмяПользователя, **Заголовок** — Имя пользователя: (рис. 15).

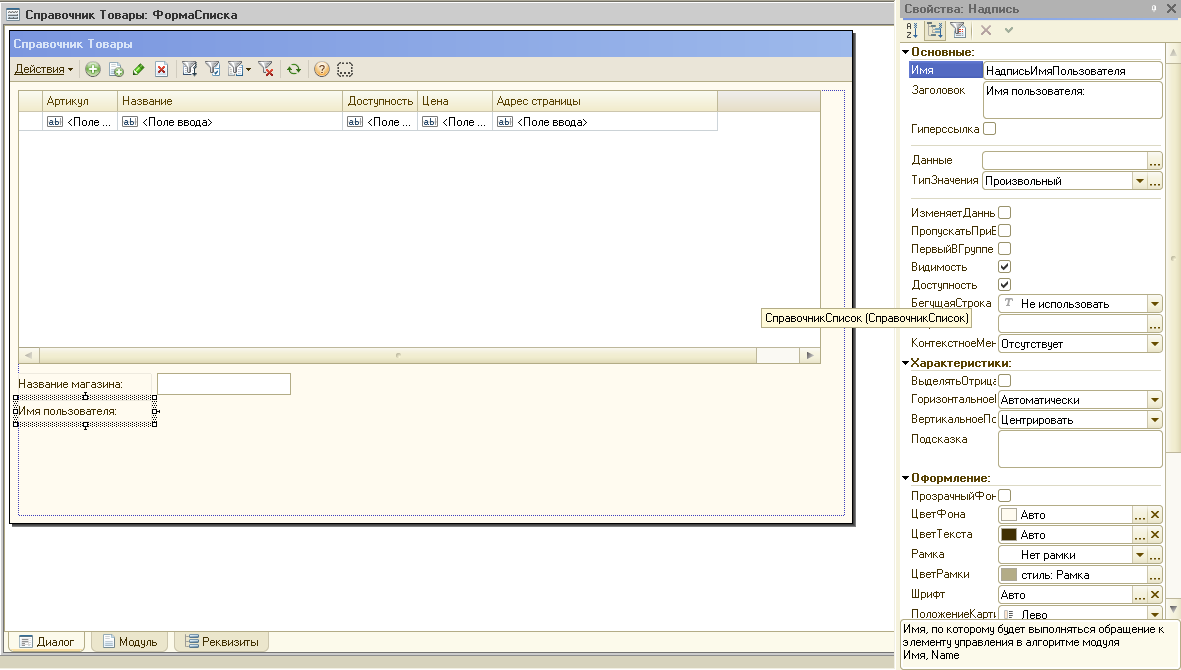


Рис. 15

Добавляем поле ввода: **Имя** — ПолеВводаИмяПользователя (рис. 16).

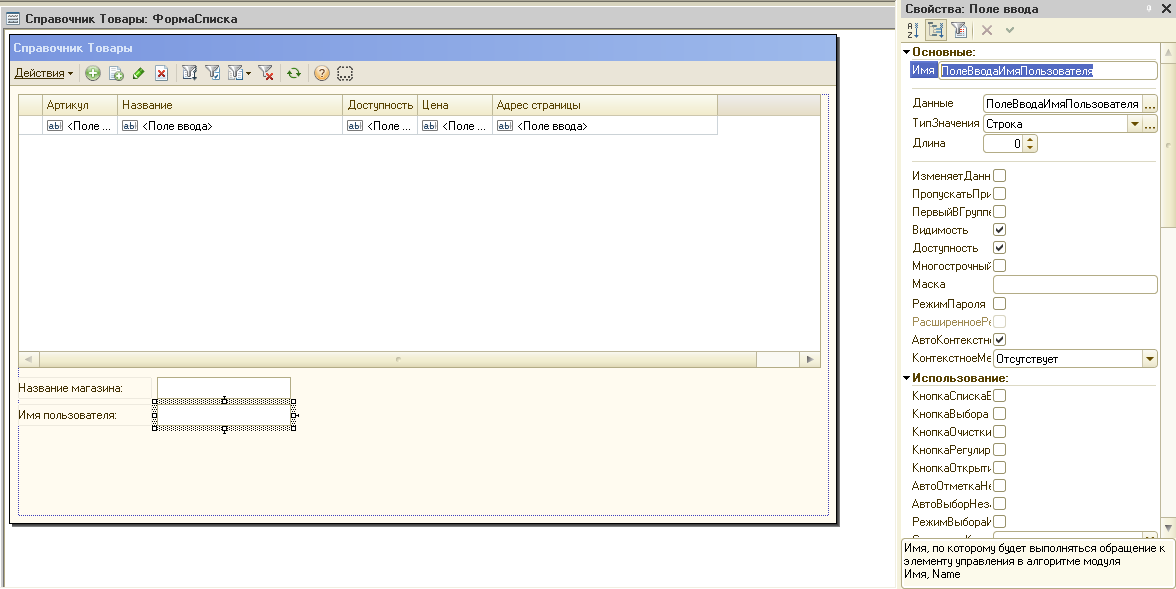


Рис. 16

Добавляем надпись: **Имя** — НадписьПарольПользователя, **Заголовок** — Пароль пользователя: (рис. 17).

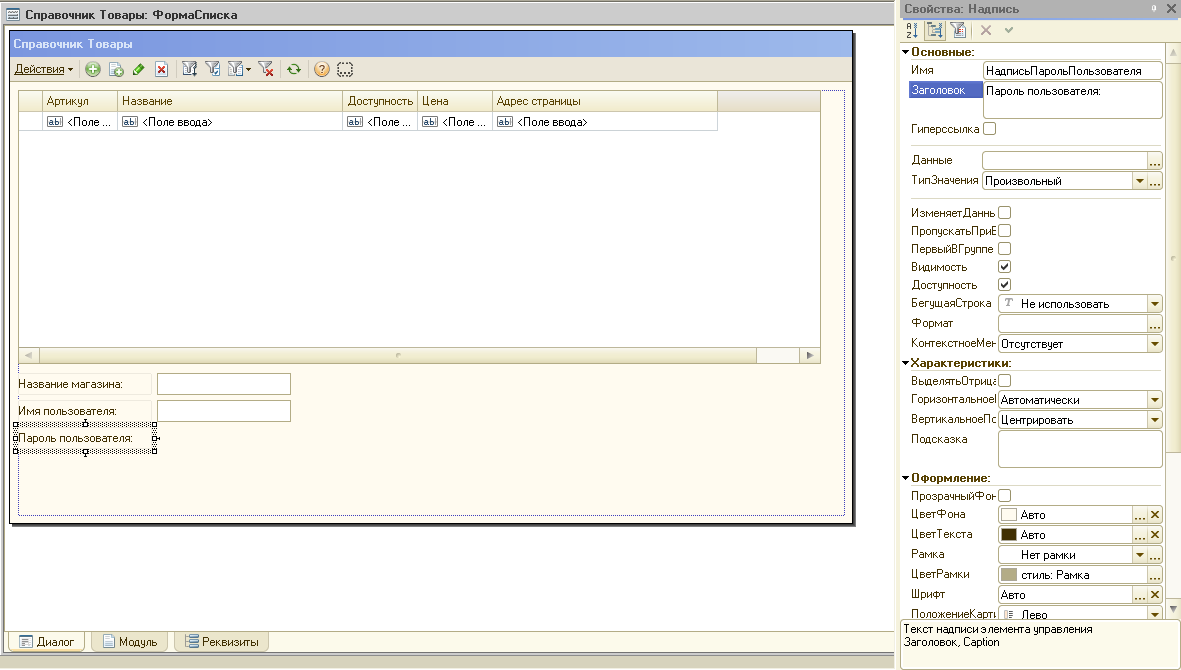


Рис. 17

Добавляем поле ввода: **Имя** — ПолеВводаПарольПользователя (рис. 18).

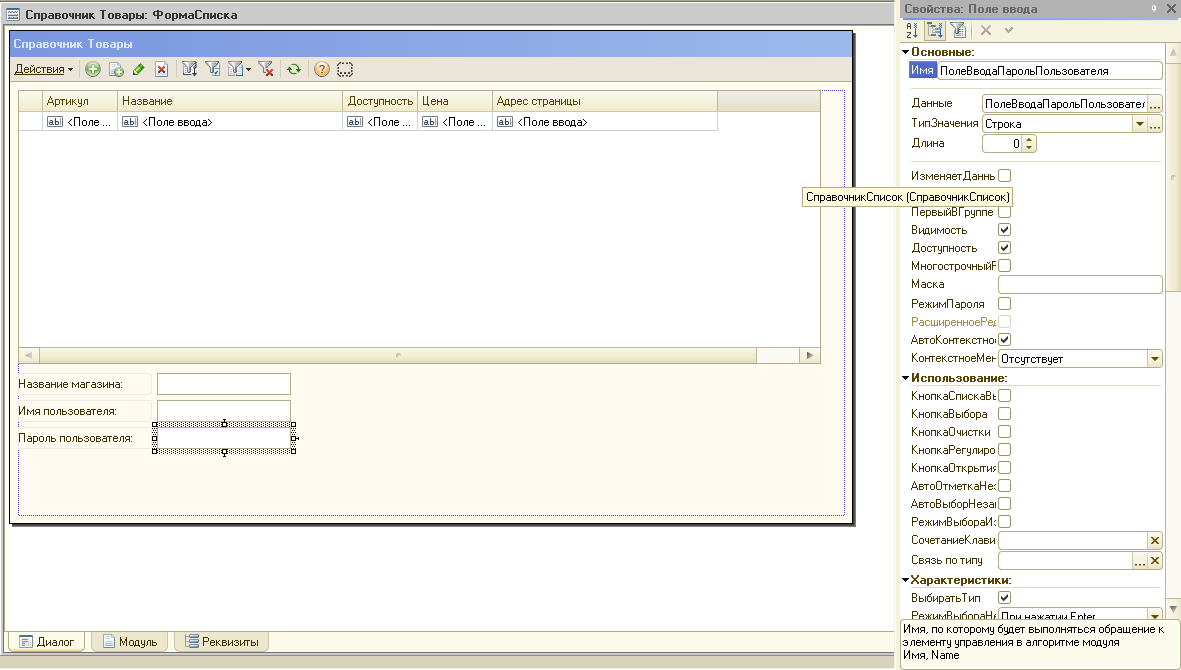


Рис. 18

Добавляем надпись: **Имя** — НадписьИмяХоста, **Заголовок** — Имя хоста: (рис. 19).

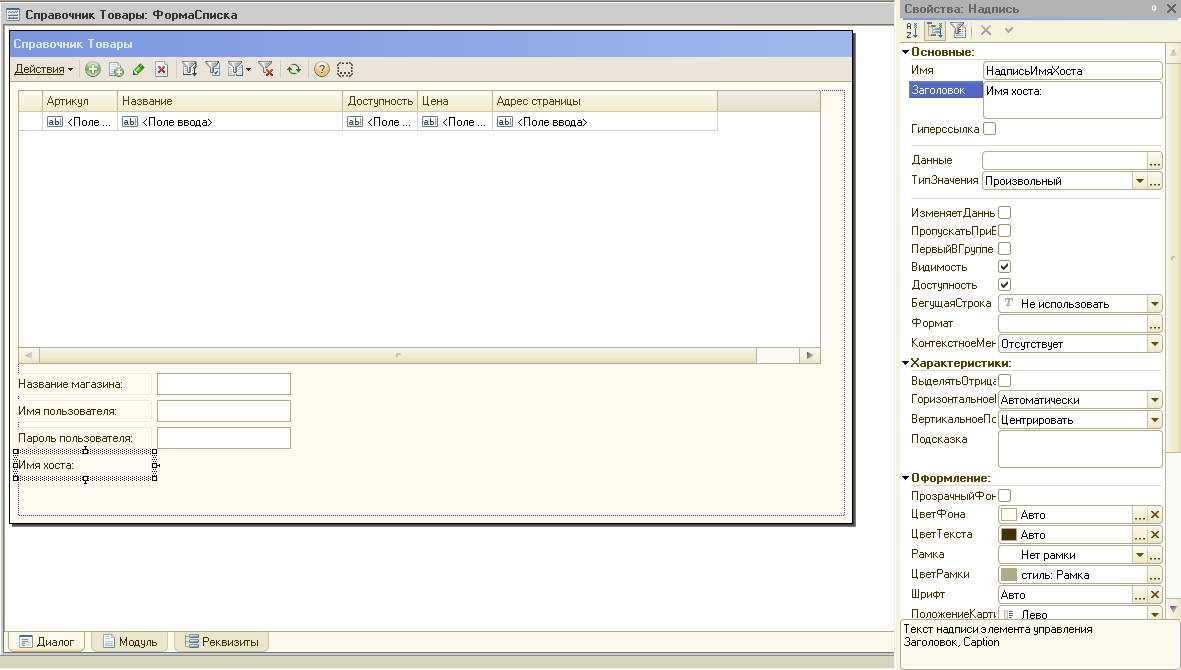
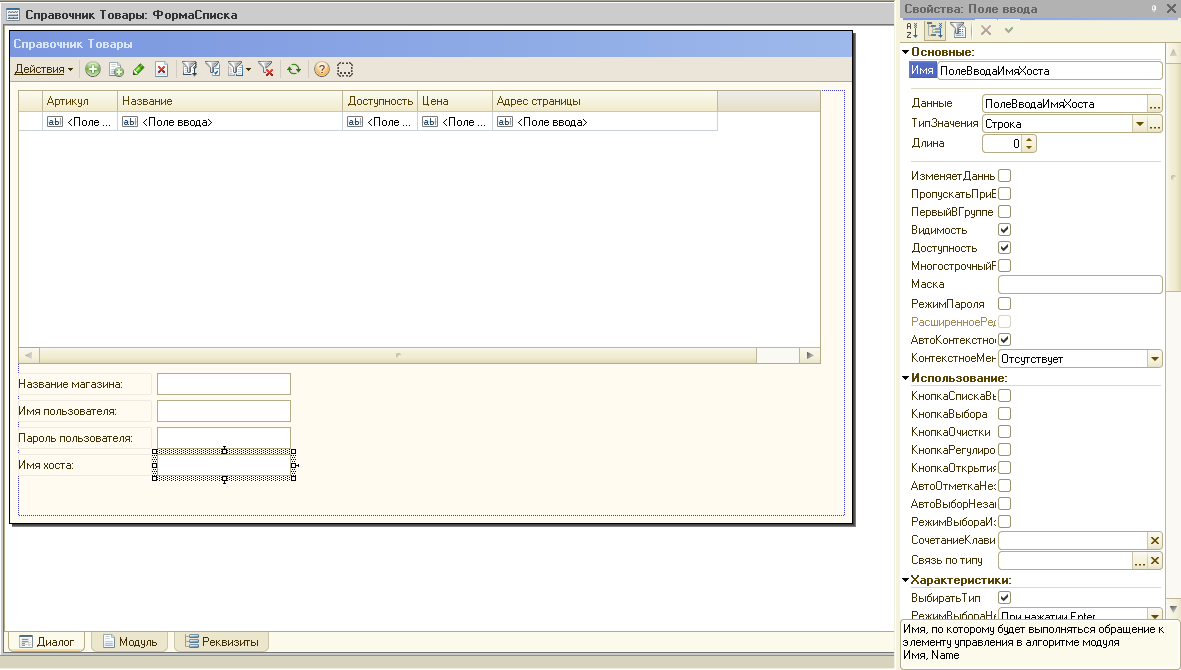


Рис. 19

Добавляем поле ввода: **Имя** — ПолеВводаИмяХоста (рис. 20).

Рис. 20

Добавляем надпись: **Имя** — НадписьНомерПорта:, **Заголовок** — Номер порта (рис. 21).

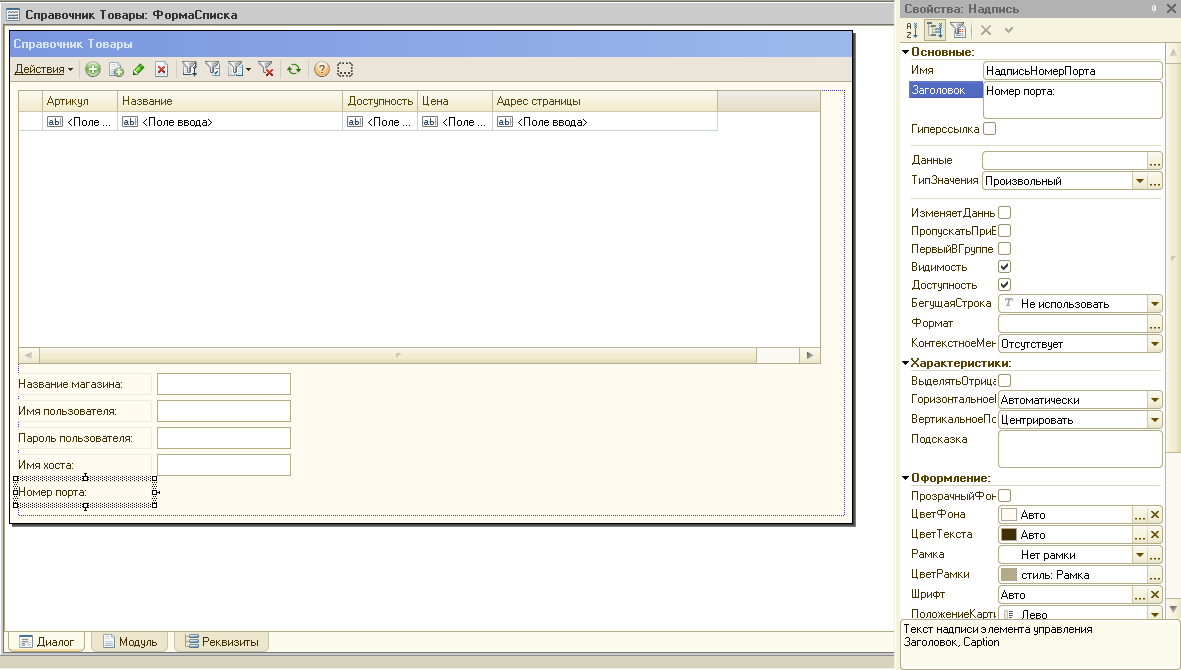


Рис. 21

Добавляем поле ввода: **Имя** — ПолеВводаНомерПорта (рис. 22).

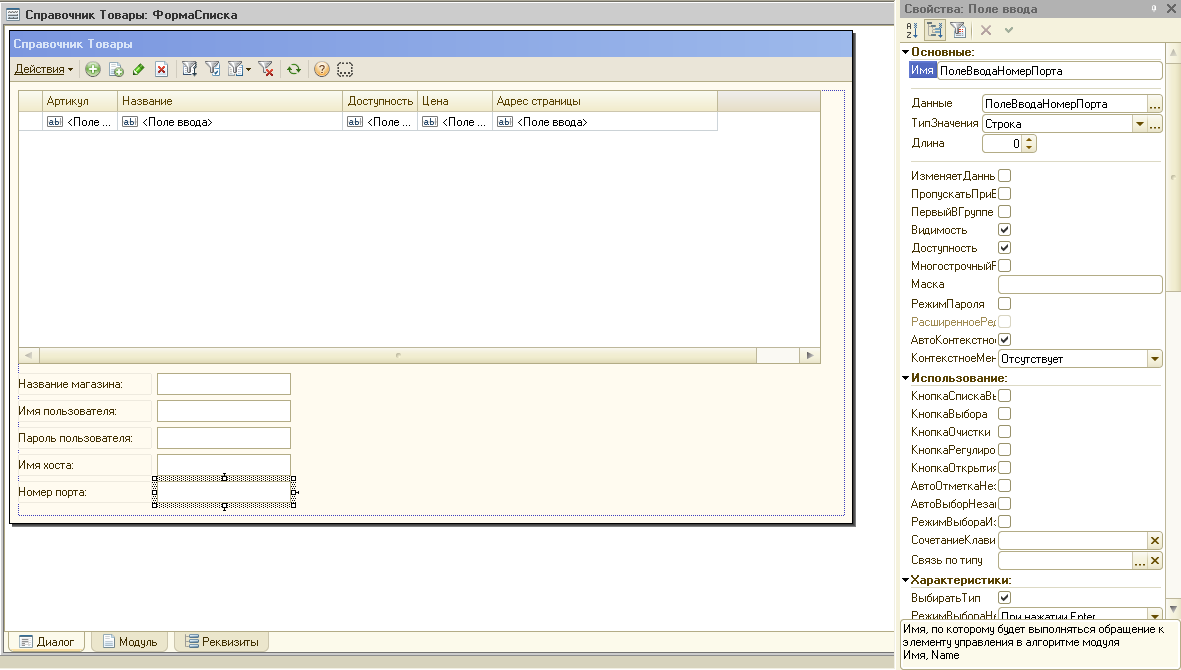


Рис. 22

Добавляем кнопку: **Имя** — КнопкаИмпортировать, **Заголовок** — Импортировать (рис. 23).

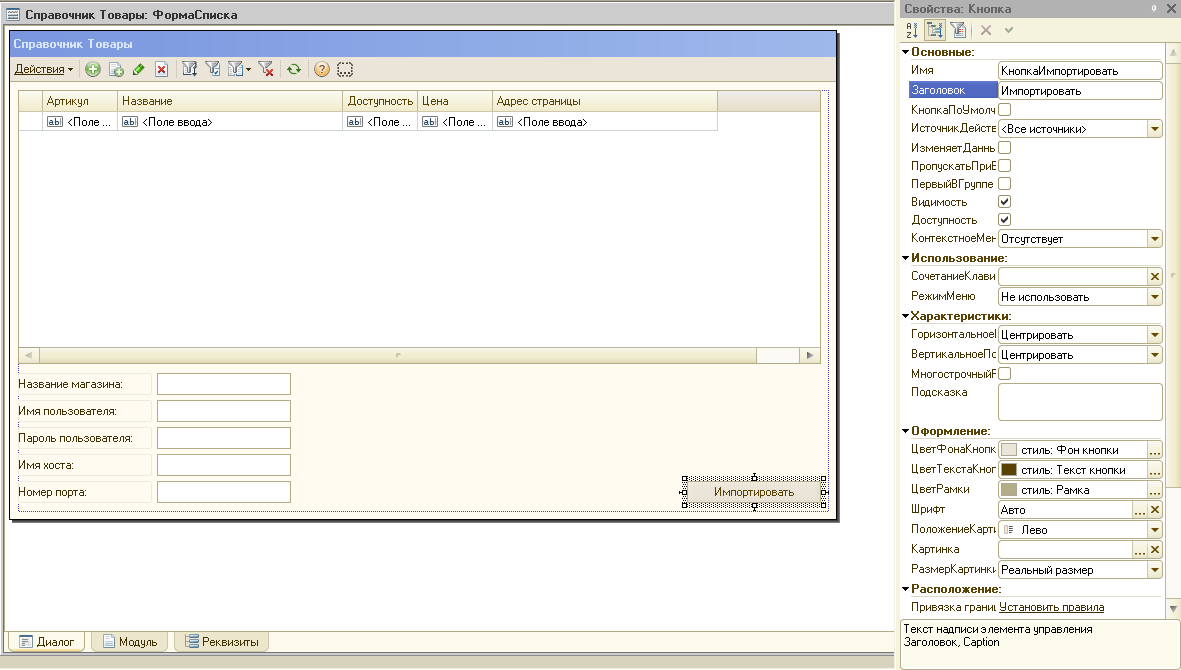


Рис. 23

В событие **Нажатие** вставляем код:

Процедура КнопкаИмпортироватьНажатие(Элемент)

// Создаем соединение с WinShop

ВинШопСоед = Новый COMОбъект("WinShopCOM.WinShopConnection");

// Подключаемся к магазину

ConРез = ВинШопСоед.Connect(СокрЛП(ПолеВводаИмяХоста), Число(ПолеВводаНомерПорта), СокрЛП(ПолеВводаИмяПользователя), СокрЛП(ПолеВводаПарольПользователя), СокрЛП(ПолеВводаНазваниеМагазина));

Если ConРез = -1 Тогда

Сообщить(ВинШопСоед.GetLastErrorStr());

Возврат

КонецЕсли;

// Количество добавленных товаров

ТовДоб = 0;

// Количество обновленных товаров

ТовОбн = 0;

СпрТовВыб = Справочники.Товары.Выбрать();

Пока СпрТовВыб.Следующий() = 1 Цикл

// Ищем товар по коду

FindРез = ВинШопСоед.FindByField("item\_code", СокрЛП(СпрТовВыб.Код));

Если FindРез = -1 Тогда

Сообщить(ВинШопСоед.GetLastErrorStr());

Продолжить;

КонецЕсли;

Если FindРез = 0 Тогда

// Товар не найден, добавляем новый в основной раздел

AddРез = ВинШопСоед.AddItem(1);

Если AddРез = -1 Тогда

Сообщить(ВинШопСоед.GetLastErrorStr());

Продолжить;

КонецЕсли;

// Назначаем товару код

SetРез = ВинШопСоед.SetValue("item\_code", СокрЛП(СпрТовВыб.Код));

Если SetРез = -1 Тогда

Сообщить(ВинШопСоед.GetLastErrorStr());

КонецЕсли;

КонецЕсли;

// Назначаем товару название

SetРез = ВинШопСоед.SetValue("name", СокрЛП(СпрТовВыб.Наименование));

Если SetРез = -1 Тогда

Сообщить(ВинШопСоед.GetLastErrorStr());

КонецЕсли;

// Назначаем товару цену

SetРез = ВинШопСоед.SetValue("price", Число(СпрТовВыб.Цена));

Если SetРез = -1 Тогда

Сообщить(ВинШопСоед.GetLastErrorStr());

КонецЕсли;

// Назначаем товару статус

Если СпрТовВыб.Доступность = "На складе" Или СпрТовВыб.Доступность = "Под заказ" Тогда

SetРез = ВинШопСоед.SetValue("status", "1");

Если SetРез = -1 Тогда

Сообщить(ВинШопСоед.GetLastErrorStr());

Продолжить;

КонецЕсли;

Если СпрТовВыб.Доступность = "Под заказ" Тогда

SetРез = ВинШопСоед.SetValue("order\_delay", "5");

Если SetРез = -1 Тогда

Сообщить(ВинШопСоед.GetLastErrorStr());

КонецЕсли;

КонецЕсли;

Иначе

SetРез = ВинШопСоед.SetValue("status", "2");

Если SetРез = -1 Тогда

Сообщить(ВинШопСоед.GetLastErrorStr());

КонецЕсли;

КонецЕсли;

// Назначаем товару адрес страницы

SetРез = ВинШопСоед.SetValue("url", СокрЛП(СпрТовВыб.АдресСтраницы));

Если SetРез = -1 Тогда

Сообщить(ВинШопСоед.GetLastErrorStr());

КонецЕсли;

Если FindРез = 0 Тогда

ТовДоб = ТовДоб + 1;

Иначе

ТовОбн = ТовОбн + 1;

КонецЕсли;

КонецЦикла;

// Отключаемся от магазина

DisРез = ВинШопСоед.Disconnect();

Если DisРез = -1 Тогда

Сообщить(ВинШопСоед.GetLastErrorStr());

КонецЕсли;

Сообщить("Импорт товаров завершен. Добавлено новых: " + ТовДоб + "; Обновлено: " + ТовОбн + ".");

КонецПроцедуры

Возвращаемся во вкладку **Формы** и добавляем новую форму: **Выберите тип формы** — **Форма элемента справочника** (Рис. 24).

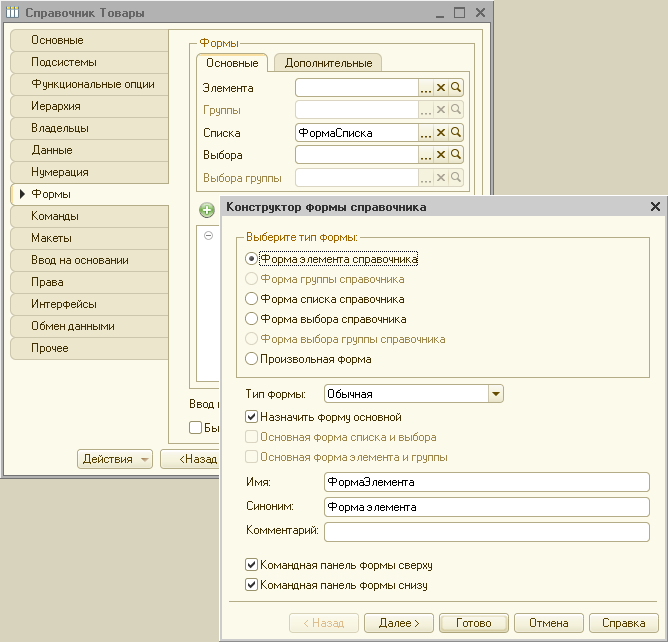


Рис. 24

Для реквизита **Доступность** меняем элемент формы с **Поле ввода** на **Поле выбора** (рис. 25).

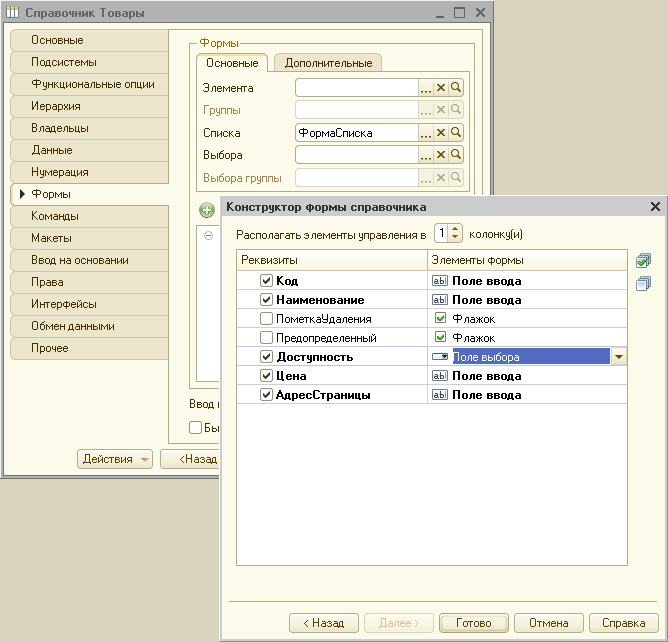


Рис. 25

На форме для поля выбора **Доступность** задаем: **Строки списка выбора** — Не складе, Под заказ, Нет в продаже (рис. 26).

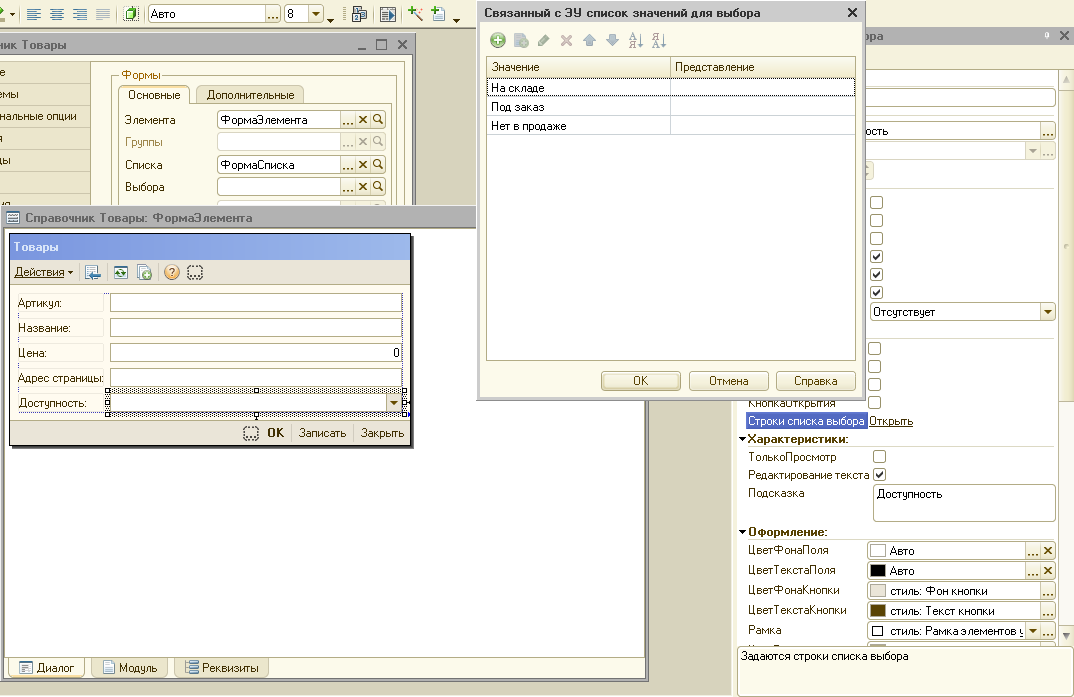


Рис. 26

Обновляем конфигурацию базы данных (рис. 27).

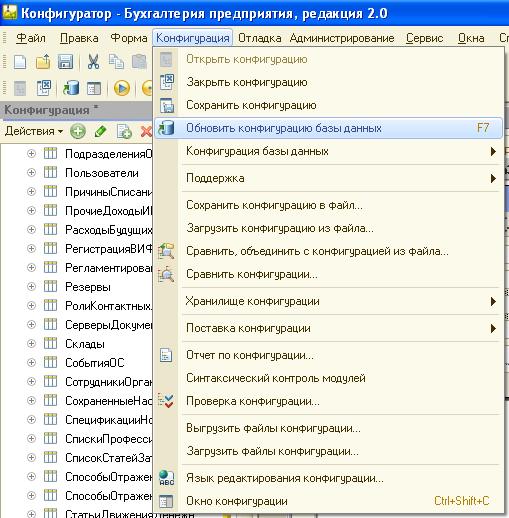


Рис. 27

Сохраняем изменения (рис. 28).

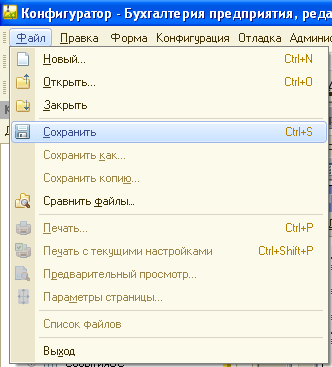


Рис. 28

Предположим, что справочник уже заполнен. Теперь можем импортировать содержащиеся в нем товары. Запускаем информационную базу в режиме 1С: Предприятия (рис. 29).

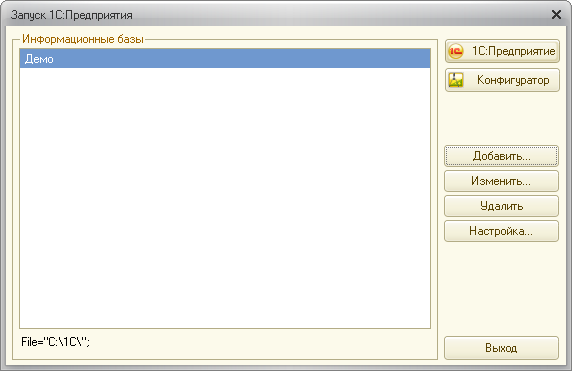


Рис. 29

Открываем список справочников (рис. 30).

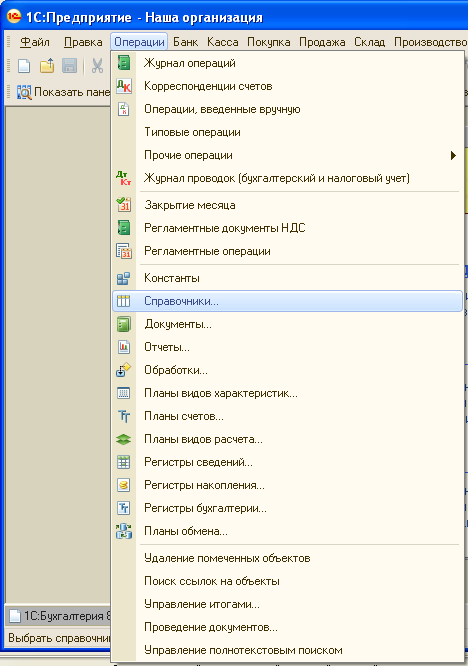


Рис. 30

Открываем справочник **Товары** (рис. 31).

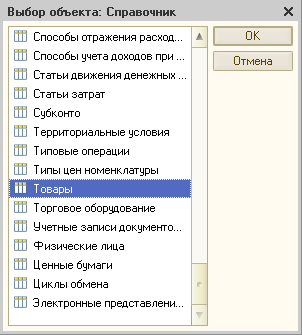


Рис. 31

Будем считать, что winshop-сервер и winshop-клиент находятся на одном компьютере, существует магазин с именем Демо и в нем содержаться товары с кодами: 01-02, 01-04, 01-06, 01-08, 01-10, 01-12, 01-14. Указываем: **Название магазина** — Демо, **Имя пользователя** — Administrator, **Пароль пользователя**, **Имя хоста** — оставляем пустыми, **Номер порта** — 0. Нажимаем кнопку **Импортировать** (рис. 32).

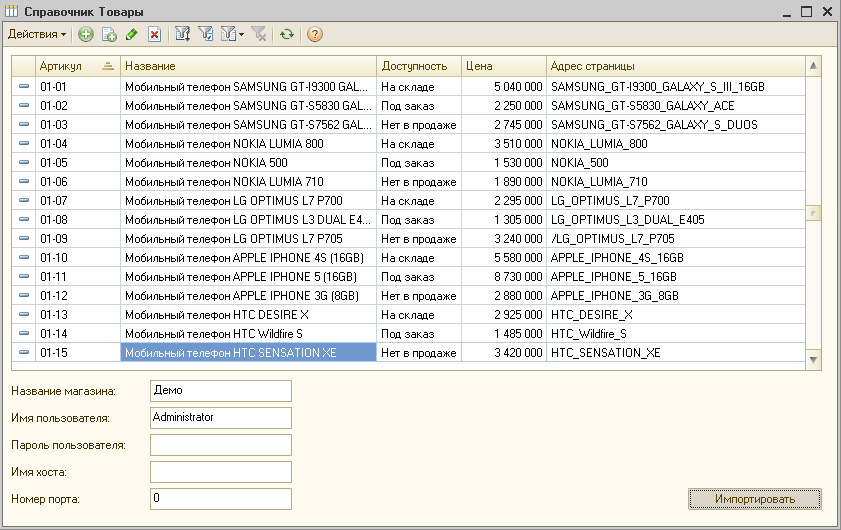


Рис. 32

Добавилось новых — 8 товаров, обновилось — 7 товаров (рис. 33).

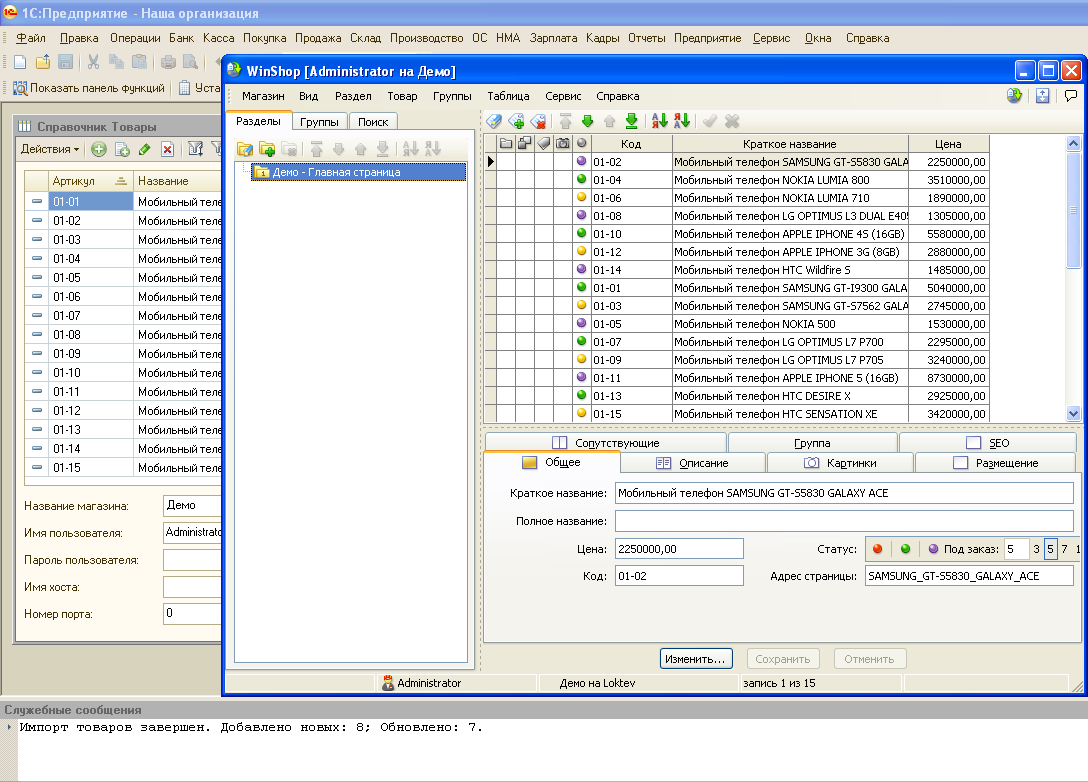


Рис. 33