cleverencesoft
программное обеспечение и консалтинг

MobileSMARTS 2008

Установка и внедрение

Версия 1.5.1.15 от 27 декабря 2010

«Mobile SMARTS. Руководство разработчика»

Для использования на территории России и СНГ.

Содержание

ООО «Клеверенс Софт»,

тел.: (495) 662-98-03,

www.cleverence.ru

Введение

Настоящее Руководство посвящено описанию системы Mobile SMARTS, её внедрению и принципам интеграции с учетными системами.

Данное Руководство ориентировано на специалистов по внедрению, программистов и разработчиков бизнес логики в учетных системах.

Структура руководства

Глава 1 дает общее представление о системе.

Глава 2 посвящена установке и основным настройкам системы.

Глава 3 посвящена разработке программ под терминалы сбора данных.

Глава 4 посвящена принципам интеграции Mobile SMARTS с учетными системами. Описывается технология информационного обмена и доступные для реализации обмена типы данных.

Что Вы должны знать

Характер изложения данного Руководства предполагает, что вы знакомы с операционной системой компьютера (Microsoft Windows 95, Microsoft Windows 98 или Microsoft Windows NT, Microsoft Windows Vista) и владеете расширенными навыками работы с ними. Также, вы должны иметь навыки разработки процессов бизнес логики в используемой вами учетной системе.

Если вы недостаточно хорошо владеете перечисленными выше понятиями и навыками, рекомендуем обратиться к документации по операционной системе и документации разработчика используемой учетной системы.

Глава 1. Общие сведения

Mobile SMARTS – это программная система, которая расширяет возможности *основной учетной системы* (ERP) предприятия, позволяя внедрить штрихкодирование с использованием специального оборудования – мобильных терминалов сбора данных (ТСД).

Mobile SMARTS состоит из четырех программных приложений:

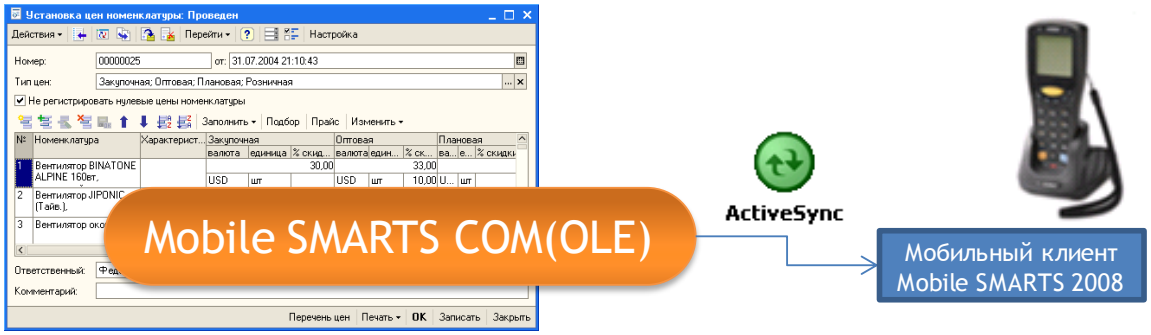
1. Серверная часть Mobile SMARTS (сервер терминалов и сервер печати);
2. Клиентская часть Mobile SMARTS для мобильного терминала;
3. COM-компонента доступа к серверу Mobile SMARTS для учетной системы;
4. Панель управления Mobile SMARTS.

Серверная часть системы (далее Сервер) представляет собой два веб-сервиса (сервер терминалов и сервер печати). Веб-сервис – это сетевое приложение, обращение к которому происходит по протоколу HTTP через определенный сетевой порт. Благодаря используемому протоколу, серверы Mobile SMARTS могут публиковаться во внешнюю сеть через Network Address Translation (NAT). По умолчанию сервер терминалов и сервер печати работают внутри двух соответствующих службах Windows, однако оба приложения могут быть запущены под Internet Information Services (IIS), начиная с версии 6.1.

Клиентская часть Mobile SMARTS (далее Клиент) представляет собой приложение на платформе *Microsoft .NET Compact Framework*, которое может быть запущено на устройствах с операционными системами, поддерживающими данную платформу (например, *Windows Mobile 2003* или *Windows CE 4.2*). Клиент Mobile SMARTS реализует общую логику складских процессов, которая может быть гибко настроена под конкретные операции. Подстройка логики Клиента осуществляется декларативно, при помощи установки набора свойств в нужные значения. Клиент Mobile SMARTS осуществляет обмен данными с Сервером для своей настройки, настройки конфигурации складов, загрузки заданий, предназначенных для текущего пользователя, а также выгрузки уже обработанных заданий. Обмен между клиентом и сервером происходит по протоколу HTTP.

Компонента доступа применяется для взаимодействия внешних систем с Сервером. Интеграция внешних систем с Mobile SMARTS заключается в разработке процедур выгрузки и загрузки данных с Сервера. Эти процедуры создаются непосредственно в учетной системе, на ее внутреннем языке. Взаимодействие с Сервером происходит через операции с объектами и функциями, предоставляемыми Компонентой доступа через механизм OLE Automation.

Возможны следующие варианты подключения и передачи данных между терминалом и учетной системой:



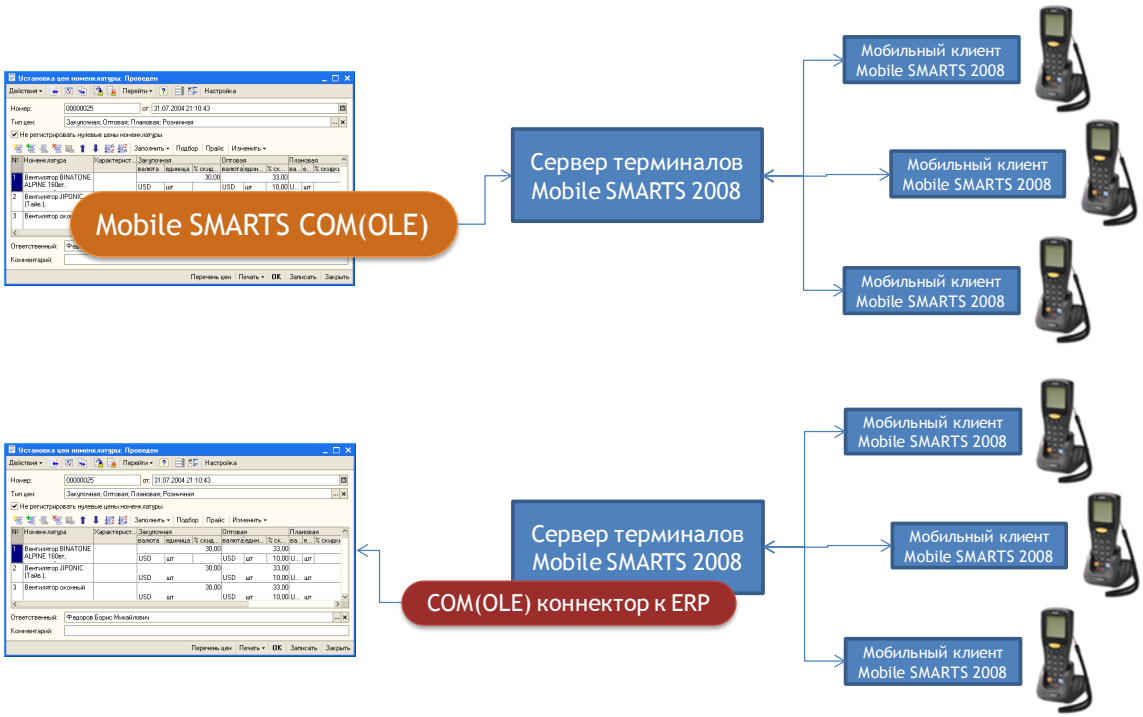


Схема взаимодействия компонентов системы

Схема взаимодействия между компонентами системы выглядит следующим образом:

1. Из учетной системы (посредством разработанных в ней процедур) на Сервер выгружается конфигурационная информация: склады и ячейки в них, пользователи и их права и т.д. Эта процедура может быть выполнена единожды, если структура статична, либо выгружаться периодически по мере изменения этих данных в учетной системе;
2. Из учетной системы выгружаются актуальные справочники товаров, единиц измерения, типов упаковки и т.д. Процедура выполняется по мере изменения этих данных в учетной системе;
3. Из учетной системы выгружаются задания пользователям на выполнение. К числу таких заданий можно отнести приемку товара, отгрузку, перемещение, инвентаризацию и другие складские операции;
4. Клиент загружает информацию о структуре системы, справочники и задания с Сервера;
5. Пользователь выполняет загруженные задания;
6. Клиент выгружает обработанные документы и измененные справочники (если такие изменения происходили) на Сервер;
7. С помощью разработанных процедур загрузки выполненные задания и измененные справочники загружаются в учетную систему и отображаются в ее информационном пространстве.

Описанный механизм представляет общую возможную схему взаимодействия и может меняться в зависимости от стоящих перед системой задач.

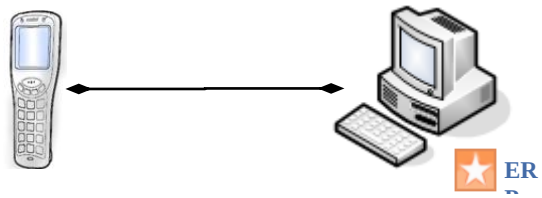
Глава 2. Установка

§ 1. Варианты развертывания

Mobile SMARTS предусматривает несколько вариантов развертывания системы. *Развертыванием* называется установка программных частей системы на разные компьютеры в сети предприятия. Различные варианты развертывания предусматривают возможность выбора количества задействованных компьютеров и распределение между ними различных частей системы.

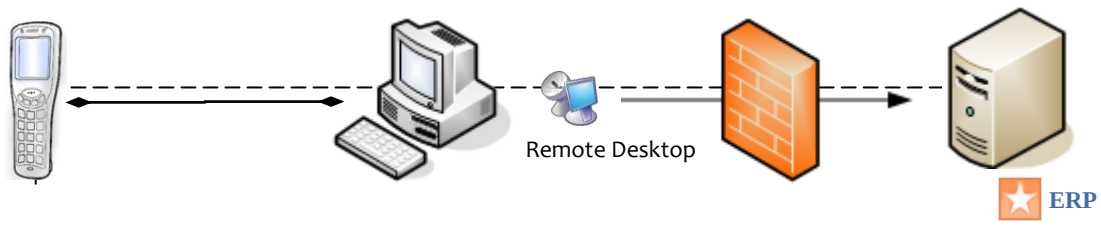
На картинках ниже представлены некоторые полезные примеры вариантов развертывания Mobile SMARTS на складе и в корпоративных сетях:

Всё установлено на одном компьютере, связь проводная:

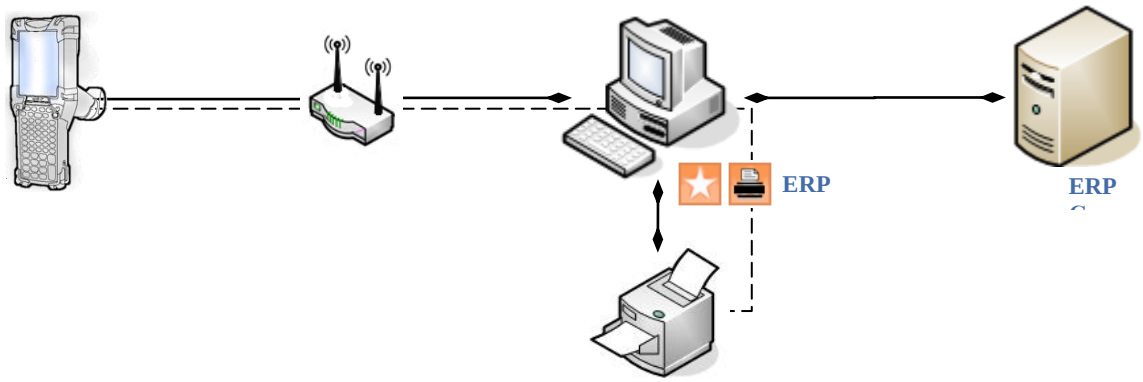


Учетная система установлена на отдельном сервере, а Mobile SMARTS просто на складском ПК, связь проводная:

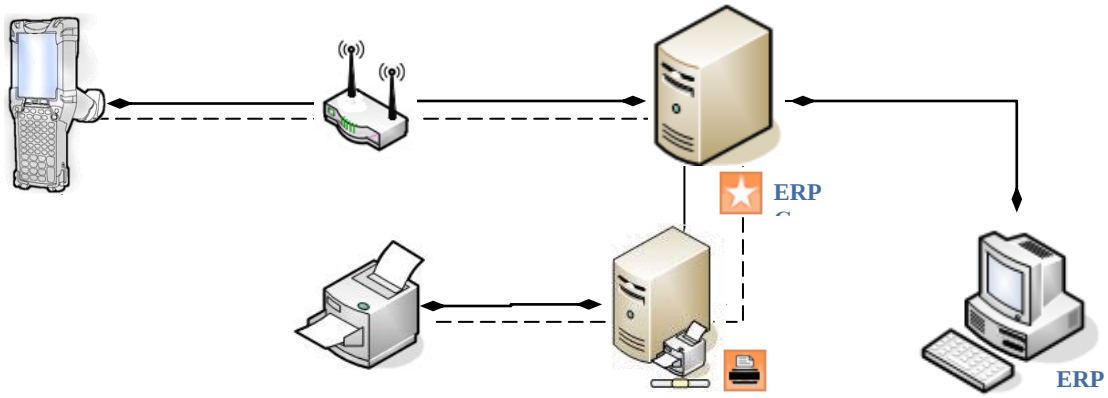
Учетная система установлена на удаленном сервере, сервер терминалов Mobile SMARTS рядом с учетной системой, связь проводная по VPN и через удаленный рабочий стол:



Учетная система установлена на отдельном сервере, а Mobile SMARTS просто на складском ПК, сервер печати установлен на складском ПК, связь беспроводная:



Учетная система установлена на отдельном сервере, сервер терминалов Mobile SMARTS рядом с учетной системой, сервер печати Mobile SMARTS также на отдельном сервере, связь беспроводная:



Возможности развертывания Mobile SMARTS не исчерпываются приведенными примерами. Поскольку части системы общаются друг с другом по HTTP, все они могут быть разнесены по подсетям и/или вынесены за файервол.

§ 2. Подготовка системы к установке

Перед установкой системы необходимо убедиться, что компьютеры, на которые предполагается установить серверное приложение Mobile SMARTS, соответствуют приведенной ниже конфигурации.

Требования к серверу

Компьютер, на котором устанавливается сервер Mobile SMARTS, должен иметь следующую конфигурацию:

- Операционная система *Windows Server 2000, Windows XP Professional* или *Windows 2003 Server* и выше;
- Microsoft .NET Framework 2.0

  Последнюю версию .NET Framework 2.0 можно взять по адресу:

  http://www.microsoft.com/downloads/details.aspx?FamilyID=0856EACB-4362-4B0D-8EDD-AAB15C5E04F5&displaylang=en

Примечание

Если у Вас уже установлена предыдущая версия программного обеспечения Mobile SMARTS, её следует удалить. Для этого в списке установленных программ диалога Windows в «Панель управления/Установка и удаление программ» следует найти пункты (если они есть):

1. «Mobile SMARTS 2008»;
2. «Mobile SMARTS 2008 – Компонента доступа»,

и для каждого из них провести процедуру удаления, нажав кнопку «Удалить».

Требования к рабочим местам учетной системы

Если на рабочих местах с учетной системой, исполняется какой-то программный код, получающий доступ к серверу Mobile SMARTS или напрямую к терминалам сбора данных через компоненту доступа Mobile SMARTS, то на каждом таком рабочем месте придется установить компоненту доступа Mobile SMARTS.

Если соответствующий код исполняется не на самих рабочих местах, а на сервере учетной системы, то компоненту доступа следует установить только на сервере учетной системы.

Компьютеры, с которых будет осуществляться доступ к серверу Mobile SMARTS через компоненту доступа, должны иметь следующую конфигурацию:

- Microsoft .NET Framework 2.0

  Последнюю версию .NET Framework 2.0 можно взять по адресу:

  http://www.microsoft.com/downloads/details.aspx?FamilyID=0856EACB-4362-4B0D-8EDD-AAB15C5E04F5&displaylang=en

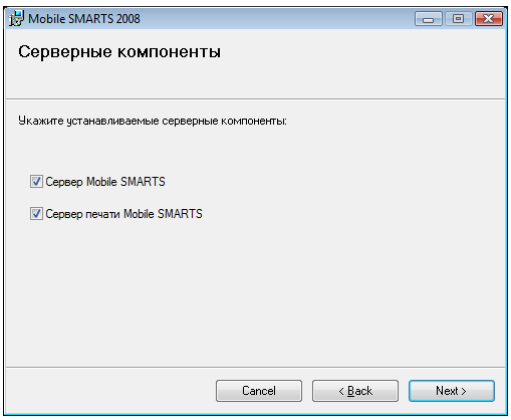
Требования к мобильным терминалам

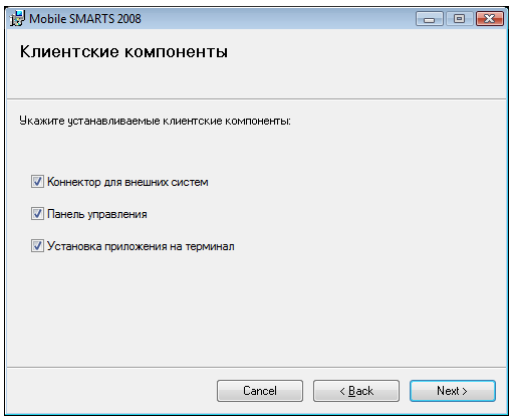
Мобильные терминалы, на которые устанавливается клиентское программное обеспечение Mobile SMARTS, должны иметь следующую конфигурацию:

- Операционная система *Windows Mobile 2003 или Windows CE 4.2 и выше;*
- *Microsoft .NET Compact Framework 2.0 SP2.*

§ 3. Порядок установки

Все компоненты Mobile SMARTS объединены в едином установщике, в котором просто галочками отмечается, что из общего дистрибутива следует устанавливать:





Установка серверов Mobile SMARTS

Сервер терминалов Mobile SMARTS устанавливается на единственном компьютере в сети, который будет выступать в качестве аппаратного сервера складской системы.

Компонента доступа требуется для конвертации документов учетной системы в формат документов Mobile SMARTS и их выгрузки на сервер Mobile SMARTS. В различных учетных системах, вне зависимости от того, как это выглядит в рабочих окнах, работа с компонентой доступа для выгрузки документов на сервер Mobile SMARTS может происходить:

a) на сервере учетной системы;

b) в клиенте на рабочих местах.

Для каждого из этих двух случаев компонента доступа Mobile SMARTS устанавливается по-разному. Либо только на сервер, либо на все рабочие места учетной системы:

1. Если выгрузка документов происходит на сервере, следует устанавливать компоненту доступа на сервер учетной системы;
2. Если выгрузка документов происходит на рабочих местах учетной системы, следует устанавливать компоненту доступа на каждое рабочее место.

Панель управления Mobile SMARTS используется для удаленного администрирования Сервера Mobile SMARTS. Следует установить панель управления на те компьютеры, с которых будет происходить администрирование системы и вестись разработка складских операций.

Для установки необходимо:

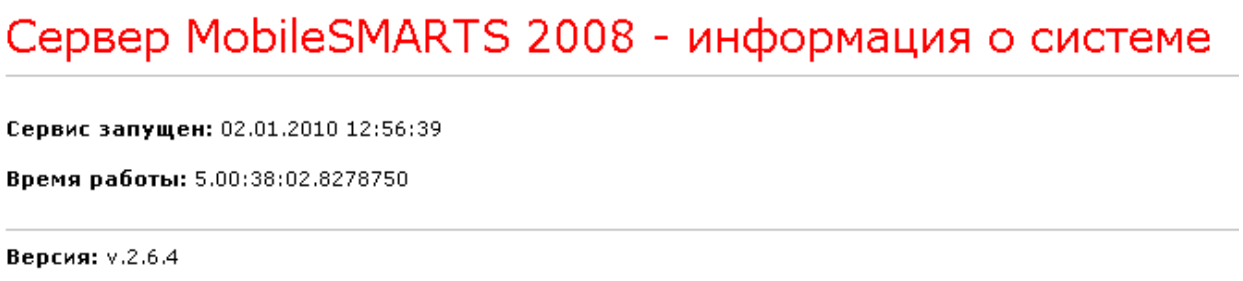
1. Вставить диск с установочным пакетом Mobile SMARTS в CD-ROM серверного компьютера;
2. Установить драйвер hasp ключа (HASP\HASPUserSetup.exe);
3. Запустить файл MobileSMARTS2008.msi, находящийся в папке дистрибутива и следовать инструкциям мастера установки;
4. Переписать ваш файл лицензии в директорию сервера. Обычно это путь вида «C:\Program Files\Cleverence Soft\Mobile SMARTS 2008\Server»;
5. Поставить HASP-драйвер из HASPUserSetup. exe;
6. Установить ключ HASP в USB порт сервера.

Проверка работоспособности сервера

На компьютере сервера запустить Web Браузер, в строке адреса вписать http://localhost:8000/

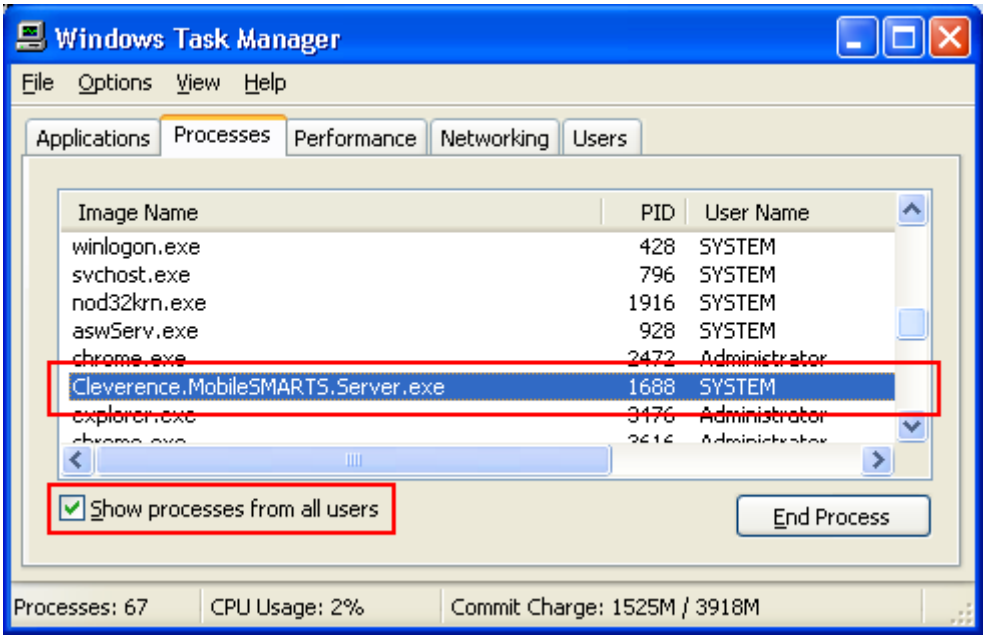
ПРИМЕЧАНИЕ: для продуктов, основанных на Mobile SMARTS (например, для Wi-Fi версии драйвера для 1С или для Легкого склада 3), номер порта может отличаться. Для выяснения реального значения номера порта следует читать документацию и смотреть конкретные настройки сервера.

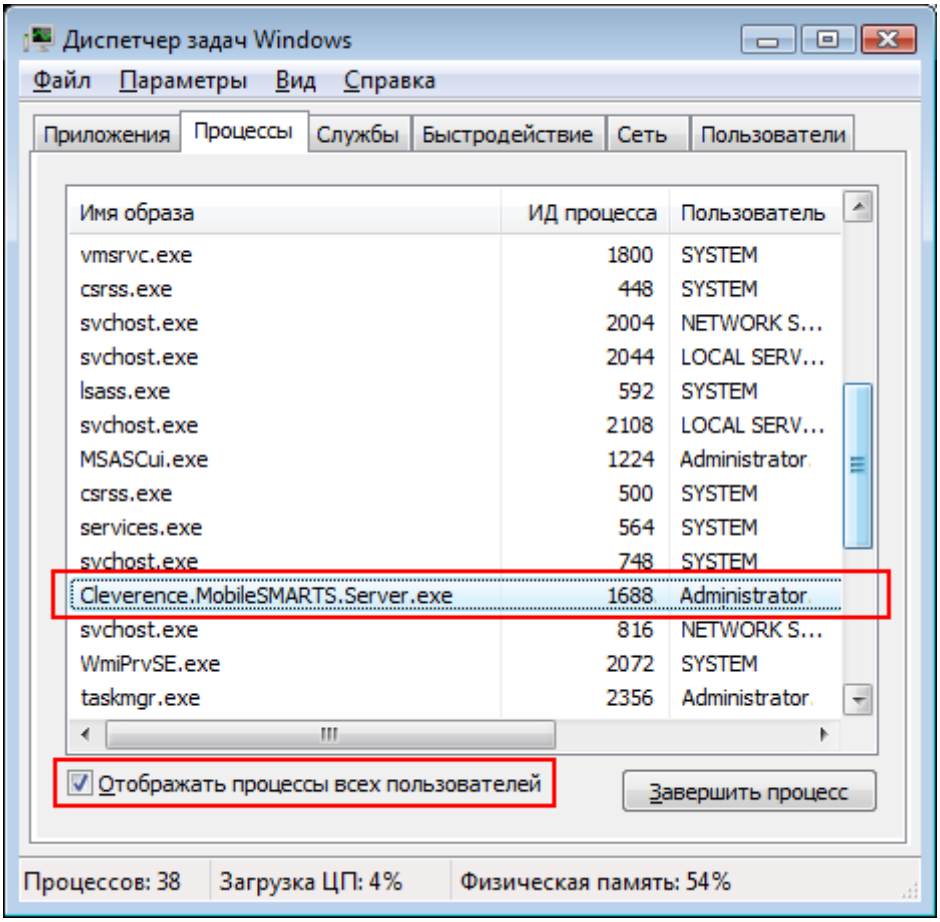
В случае успешной установки вы должны увидеть примерно следующее:



Как проверить, что сервер Mobile SMARTS запущен, и на каком сетевом порте

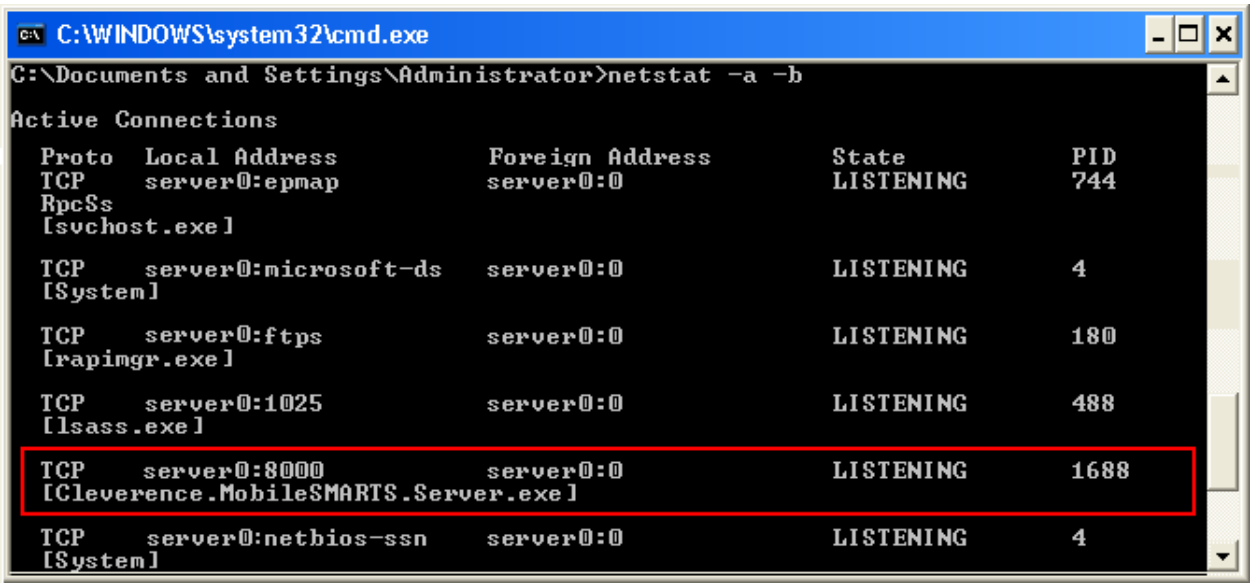
Самый надежный способ узнать, запущен ли сервер Mobile SMARTS, – это заглянуть в Task Manager (Диспетчер задач):





Если процесс «Cleverence.MobileSMARTS.Server.exe» присутствует в памяти, то следующим шагом можно узнать, на каком сетевом порте он ждет входящие соединения. Для этого в командной строке

запускаем «netstat –a -b» и ищем строку с процессом «Cleverence.MobileSMARTS.Server.exe». В колонке Local Address мы должны увидеть строку «компьютер:порт» (на картинке внизу это «server0:8000»):



Состав установки сервера терминалов

Сервер терминалов по умолчанию устанавливается в папку «C:\Program Files\Cleverence Soft\Mobile SMARTS 2008 \Server» и содержит следующие основные файлы и папки*:

bin		основная папка с рабочими dll сервера (поскольку сервер терминалов оформлен как веб-сервис).
Connectors	–	папка dll коннекторов к внешним системам. Они просто хранятся в ней и не активны. Для того использования коннектора, соответствующие ему dll следует перенести в папку bin и перезапустить сервер.
Documents	–	папка с файлами, составляющими базу данных сервера.
Cleverence.Warehouse.Environment.xml	–	файл с данными о пользователях, группах, типах документов и описание логики работы приложения на мобильном терминале.
Cleverence.Warehouse.ProductsBook.xml	–	файл с данными номенклатуры Для того чтобы убедиться, что номенклатура попала на сервер в нужном виде, нужно открыть этот файл и поискать по его содержимому. Файл ProductsBook.xml может иметь размер до 100Мб и более, и в оперативную память не поместится. В отсутствие базы данных SQL индексирование базы данных номенклатуры производится при помощи альтернативных средств (см. ниже про файлы индексов).

productsIndex.txt	–	файл с индексом номенклатуры, который содержит коды, артикулы и штрихкоды номенклатуры для поиска и соответствующие им адреса номенклатуры в файле Cleverence.Warehouse.ProductsBook.xml Это один из двух одновременно поддерживаемых реализаций индекса для быстрого поиска товаров. Позволяет просматривать справочник товаров на ТСД по порядку в окне выбора номенклатуры.
products.trie products.trie.1 products.trie.2 products.trie.3	–	файл с индексом номенклатуры, который содержит коды, артикулы и штрихкоды номенклатуры для поиска и соответствующие им адреса номенклатуры в файле Cleverence.Warehouse.ProductsBook.xml Вторая реализация индекса для быстрого поиска товаров, позволяющая искать по базам в миллион товаров, не загружая ТСД оперативную память.
Cleverence.Warehouse.UnitsBook.xml	–	файл с данными единиц измерения. Понятие «единица измерения» устарело и больше не используется. Файл сохранен из целей совместимости.
Cleverence.Warehouse.ClassificatorsBook.xml	–	файл признаков и типов признаков (см. «Признаки»).
Cleverence.Warehouse.PalletsBook.xml	–	файл с шаблонами штрихкодов паллет и контейнеров.
Cleverence.Warehouse.PrintersBook.xml	–	файл принтеров и привязок принтеров.
Cleverence.Warehouse.ServerEvents.xml	–	файл обработчиков событий сервера.
LabelTemplates	–	папка с файлами шаблонов этикеток.
Licenses	–	папка с файлами лицензий на терминалы сбора данных.
Update	–	папка с файлами обновлений программного обеспечения терминалов сбора данных. Файлы разложены по подпапкам: сначала папка с именем приложения, а в них папки с датами обновления. Их не обязательно рассказывать вручную – для работы с обновлениями в панели управления предусмотрен соответствующий пользовательский интерфейс (см. «Панель управления»).
Cleverence.MobileSMARTS.Com.Resources.dll	–	файл с ресурсами сервера: текстовые сообщения, иконки и т.п.
Cleverence.MobileSMARTS.Server.exe	–	собственно приложение сервера. При обычном запуске от лица пользователя

		запускается с ошибкой. Диалог конфигурирования сервера вызывается по ключу «/config», а сам сервер терминалов запускается с ключом «/debug», либо как служба при запуске Windows.
Cleverence. MobileSMARTS.Server.exe.config	–	.NET файл конфигурации параметров запуска приложения.
Web.config	–	.NET файл конфигурации работы веб-сервиса (приложения ASP NET).
license.xml	–	файл лицензии на сервер.
server_errors.log	–	лог всех ошибок и подозрительных ситуаций в работе сервера Если что-то не сработало, первым делом следует смотреть этот файл.
messages.log	–	лог вызовов к методам сервера. Лог вызовов ведется, если установить соответствующую галочку в диалоге конфигурации сервера. Лог вызовов помогает выявить с каких IP-адресов происходят обращения к серверу, на каких вызовах происходят дедлоки или зависания, а также проходят ли вообще интересующие вызовы.

* таблица содержит только файлы и папки, важные для понимания работы сервера.

Состав установки сервера печати

Сервер терминалов по умолчанию устанавливается в папку «C:\Program Files\Cleverence Soft\Mobile SMARTS 2008 \PrintServer» и содержит следующие основные файлы и папки*:

bin		основная папка с рабочими dll сервера (поскольку сервер терминалов оформлен как веб-сервис).
LabelTemplates	–	папка с файлами текстовых шаблонов этикеток. В отличие от визуальных этикеток, которые редактируются в редакторе этикеток панели управления (см. ниже «Редактор этикеток») и могут быть распечатаны на любом принтере, текстовые этикетки – это набор команд конкретного принтера.
Cleverence.MobileSMARTS.Com.Resources.dll	–	файл с ресурсами сервера: текстовые сообщения, иконки и т.п.
Cleverence.PrintServer.exe	–	собственно приложение сервера. При обычном запуске от лица пользователя запускается с ошибкой. Диалог конфигурирования сервера вызывается по ключу «/config», а сам сервер печати запускается с ключом «/debug», либо как служба при запуске Windows.

Cleverence.PrintServer.exe.config	–	.NET файл конфигурации параметров запуска приложения.
Web.config	–	.NET файл конфигурации работы веб-сервиса (приложения ASP NET).
printer_errors.log	–	лог всех ошибок и подозрительных ситуаций в работе сервера

Если что-то не сработало, первым делом следует смотреть этот файл.

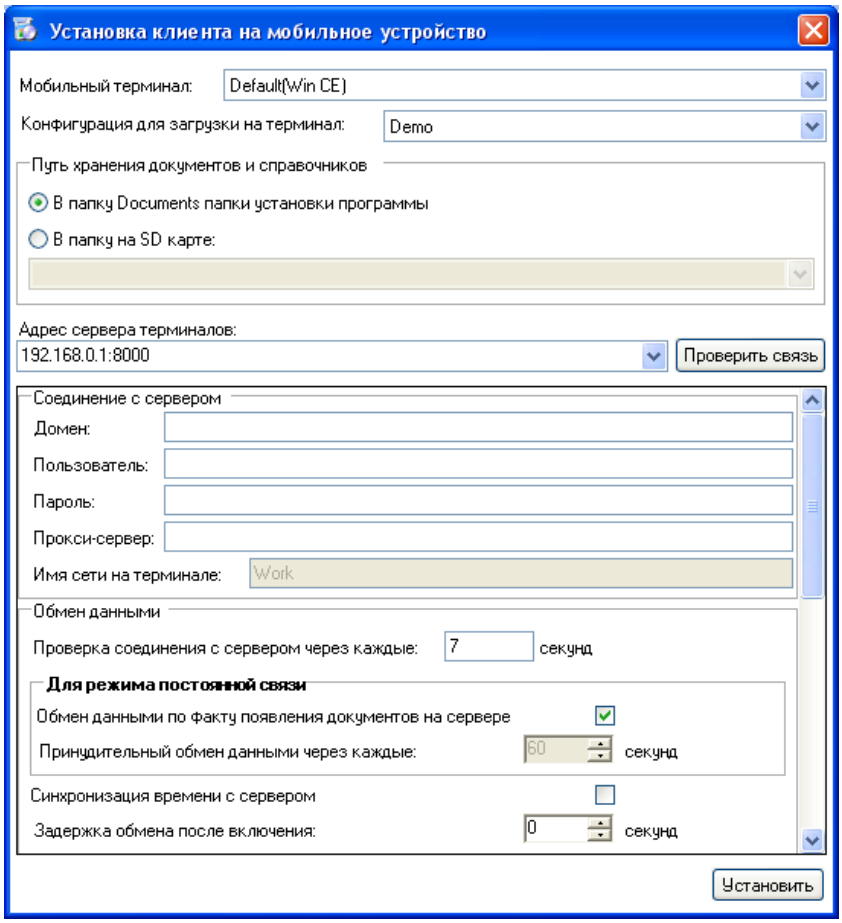
***** таблица содержит только файлы и папки, важные для понимания работы сервера.

В связи с усиленной политикой безопасности в последних операционных системах Windows, копии лог файлов сохраняются также в системную папку для общих программных файлов.
В Windows XP этот путь имеет вид: <ДИСК>:\Users\All Users\Cleverence\Logs\
В Windows Vista и Windows 7: <ДИСК>:\ProgramData\Cleverence\Logs\
Эти папки имеют статус скрытых, поэтмоу чтобы их увидеть необходимо включить в системе отображение скрытых папок и файлов.

Установка клиентского приложения на терминал

Для установки мобильного клиента на терминал необходимо:

1. Подключить к компьютеру док[1] и вставить в него терминал;
2. Запустить установщик «Пуск – Cleverence Soft – Mobile SMARTS 2.x – Установка клиента» и следовать инструкциям мастера установки.



Первое, что необходимо выбрать из предложенных или ввести в окне мастера – это IP-адрес компьютера, на котором установлен сервер Mobile SMARTS.

> **Примечание:**
> Должен быть введен именно IP-адрес, а не имя компьютера в сети.

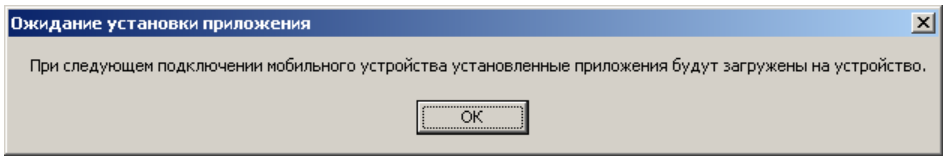
В том случае, если в локальной сети предприятия развернут домен Windows, и серверный компьютер с Mobile SMARTS в него входит, необходимо ввести название домена, а также логин и пароль одного из пользователей домена, для которого разрешен доступ в сеть. Это необходимо для того, чтобы мобильный клиент Mobile SMARTS мог зайти на страничку Web-сервиса системы с мобильного устройства, не являющегося частью домена. В случае надобности вы можете также указать прокси-сервер для доступа. На этой же странице следует указать интервал проверки соединения мобильного терминала с сервером и режим обмена данными (для online режима

[1] Док – специальное устройство для подключения и подзарядки мобильных терминалов, похожее по внешнему виду на квадратную пластмассовую пепельницу. Существуют доки с возможностью COM-порт или USB- подключения к персональному компьютеру.

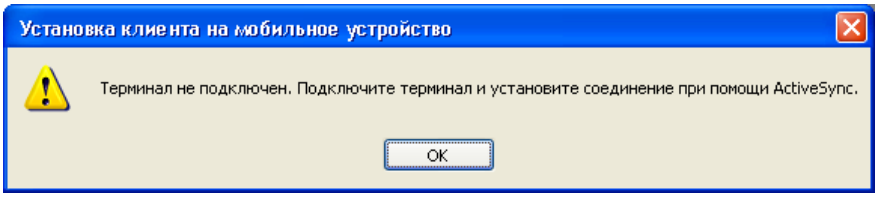
работы). Обмен данными может быть определен как по факту появления документов на сервере, так и принудительно с заданным интервалом.

Примечание:
Во избежание проблем с производительностью, не следует ставить данные временные значения слишком малыми.
Оптимальный вариант:
проверка соединения – 5 - 7 секунд;
обмен данными – 30 и более секунд.

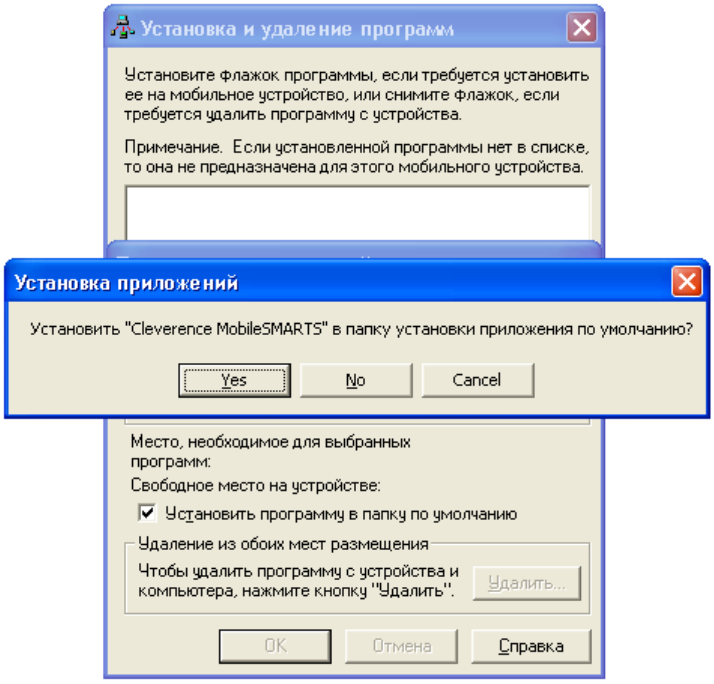
После проверки того, что все данные введены верно, следует нажать «Установить». В случаях, если на компьютере, с которого производится установка клиента, отсутствует Microsoft® ActiveSync™, док не подключен или терминал в доке отсутствует, на экран будет выведено следующее сообщение:



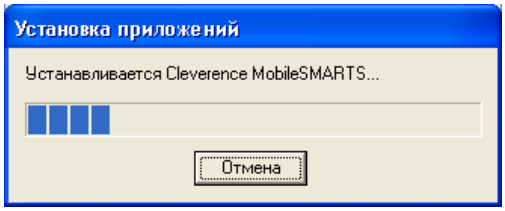
Если терминал не подключен:

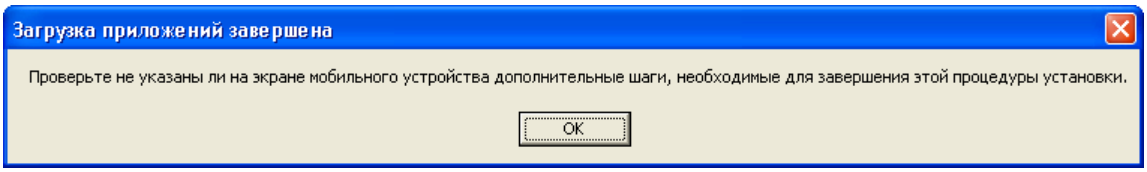


Если же всё хорошо, то на экране будет присутствовать следующая комбинация окон:

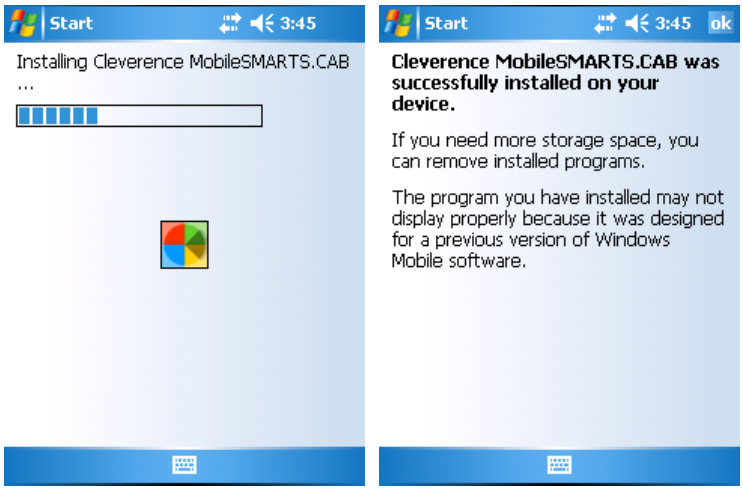


После нажатия «Да», «Yes» или «ОК» начнется копирование дистрибутива на терминал и запуск установки:

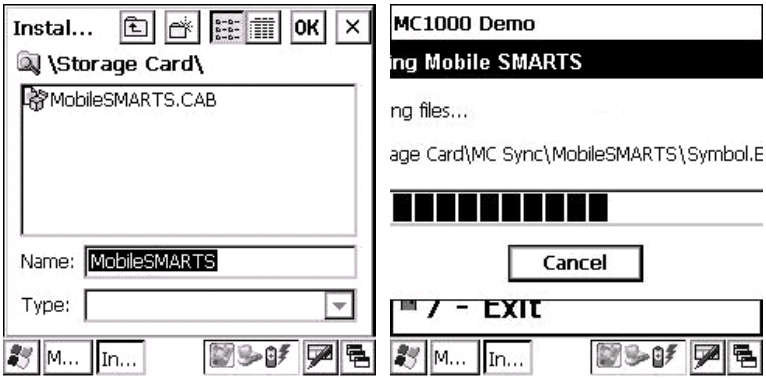




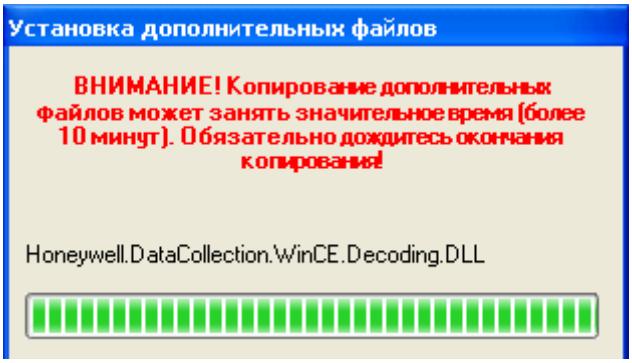
В этот момент на терминале будет высвечено диалоговое окно выбора папки для установки, и тут следует нажать «Enter»:



или



Сразу после этого происходит установка дополнительных файлов программы. Время установки таких файлов отличается в зависимости от модели, от нескольких секунд до 10-12 минут.



Состав установки клиентского приложения

Клиентское приложение для терминала сбора данных по умолчанию устанавливается в папку «\Program Files\MobileSMARTS» или «Application\MobileSMARTS» (в зависимости от модели терминала) и содержит следующие основные файлы и папки*:

Configuration	–	папка с главными файлами базы данных клиента. Эта папка лежит в основной директории программы.
Cleverence.Warehouse.Environment.xml	–	файл с данными о пользователях, группах, типах документов и описание логики работы приложения на мобильном терминале.
Cleverence.Warehouse.ClassificatorsBook.xml	–	файл признаков и типов признаков (см. «Признаки»).
Cleverence.Warehouse.PalletsBook.xml	–	файл с шаблонами штрихкодов паллет и контейнеров.
Documents	–	папка с остальными файлами, составляющими базу данных клиента. Эта папка может лежать и в любом другом месте на ТСД, не обязательно в папке установки программы. Используемое расположение папки берется из файла MobileSMARTS.exe.config (см. ниже).
Cleverence.Warehouse.ProductsBook.xml	–	файл с данными номенклатуры Файл ProductsBook.xml может иметь размер до 100Мб и более, и в оперативную память не поместится. В отсутствие базы данных SQL индексирование базы данных номенклатуры производится при помощи альтернативных средств (см. ниже про файлы индексов).
productsIndex.txt	–	файл с индексом номенклатуры, который содержит коды, артикулы и штрихкоды номенклатуры для поиска и соответствующие им адреса номенклатуры в файле Cleverence.Warehouse.ProductsBook.xml Это один из двух одновременно поддерживаемых

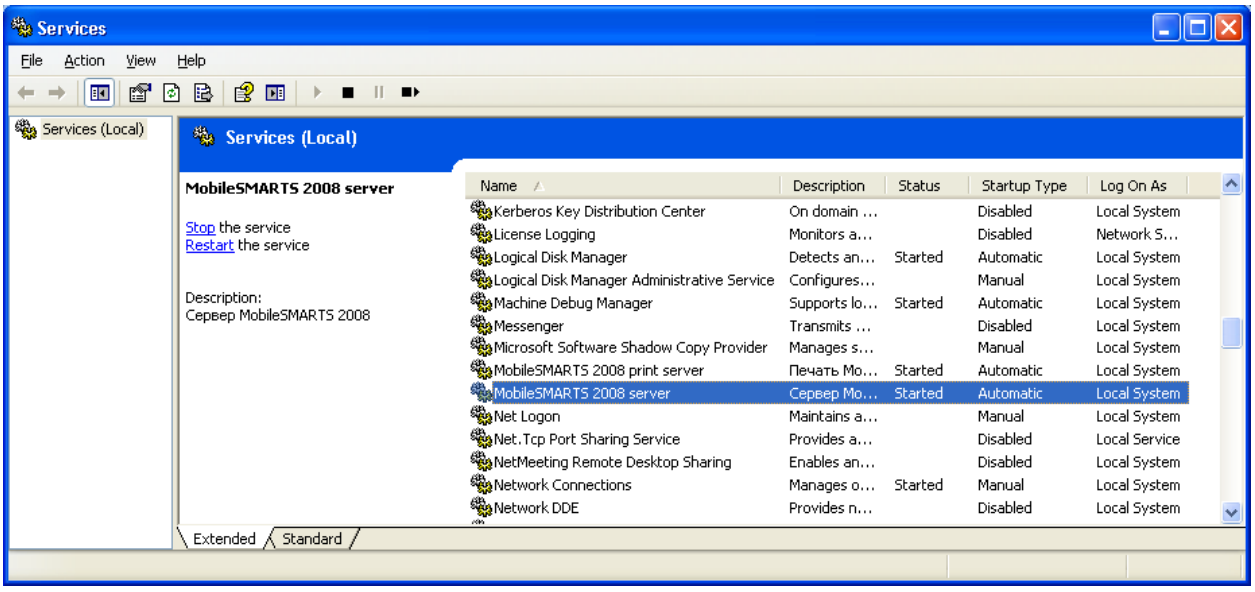
		реализаций индекса для быстрого поиска товаров. Позволяет просматривать справочник товаров на ТСД по порядку в окне выбора номенклатуры.
products.trie products.trie.1 products.trie.2 products.trie.3	–	файл с индексом номенклатуры, который содержит коды, артикулы и штрихкоды номенклатуры для поиска и соответствующие им адреса номенклатуры в файле Cleverence.Warehouse.ProductsBook.xml Вторая реализация индекса для быстрого поиска товаров, позволяющая искать по базам в миллион товаров, не загружая ТСД оперативную память.
Cleverence.Warehouse.UnitsBook.xml	–	файл с данными единиц измерения (Понятие единиц измерения устарело, файл сохранен для совместимости).
LabelTemplates	–	папка с файлами шаблонов этикеток.
Cleverence.MobileSMARTS.Compact.Resources.dll	–	файл с ресурсами клиента: текстовые сообщения, иконки и т.п.
MobileSMARTS. exe	–	собственно клиентское приложение.
MobileSMARTS.exe.config	–	.NET файл конфигурации параметров запуска приложения – описание полей см. ниже в разделе о настройках установки.
client_errors.log	–	лог всех ошибок и подозрительных ситуаций в работе клиентского приложения Если что-то не сработало, первым делом следует смотреть этот файл.

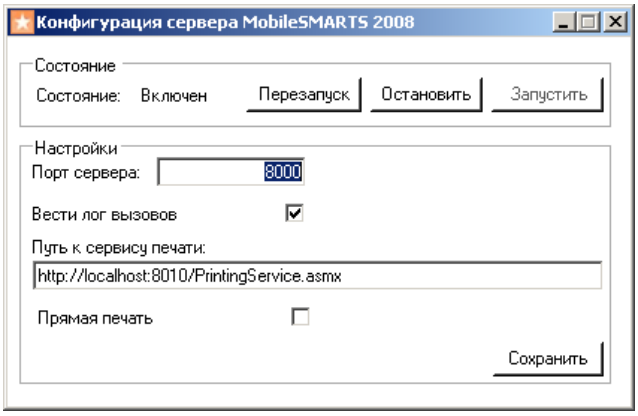
***** таблица содержит только файлы и папки, важные для понимания работы клиентского приложения.

§ 4. Настройки

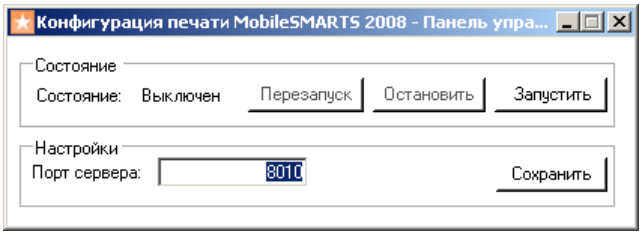
Настройки сервера терминалов и сервера печати

Сервер Mobile SMARTS и сервер печати запускаются в виде служб Windows.





Но для настройки сервисов Mobile SMARTS следует использовать программы «Конфигурация сервера Mobile SMARTS» и «Конфигурация печати Mobile SMARTS» из меню «Пуск –> Программы –> Cleverence Soft –> Mobile SMARTS 2008». Эти программы позволяют указать пути к сервисам, номера портов отдельно для каждого сервиса, а так же производить запуск и остановку соответствующих им служб в Windows.



Флажок «вести лог вызовов» влияет на ведение файла «messages.log», в котором отражаются вызовы к серверу терминалов: какой метод, когда был вызван, когда завершился. Лог помогает выловить на каких вызовах происходят дедлоки или зависания, а также проходят ли вообще интересующие вызовы.

Хотя обе программы умеют сворачиваться в трей, их совершенно не обязательно держать запущенными для того, чтобы сервер Mobile SMARTS работал.

В папке каждого из серверов находится файл Web.config, который и содержит все настройки. Номер порта, пути и галочки, показанные в предыдущих окнах, содержатся в файле Web.config соответствующего сервера. Помимо рассмотренных настроек в нем содержатся ASP.NET настройки приложения, поскольку серверы Mobile SMARTS являются веб-сервисами и их хостинг осуществляется под ASP.NET. Файл Web.config содержит примерно следующее:

```
<?xml version="1.0" ?>
<configuration>
        <configSections>
                <sectionGroup name="applicationSettings" …
        </configSections>
        <appSettings>
                <add key="DbConnection" value="" />
                <add key="processMessages" value="True" />
                <add key="printingServiceUrl" value="http://localhost:8010/PrintingService.asmx" />
                <add key="directPrint" value="False" />
                <add key="zipdata" value="True" />
        </appSettings>
        <connectionStrings />
        <system.web>
                <httpRuntime maxRequestLength="20000" executionTimeout="3600" />
                <!--
            Set compilation debug="true" to insert debugging
            symbols into the compiled page. Because this
            affects performance, set this value to true only
            during development.
                -->
                        <compilation debug="true" />
                <!--
            The <authentication> section enables configuration
            of the security authentication mode used by
            ASP.NET to identify an incoming user.
                -->
                        <authentication mode="Windows" />
                        …
                        и т.д.
```

Изменяя Web.config можно решить множество административных задач, таких как назначение прав на ресурсы, ограничения на ресурсы (используемая память, процессорное время), авторизация и многие другие.

Наиболее частая проблема, решаемая правкой Web.config связана с длиной HTTP-запроса к серверу. При выгрузке больших объемов данных (например, большого справочника номенклатуры), размер передаваемых серверу XML-данных может превысить административные ограничения. Зачем вообще нужны подобные ограничения? Безусловно, для предотвращения атак на сервер. Если размер передаваемого XML превышает максимально разрешенную величину, выдается следующая ошибка:

```
System.Web.Services.Protocols.SoapException: There was an exception running the extensions specified in the config file. ---> System.Web.HttpException: Maximum request length exceeded.
  at System.Web.HttpRequest.GetEntireRawContent()
  at System.Web.HttpRequest.get_InputStream()
  at System.Web.Services.Protocols.SoapServerProtocol.Initialize()
  --- End of inner exception stack trace ---
  at System.Web.Services.Protocols.SoapServerProtocol.Initialize()
  at System.Web.Services.Protocols.ServerProtocolFactory.Create(Type type, HttpContext context, HttpRequest request, HttpResponse response, Boolean& abortProcessing)

  at System.Web.Services.Protocols.SoapHttpClientProtocol.ReadResponse(SoapClientMessage message,
  …
```

либо русский вариант этой же ошибки, сообщающий о превышении максимальной длины запроса.

Для исправления этой ситуации следует править следующую строку в Web.config:

```
<httpRuntime maxRequestLength="20000" executionTimeout="3600" />
```

Красным выделены интересующие нас «максимальная длина запроса» (в килобайтах) и «максимальное время обработки запроса» (в секундах).

Для наиболее полной информации по редактированию данного файла следует смотреть раздел MSDN «Изменение файлов конфигурации ASP.NET» (http://msdn.microsoft.com/ru-ru/library/ackhksh7.aspx).

Настройки клиентского приложения для ТСД

Все настройки клиента Mobile SMARTS для ТСД хранятся в файле MobileSMARTS.exe.config. Файл имеет следующую структуру:

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
        <appSettings>
                <add key="webService" value="http://192.168.83.18/ " />
                <add key="updateService" value="http://192.168.83.18/ " />
                <add key="lastUpdate" value="01.01.2001 01:01" />
                <add key="processLog" value="false" />
                ...
                <add key="screenMode" value="Color" />
        </appSettings>
</configuration>
```

Параметры настройки указываются в узлах «<add key=», где «key» задает имя параметра, а «value» – его значение.

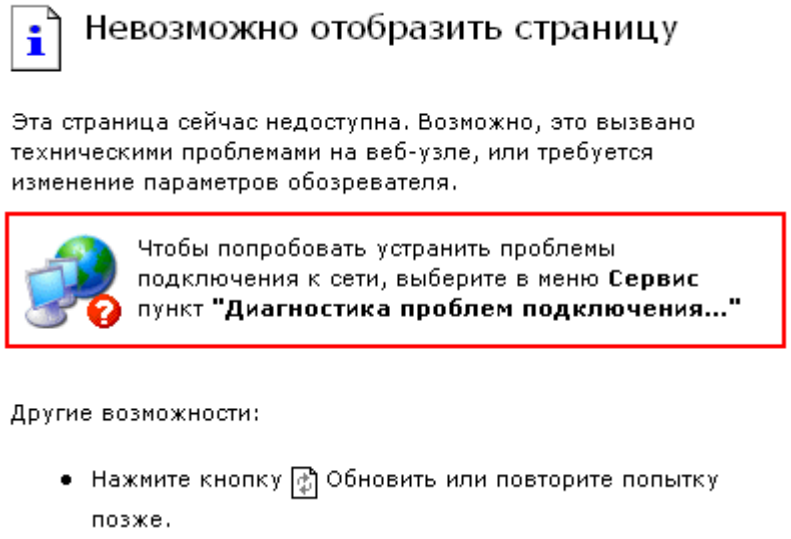
Key	Value
osVersion	Вариант операционной системы ТСД: Mobile – Windows Pocket, Windows Mobile, Windows Phone … WinCE – Windows CE
webService	Url до сервера Mobile SMARTS вида «http://...:port/DataStorage.asmx», «https://...:port/DataStorage.asmx» и т.д.
updateService	Url до сервера обновлений ПО Mobile SMARTS
lastUpdate	Дата последнего обновления ПО с сервера обновлений Mobile SMARTS
documentsPath	Путь к папке с файлами справочников и документов (по умолчанию это будет подпапка «Documents» папки запуска файла MobileSMARTS.exe)
folderBasedExchange	true – работать в offline режиме, иконку наличия подключения ассоциировать с тем, стоит ли ТСД в кредле и есть ли подключение к ActiveSync false – работать в online режиме, иконку наличия подключения ассоциировать с тем, есть ли коннект к серверу Mobile SMARTS
storageMemoryAllocation	Объем памяти в мегабайтах, которая будет выделена под хранение файлов (включая уже лежащие файлы), отдав всю оставшуюся память под оперативную
processLog	true – вести детальный лог всех операций на ТСД false – не вести
networkName	Имя профиля сетевых настроек на ТСД: Work – рабочая сеть (профиль внутренней сети предприятия) Internet – внешняя сеть через провайдера услуг Интернет <какой-то другой> – какой-то другой, конкретный, добавленный вручную
domain	Имя домена, если для входа в сеть требуется авторизация в домене
domainUser	Имя пользователя домена (см. domain)
userPassword	Пароль пользователя домена (см. domain)
proxy	Имя прокси-сервера в домене для доступа к серверу Mobile SMARTS по HTTP
onlineSyncInterval	Интервал между проверками необходимости обмена данными с сервером Mobile SMARTS, в секундах

syncTime	true – синхронизировать время на ТСД с часами на сервере Mobile SMARTS false – не синхронизировать
checkConnectionInterval	Интервал между проверками наличия подключения к серверу Mobile SMARTS (или к ActiveSync, см. folderBasedExchange), в секундах
beginCheckConnectionDelay	Задержка перед первой проверкой наличия сети, в секундах (для случая автозапуска программы при перезагрузке терминала, когда Wi-Fi просыпается позже и попытка проверить состояние сети может привести к проблемам)
vibrateNumber	Номер устройства вибрации, которое будет включаться для более наглядной индикации о важных сообщениях и ошибках при работе с ТСД
playSounds	true – проигрывать звуки сообщений, предупреждений и ошибок false – не проигрывать
screenMode	Color – использовать красочный профиль для цветного экрана BW – использовать контрастный профиль для черно/белого экрана
blockMode	true – блокировать меню «Пуск» и сочетания клавиш для его вызова и не давать пользователю возможности выходить в меню и запускать другие программы false – не блокировать
kiosk	true – работать в полноэкранном режиме, прятать меню «Пуск» и все остальные лишние меню false – работать в обычном режиме
nomenu	Параметр работает только на Windows CE терминалах и дополняет параметр kiosk. true – основное меню программы не показывается false– работать в обычном режиме
escapeKey	Имя клавиши-замены для Escape, если на клавиатуре ТСД нет клавиши Escape Например, можно указать Multiply (*) или Left (<)
forbidbacklightcontrol	true – запрещает программе управление подсветкой экрана false– программа будет управлять подсветкой, включая ее при различных действиях на ТСД и выключая при простое

§ 5. Известные ошибки установки Mobile SMARTS и пути их решения

При попытке посмотреть страницу информации сервера Вы получаете ошибку

Если вместо странички «Сервер Mobile SMARTS» вы видите следующее:



Попробуйте выполнить следующие действия:

1. Убедитесь, что Сервер Mobile SMARTS запущен. Для этого необходимо открыть службы: «Пуск –> Панель управления –> Администрирование –> Службы». В списке найти службу «MobileSMARTS 2008 server», и если в столбце «Состояние» напротив этой службы нет надписи «Работает», произвести запуск, щелкнув правой клавишей мыши по службе, и выбрав пункт «Пуск». Альтернативный способ проверки см. ниже;
2. Проверьте, включен ли файервол, не запрещает ли он доступа по порту 8000;
3. Если это не помогло, обратитесь в службу технической поддержки support@cleverence.ru.

Программа на терминале сбора данных не запускается

Иногда клиент ТСД в принципе не запускается или падает сразу при запуске с системными ошибками. Это происходит из-за неверной установки .NET Compact Framework 2.0.

Способы решения проблемы

Убедитесь, что на ТСД стоит именно .NET Compact Framework 2.0. Скачайте с веб-узла Microsoft последнюю версию фреймворка и установите его на терминал.

Необходимую версию .NET Compact Framework 2.0 SP2 можно загрузить по адресу:

http://www.microsoft.com/downloads/details.aspx?familyid=aea55f2f-07b5-4a8c-8a44-b4e1b196d5c0&displaylang=en

Программа на терминале не может соединиться с сервером

Клиент на терминале не видит сервера. Значок соединения перечеркнут, кнопка омбен данными недоступна.

Способы решения проблемы

Убедитесь, что терминал имеет соединение с сервером через ActiveSync или WiFi. Самый простой способ проверки – ввести в Internet Explorer на терминале адрес сервера, например, http://192.168.0.1:8000. Должна открыться страница информации о сервере.

Убедитесь, что сервер запущен и работает, панель управления Mobile SMARTS успешно подключается к нему.

Проверьте настройки файервола.

Убедитесь, что в конфигурационном файле клиента (MobileSMARTS.exe.config) задан правильный адрес сервера.

Программа на терминале сбора данных зависает при запуске, а в файле errors.log на ТСД написано «Не найден файл настроек»

Программа на ТСД запускается, но висит в окне загрузки или выдает ошибку «Настройки не найдены», а в файле errors.log на ТСД написано «Не найден файл настроек». Имеется ввиду файл MobileSMARTS.exe.config.

Что за файл: это файл, который содержит настройки клиента для выбранной модели ТСД и другие необходимые вещи.

Почему не найден: установщик программы терминала по какой-то причине не смог переписать его в папку приложения.

Способы решения проблемы

Поставить всё заново более терпеливо или же руками списать файл MobileSMARTS.exe.config из папки установщика клиента в папку MobileSMARTS на ТСД.

Очень медленная работа программы терминала, зависание при запуске во время работы

В большинстве случаев возникает из-за недостатка оперативной памяти терминала (Program memory). Для работы программы просто необходимо, чтобы перед запуском было свободно примерно 5Мб + 800*(кол-во номенклатур) байт RAM, но не менее 9МБ. Как считать свободную память зависит от операционной системы терминала.

Основной совет

Если не работает, нужно стараться освободить больше оперативной памяти, не забывая о том, что в отсутствии SD-карты на диске должна быть память для справочника номенклатуры (Storage memory).

Глава 3. Разработка операций для ТСД

§ 1. Самая общая вводная

Процесс разработки системы на базе Mobile SMARTS сводится к двум вещам:

1. Конфигурированию схемы метаданных Mobile SMARTS для продержки автоматизируемых процессов;
2. Разработке кода обмена данными справочников и документов между учетной системой и сервером Mobile SMARTS.

Задача 1 решается при помощи визуальных средств администрирования Mobile SMARTS. Задача 2 требует знания объектной модели Mobile SMARTS – весь код обмена данными при интеграции Mobile SMARTS с любой учетной системой разрабатывается либо внутри самой учетной системы, либо на языке программирования общего назначения (C#, Visual Basic, Delphi …) путем использования объектов компоненты доступа к серверу Mobile SMARTS.

С чего следует начать

Итак, на предприятии существует некоторая *учетная система*, которая давно и успешно используется для автоматизации учета потока товаров, сырья и материалов, готовой продукции, расходных материалов и т.д. (или же будет использоваться).

Руководством поставлена задача внедрить систему штрихкодов, для чего планируется приобрести мобильные терминалы сбора данных (ТСД) со встроенными сканерами штрихкодов.

Самое первое, с чего стоит начать – это определить, какие процессы будут автоматизированы. В самую крупную клетку это могут быть следующие процессы:

- Для торгово-распределительного склада - *поступление*, *внутренние перемещения*, *подбор*, *отгрузка*, *возвраты*, *инвентаризация*;
- Для прямой доставки товаров и грузов – *доставка* (*продажа*), *заказ*, *возврат*;
- Для склада сырья и материалов: *приход сырья и материалов*, *возврат поставщику*, *внутренние перемещения*, *подбор*, *передача на производство*, *возврат с производства*, *инвентаризация*.
- Для склада готовой продукции: *приход готовой продукции*, *возврат на производство*, *подбор*, *отгрузка*, *инвентаризация*.

Каждый процесс может исполняться в нескольких вариантах. Например, для *поступления* это будут:

1. Поступление товара на склад по предварительной заявке поставщику;
2. Поступление товара на склад без предварительной заявки (по факту).

Далее необходимо разбить каждый процесс на составляющие его операции (такие как *выгрузка*, *проверка*, *размещение* и т.д.), в которых могут использоваться ТСД. Не слишком мелко; основное внимание – на разделении обязанностей между сотрудниками. Например, для *поступления*:

1. Выгрузка товара;

2. Проверка по накладной;
3. Расфасовка по паллетам;
4. Ввод данных о каждой паллете;
5. Размещение в места постоянного хранения.

Все это составляет основу для *списка типов документов Mobile SMARTS* и *схем обработки* каждого такого документа.

Mobile SMARTS и складские процессы

Основой любой серьезной системы автоматизации учета является ориентация не на *документы*, а на *процессы*. Под «процессом» понимается связанная цепочка документов и операций, в исполнении которых принимают участие сразу несколько сотрудников. Типичный процесс – обработка поступления товара на склад.

Тем не менее, с каждым процессом связан один или несколько документов. Например, для поступления товара это как минимум *заявка на приход* и *приходная накладная* (как результат исполнения заявки).

В текущем состоянии Mobile SMARTS ориентирован на *документы*, и только во взаимодействии с учетной системой эти документы сливаются в единый процесс. Такой подход обусловлен тем, что большинство учетных систем сами по себе ориентированы больше на документы, чем на процессы, и интеграция с ними будет тем проще, чем чаще обмен данными с внешней системой будет сводиться к простому обмену документами. При использовании Mobile SMARTS можно легко добиться того, чтобы на терминал сбора данных уходили обычные складские документы-заявки, а с терминала в учетную систему возвращались готовые заполненные документы-отчеты. Это, однако, не отменяет возможностей online работы с учетной системой, с запросами и транзакциями, как этого требуют более серьезные процессы.

Рассмотрим все вышесказанное на примере все того же *поступления товара на склад*. Итак, мы условно разбили процесс поступления на 5 составляющих его операций:

1. Выгрузка товара;
2. Проверка по накладной;
3. Расфасовка по паллетам;
4. Ввод данных о каждой паллете;
5. Размещение в места постоянного хранения.

В идеале все 5 операций могут быть выполнены с использованием ТСД. При выгрузке товара – для ввода номера пломбы и проверки состояния транспортного средства; при проверке по накладной – для сверки количества; при расфасовке – для определения номера подходящей паллеты; при вводе данных – для ввода данных и, наконец, при размещении – для указания адреса постоянного хранения. На практике ТСД используются только в части из них.

Самый простой вариант процесса – когда все операции выполняет один человек. Это соответствует одному единственному документу Mobile SMARTS с самой общей схемой обработки. Документ накладной приходит из учетной системы на ТСД, исполняется, и отправляется обратно в учетную систему. Если в процессе поступления были выявлены расхождения, они будут отражены в исполненном документе с ТСД.

Более сложная схема – когда выгрузку и расфасовку по паллетам выполняет один человек (с ручным ТСД), а размещение – другой (со стационарным ТСД на автопогрузчике). Документ накладной приходит из учетной системы на ручной ТСД, исполняется, результат уходит в учетную систему, после чего она формирует документ размещения, который приходит на стационарный ТСД, исполняется, и возвращается назад. Если во время расфасовки были выявлены расхождения, или во время размещения место постоянного хранения оказалось занятым, это отражается в соответствующих документах с ТСД.

Еще более сложная схема может состоять из трех последовательных стадий (проверка машины, выгрузка/расфасовка и размещение), перепроверки товара в случае ошибок, отправки части товара в зону брака и в конечном итоге состоять из 3-4 документов ТСД.

Каждому документу ТСД соответствует определенный *тип документа Mobile SMARTS*. Например, для документов «Разместить паллету в ячейку А-01-00» и «Отправить паллету в зону брака» соответствует один единственный тип документа «Размещение паллеты». А документам «Проверка транспортного средства по накладной №ТН-2234» и «Поступление по накладной №ТН-2234» (выгрузка + расфасовка) соответствуют типы «Проверка транспортного средства» и «Поступление».

При планировании процесса следует учесть, что документы учетной системы и типы документов Mobile SMARTS не отображаются один к одному. Одному документу учетной системы (поступлению) потенциально может соответствовать любое количество типов документов Mobile SMARTS, – это будет определяться существенными различиями в работе склада при выполнении им «одних и тех же приемок», если такое имеет место, а также кодом обмена данными с учетной системой.

§ 2. Документы и их типы

Тип документа

Каждый документ относится к определенному типу (объекты Cleverence.Warehouse.DocumentType), который и описывает правила обработки документа пользователем и системой. В качестве таких типов могут быть названы, например, операция приемки ТМЦ, операция отгрузки ТМЦ, операция внутреннего перемещения и т.д.

Таким образом, для функционирования системы разработчик должен описать возможные типы документов и правила их обработки.

Каждый тип документа содержит:

Поле	Описание
Действия	Дерево действий, задающих точную последовательность обработки документов данного типа на терминале сбора данных.
Основные поля шапки	Набор основных полей шапки документа.
Основные поля строки	Набор основных полей для строк документа.
Дополнительные поля шапки	Набор дополнительных полей шапки документа. Например, для документов проверки состояния транспортного средства в шапку можно добавить номер пломбы.
Дополнительные поля строки	Набор дополнительных колонок в табличных частях документа. Указанные данные будут присутствовать в каждой строке документа. Например, это могут быть «Цвет», «Размер», «Цена» и т.д.

Дополнительные табличные части	Произвольное число дополнительных табличных частей с произвольно заданными колонками. Например это могут быть таблицы поиска чего-нибудь или локальные справочники документа.
Ошибки	Список ошибок, которые могут возникнуть при выполнении документов данного типа и быть прикрепленными к такому документу. Например, ошибка «Ячейка занята» для документов типа «Размещение».
Строки подвала	Список строк, выводимых в нижней части окна ТСД при обработке документов данного типа (по аналогии со строкой состояния в офисных приложениях).

Основной единицей любой складской операции в системе Mobile SMARTS служит документ (объект Cleverence.Warehouse.Document). Он представляет собой описание задания, которое должен выполнить пользователь, а также условий для его выполнения.

Каждый документ состоит из шести частей:

- Шапка документа. Содержит общую информацию о документе, такую как его тип, имя, текстовое описание документа, кому назначен, штрихкод, дата создания документа, кто выполнял и т. д.;
- Декларативная табличная часть документа (Document.DeclaredItems). Содержит строки с информацией о товаре, его количестве, ячейке размещения и т.д. По сути, описывает задание для выполнения. Эта коллекция строк обычно заполняется в учетной системе при выгрузке документа. По мере работы с документом на мобильном терминале происходит изменение значения DocumentItem.CurrentQuantity, которое и может быть обработано при загрузке документа назад, в учетную систему;
- Реальная табличная часть документа (Document.CurrentItems). По мере сканирования товара в мобильном клиенте, в нее будут заноситься строки записей обо всех выполненных операциях. Заполняются только те колонки, содержимое которых было получено из штрихкода или введено вручную пользователем. При выгрузке документа из учетной системы CurrentItems в большинстве случаев остается незаполненной, и заполняется уже по ходу выполнения операции на ТСД;
- Дополнительные табличные части, определяемые разработчиком;
- Коллекция ошибок, привязанных к документу. Подробнее о пользовательских ошибках в документе смотрите главу «Тип документа»;
- Коллекция привязок признаков (Document.ClassificatorUsings), т.е. флажки, проставленные различным объектам в процессе работы пользователя с документом. В качестве таких объектов могут выступать ячейки, паллеты, товары и сам документ. Эти флажки будут содержаться в документе и не имеют отношения к полям самих объектов, т.е. простановка этих флажков не влияет на объекты, которым они проставлены. В качестве флажков могут выступать «тип брака», «занято» и т.д. Указанные в документе флажки используются кодом загрузки документа в учетную систему для исполнения той или иной складской логики, связанной с ними.

Как терминал работает с документом

Вся осмысленная работа на ТСД проводится в рамках того или иного документа. Даже если задача сводится к тому, чтобы сканировать штрихкод товара, передать его в учетную систему для запроса

остатка и показать результат на экране, - технически Mobile SMARTS требует для этого создания документа, пусть даже фиктивного (типы таких документов называются виртуальными, а сами документы автоматически сознаются системой).

Открыть документ для работы можно следующими способами:

1. Получить документ-задание автоматически в рамках раздачи сервером Mobile SMARTS нужных документов нужным людям;
2. Сканировать штрихкод (с бумажной накладной, с этикетки паллеты и т.д.) и получить с сервера соответствующий ему документ;
3. Создать новый документ-задание на ТСД самостоятельно (если это разрешено конфигурацией).

Как только документ открыт, клиентская программа Mobile SMARTS для ТСД начинает шаг за шагом исполнять действия, определяющие логику выполнения задания и указанные разработчиком операции в редакторе метаданных Mobile SMARTS.

Данные в шапке документа могут быть изменены в любой момент. Однако, в отличие от учетной системы, изменения в строки документа на ТСД вносятся не путем редактирования новой или старой строки документа, а немного иным способом. По шагам он сводится к следующему:

1. Внести данные – штрихкоды, числа и строки – в именованные переменные текущей *сессии* ТСД, в которой отражен редактируемый документ, текущий пользователь и вообще все непосредственно доступные в данный момент данные. Сессию можно представить себе как вместилище записей вида «имя – значение»;
2. Когда все данные, необходимые для формирования строки документа, введены, выполняется специальное действие «Занесение данных в буфер», которая формирует из переменных сессии строку документа, но пока вносит ее не в документ, а в специальную табличную часть, называемую *буфером*, и также находящуюся в сессии. Это позволяет набрать несколько строк с возможностью отменить их все еще до внесения изменений в документ. Когда новая строка заносится в буфер, она не обязательно дописывается в конец – этим занесение и отличается от добавления. В зависимости от настроек конфигурации, она может быть объединена с какой-нибудь уже существующей строкой буфера, например для того, чтобы вместо множества строк с одинаковой номенклатурой и разным количеством иметь в буфере строки «суммированные» по номенклатуре. Всё это настраивается свойствами действия «Занесение данных в буфер»;
3. В нужный момент, когда буфер уже содержит хотя бы одну строку, может быть выполнено специальное действие «Занесение буфера в документ», которая заносит строки буфера в реальную табличную часть документа (CurrentItems). Как и в случае с буфером, занесение не обязательно означает добавление строк в конец – строки буфера могут быть объединены со строками документа по нужным критериям с суммированием количеств. Именно в момент занесения буфера содержимое документа изменяется а сам документ сохраняется;

   * Можно собрать данные из сессии и занести их в документ за один шаг, Выполнив действие «Прямая запись в документ». Результат будет тот же, что и для пунктов 2 и 3, но делается всё без создания буфера.

4. В любой момент можно удалить ненужные строки документа, при этом будут автоматически выполнены все корректировки количеств;

Рассмотрим всё это на конкретном примере документа приемки, основанном на некоторой заявке на поставку из учетной системы. Вот как бы он выглядел до начала какой-либо работы на терминале:

Шапка

Тип документа: «Приемка», **ID:** «PO1203122-08» **Имя:** «ПП №ЗПН1203122/08», **Штрихкод:** «120312208», **Назначено:** «Кладовщики», **Поставщик:** «ООО Мойдодыр», **Распространять по штрихкоду:** «Да», **Признаки:** {«Вводить партию»}.

Декларировано (свойство DeclaredItems, тип - коллекция объектов типа DocumentItem, т.е. строк документа):

Товар	Упаковка	Кол-во план	Кол-во факт	Партия	Срок годности	Паллета (SSCC)	<… и т.д.>
0001; Умывальник	061; шт.	4	0	-	-	-	
4101; Мыло душистое	060; короб	17	0	-	-	-	
0010; Зубной порошок	070; кг	20	0	-	-	-	
7564; Густой гребешок	061; шт.	300	0	-	-	-	
7564; Густой гребешок	061; шт.	100	0	-	-	-	

Реально (свойство CurrentItems, тип - коллекция объектов типа DocumentItem, т.е. строк документа):

Товар	Упаковка	Кол-во план	Кол-во факт	Партия	Срок годности	Паллета (SSCC)	<… и т.д.>
пусто							

Привязанные ошибки (свойство Errors, тип – коллекция объектов типа Error, т.е. описаний ошибок)

Имя	ID	Комментарий	Штрихкод	Фатальная	Поля
пусто					

Привязки признаков (свойство ClassificatorUsings, тип – коллекция объектов типа ClassificatorUsing, т.е. применений признака к объекту с указанным кодом)

Код признака	Код объекта
пусто	

конец

А вот как бы он выглядел после окончания приемки на терминале:

Шапка

Тип документа: «Приемка», **ID:** «PO1203122-08» **Имя:** «ПП №ЗПН1203122/08», **Штрихкод:** «120312208», **Назначено:** «Кладовщики», **Поставщик:** «ООО Мойдодыр», **Распространять по штрихкоду:** «Да», **Признаки:** {«Вводить партию», «Код состояния транспортного средства: ok»}, **Номер ворот:** «5».

Декларировано (свойство DeclaredItems, тип - коллекция объектов типа DocumentItem, т.е. строк документа):

Товар	Упаковка	Кол-во план	Кол-во факт	Партия	Срок годности	Паллета (SSCC)	<… и т.д.>

0001; Умывальник	061; шт.	4	3	-	-	-	
4101; Мыло душистое	060; короб	17	22	-	-	-	
0010; Зубной порошок	070; кг	20	24	-	-	-	
7564; Густой гребешок	061; шт.	300	300	-	-	-	
7564; Густой гребешок	061; шт.	100	20	-	-	-	
3370; Полотенце пушистое	061; шт.	0	50	-	-	-	

Реально (свойство CurrentItems, тип - коллекция объектов типа DocumentItem, т.е. строк документа):

Товар	Упаковка	Кол-во план	Кол-во факт	Партия	Срок годности	Паллета (SSCC)	<… и т.д.>
0001; Умывальник	061; шт.	3	3	-	-	2000010	
4101; Мыло душистое	060; короб	9	9	CM121007	12.04.08	2000011	
4101; Мыло душистое	060; короб	8	9	CM121007	12.04.08	2000012	
4101; Мыло душистое	060; короб	0	4	CM121007	12.04.08	2000013	
3370; Полотенце пушистое	061; шт.	0	25	У1415224	-	2000014	
3370; Полотенце пушистое	061; шт.	0	25	E1363462	-	2000017	
0010; Зубной порошок	060; короб	5	6	131108PP	13.11.12	2000019	
7564; Густой гребешок	061; шт.	300	300	-	-	2000020	
7564; Густой гребешок	061; шт.	100	20	-	-	2000020	

Привязанные ошибки (свойство Errors, тип – коллекция объектов типа Error, т.е. описаний ошибок)

Имя	ID	Комментарий	Штрихкод	Фатальная	Поля
«Поддон битый»	101	-	101	нет	Кол-во: 3 Тип: «Евро»

Привязки признаков (свойство ClassificatorUsings, тип – коллекция объектов типа ClassificatorUsing, т.е. применений признака к объекту с указанным кодом)

Код признака	Код объекта
772; «Брак»	2000013 (Паллета)

конец

Чтобы получить из первого документа второй, кладовщик сделал следующие вещи:

1. Ввел номер ворот приемки (в шапку);
2. Осмотрел внутренность транспортного средства и выбрал код состояния «ok» (в признаки документа);
3. Принял 3 умывальника в паллету 2000010;
4. Принял по очереди 3 паллеты мыла душистого все одной партии и срока годности. Две паллеты по 9 коробок принял хорошего, а брак сложил в последнюю паллету 2000013 и отметил на

терминале, что это брак. Внес на терминале данные о той неприятности, что мыло от поставщика пришло на 3х битых евро-поддонах;

5. Принял две паллеты по 25 штук пушистых полотенец, которые не значились в накладной (пересорт). Разных партий. В процессе испортил этикетки паллет 2000015 и 2000016;
6. Принял паллету зубного порошка в 6 коробов по 4 килограмма в каждом (а по накладной было 20 кг, т.е. 5 коробов);
7. Принял паллету густых гребешков в 320 штук.

На примере видно, как Mobile SMARTS работает с количествами и единицами измерения план/факт в строках документа. Для декларированных строк в колонке «план» отображается плановое количество, полученное из учетной системы. А в колонке «факт» после исполнения операции видно сколько в итоге было принято в плановых единицах измерения. Например, для пушистых полотенец, которые первоначально отсутствовали в плане, Mobile SMARTS была заведена новая строка декларированной части с плановым количеством «0». Для строк реальной части в колонке «план» отображается то количество из плана, которое приходится на данную конкретную реально принятую строку, т.е. сколько из плана она выполнила уже в реально принимаемых единицах измерения. Например, для зубного порошка там стоит 5 коробов (по 4 кг) – ведь по плану всего надо было принять 20 кг. А в колонке «факт» отображается тот факт, который был введен кладовщиком, сканирован с этикетки поставщика, либо был накоплен в результате нескольких вводов/сканирований (реальные строки документа могут объединяться, если они отличаются только количеством – это определяется настройкой конфигурации).

Все это необходимо для того, чтобы упростить построение на терминале списков просмотра план/факт и алгоритмов, проверяющих недобор/переполнение.

Еще одно важное замечание: декларативная часть документа может содержать как много строк с одной и той же номенклатурой, так и много строк с одной и той же упаковкой и всем прочим. Каждая строка реальной части содержит ссылку на строку плана, которой она соответствует. Именно потому для тех реальных строк, для которых в плане ничего не предусмотрено, создаются новые плановые строки с плановым количеством «0». А те реальные строки, которые разносятся по нескольким строкам плана, разбиваются на несколько реальных строк так, чтобы каждой новой реальной строке соответствовала только одна строка плана. В целом алгоритм разноски сводится к следующему:

1. найти строку задания, которая совпадает с реальной строкой по таким-то и таким-то столбцам, а затем пересчитать и начать прибавлять количество. Правила совпадения «по таким-то и таким-то столбцам» задаются в конфигурации;
2. если окажется, что найденная строка задания окажется переполненной (количество план больше количества факт), то попытаться найти следующую подходящую строку:
    a. если такой строки не окажется, переполнить найденную;
    b. если подходящая строка найдется, разбить разносимую реальную строку на две так, чтобы в первой осталось количество ровно по плану первой найденной плановой строки, а во вторую попал остаток. Продолжить с шага 2.

При выполнении задания пользователем могут возникать логические ошибки, требующие обработки в учетной системе. Это не системные ошибки выполнения программы, а именно логические ошибки работы самого процесса. Например ситуация, когда штрихкод товара неизвестен или ячейка, куда должна быть установлена паллета, оказалась занята. Для таких случаев в системе Mobile SMARTS предусмотрена возможность привязки к документу ошибок (Cleverence.Warehouse.Error). На этапе

выполнения ошибка может быть выбрана оператором либо из списка, либо по штрихкоду. Также можно указать, требуется ли пользовательское описание к ошибке.

Ошибка может быть двух типов: фатальная, приводящая к прекращению обработки документа, либо некритическая, позволяющая продолжить обработку задания.

Распределение документов пользователям

Система Mobile SMARTS поддерживает ряд различных настроек, позволяющих управлять правилами доставки документов на мобильные терминалы.

На процесс распределения и загрузки документов влияют как параметры конкретного документа, так и настройки операции.

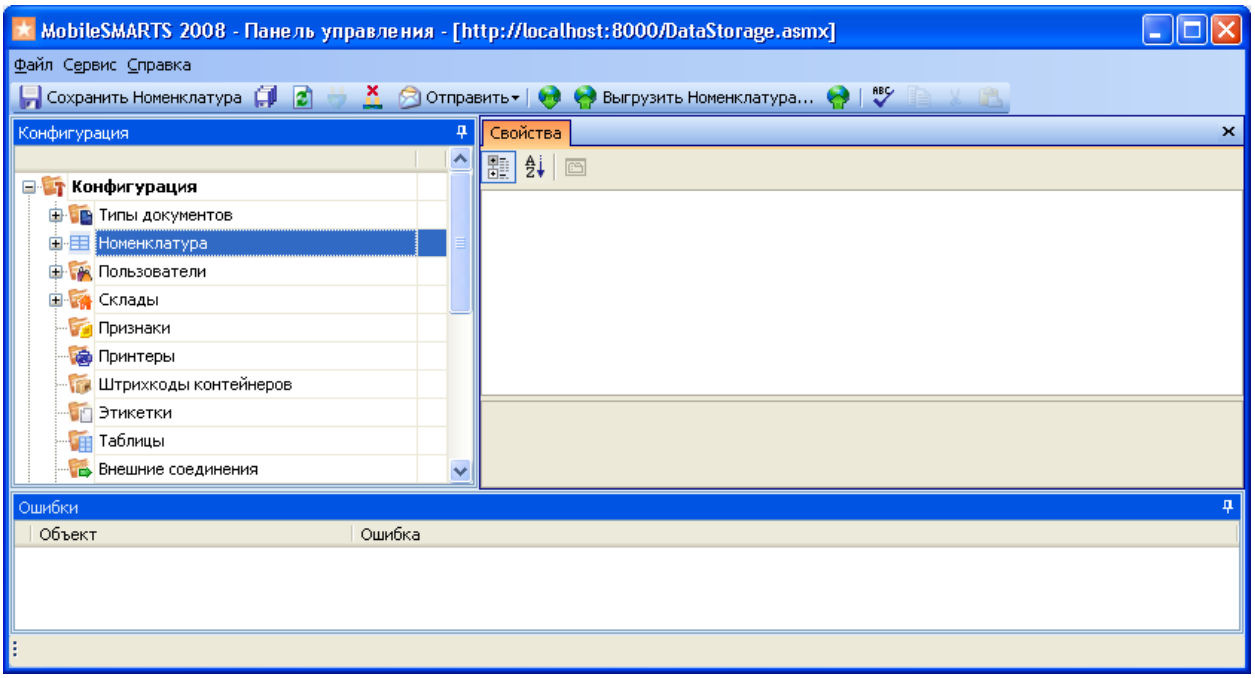
Настройки документа	Настройки операции	Поведение системы
Document.Appointment = идентификатор пользователя		Документ загрузится конкретному пользователю при первом обмене данными. Загрузятся все документы, назначенные данному пользователю.
Document.Appointment = идентификатор группы		Документ загрузится первому проводившему обмен данными пользователю из указанной группы, у которого нет других заданий для выполнения. Неперсонализированные документы выдаются не более одного на пользователя и только в том случае, если у него нет других невыполненных заданий.
Document.Appointment = “” (пустое значение)		Документ загрузится первому проводившему обмен данными пользователю, которому разрешен данный тип документа, у которого нет других заданий для выполнения. Неперсонализированные документы выдаются не более одного на пользователя и только в том случае, если у него нет других невыполненных заданий.
Document.DistributeByBarcode = True **Document.Appointment** = различные значения	Выбирать по штрихкоду с сервера = Да	Документ загружается только по запросу пользователя, когда тот сканирует штрихкод документа в окне выбора документа. Значение поля **Appointment** учитывается как разрешающий/запрещающий параметр и работает так же, как это описано выше.
Document.DistributeByBarcode = True **Document.Appointment** = различные значения	Показывать документы на сервере = Да	Список документов на терминале начинает отображать и позволять выбирать из него лежащие на сервере документы. Можно комбинировать с параметром «**Выбирать по штрихкоду с сервера**». Значение поля **Appointment** учитывается как разрешающий/запрещающий параметр и работает так же, как это описано выше.

Документы Mobile SMARTS имеют приоритет выполнения (свойство Document.Priority). При выделении общих (неперсонализированных) документов Сервер сначала распределяет документы с бОльшим приоритетом. В случае равенства приоритетов, предпочтение отдается более ранним (Document.CreateDate) заданиям. В клиентском приложении приоритет также учитывается, и документы с более высоким значением располагаются в списке выбора выше.

Обратите внимание, что в случае равенства значений приоритетов и дат создания для двух заданий, сказать заранее какой документ система даст на обработку нельзя. Если Вам необходимо точно знать очередность распределения заданий – используйте приоритеты.

§ 3. Редактор метаданных Mobile SMARTS

Основным инструментом внедрения Mobile SMARTS является *редактор метаданных Mobile SMARTS*, встроенный в *панель управления Mobile SMARTS*. В нем отражается информация о составе *списка типов документов Mobile SMARTS* и соответствующих им *схемам обработки*. Также в нем представлены данные о шаблонах ячеек и паллет, зарегистрированных принтерах этикеток и некоторая другая полезная информация.



Дерево конфигурации

Рассмотрим подробнее каждый из узлов дерева конфигурации:

Документы

В узле «Документы» содержится *список типов документов Mobile SMARTS*. На представленном рисунке это «Приемка», «Размещение», «Ручное размещение, «Заведение кодов» и «Подбор».

Номенклатура

В узле «Номенклатура» содержатся настройки справочника номенклатуры, такие как список дополнительных полей справочника.

Пользователи

Узел «Пользователи» содержит данные о *пользователях* и *группах пользователей*, зарегистрированных для работы на ТСД. Эта информация может автоматически выгружаться из учетной системы на основе существующих в ней пользователей. Группы пользователей определяют список типов документов, доступных для обработки пользователям такой группы.

Склады

Узел «Склады» содержит данные о *складах* и *шаблонах ячеек* в них. Склады определяют существующие физически отделенные друг от друга склады, а шаблоны ячеек – зоны внутри складов, целые стеллажи или даже все ячейки склада в одном шаблоне.

Признаки

Узел «Признаки» содержит данные о *признаках* и *типах признаков* – специального механизма маркировки номенклатуры, складских зон, паллет и документов дополнительными справочными данными («Скоропортящийся товар», «Зона безадресного хранения», «Паллета на ответственном хранении» и т.д.).

Штрихкоды контейнеров

Узел «Штрихкоды контейнеров» содержит список шаблонов для штрихкодов контейнеров. Под контейнерами понимаются паллеты, коробки и т.д. Задавая шаблон штрихкода контейнера, можно указать формат штрихкода, используемого для контейнеров определенного типа. Считывая штрихкод, подходящий под шаблон, терминал сбора данных будет знать, какому контейнеру он соответствует, а также сможет вынуть из штрихкода поля данных, указанные в шаблоне.

Принтеры

Узел «Принтеры» содержит данные по *принтерам* этикеток и их *привязках* к складам, типам документов и пользователям. Разные пользователи из разных документов при работе на разных складах могут печатать этикетки на заранее заданных принтерах (например, «принтер на воротах прихода А», «принтер на воротах отгрузки» и т.д.).

Этикетки

Узел «Этикетки» содержит *шаблоны этикеток* для печати на принтерах. Шаблоны редактируются в специальном визуальном редакторе этикеток и заполняются конкретными данными при их печати с ТСД.

Внешние вызовы

Узел «Внешние вызовы» содержит данные о *коннекторах* к внешним системам (например, к учетной системе) и именах *обработчиков событий сервера Mobile SMARTS*. Например, обработчик прихода выполненного документа ТСД (событие DocumentCompelted):

«`Моя_База1С:МетодГлобальногоМодуля_ДокументПришел`»

или

«`MyAxapta:EasyWarehouseClass.OnDocumentCompleted`».

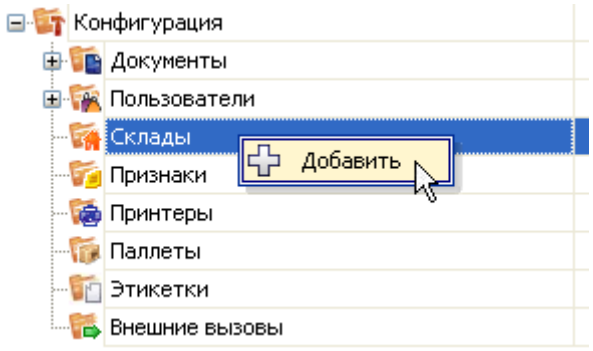
§ 4. «Hello World» для Mobile SMARTS

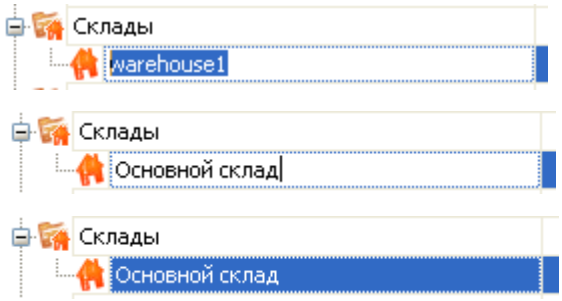
Рассмотрим самый простой пример программы под Mobile SMARTS, в котором просто что-то выводится на экран мобильного терминала. Для этого нам придется создать пустую конфигурацию, завести в ней склад, группу пользователя, пользователя в ней и, наконец, один виртуальный тип документа, в котором мы и создадим нашу самую простую программу.

Создание пустой конфигурации

Для создания пустой конфигурации Mobile SMARTS предусмотрена специальная операция в меню «Файл→Новая конфигурация».

Заведение новых складов

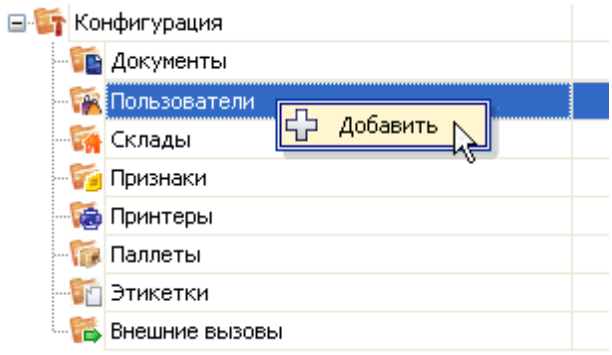


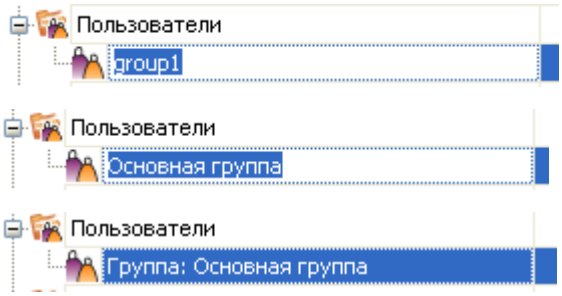


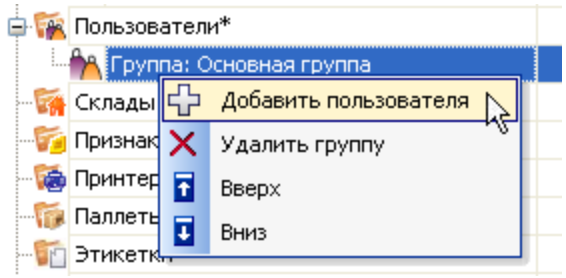
Склады необходимо заводить ДО заведения пользователей.

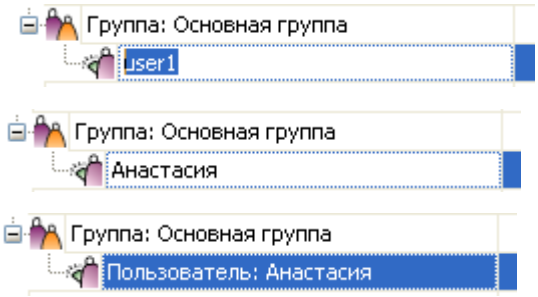
Заведение новых пользователей и групп пользователей

Правильные сценарии внедрения Mobile SMARTS предполагают, что выгрузка пользователей и групп пользователей производится автоматически (см. «Выгрузка среды»). В тех случаях, когда пользователи основной учетной системы и работники склада – это совершенно разные люди, работники склада и их группы могут быть заведены вручную в редакторе метаданных Mobile SMARTS. **До заведения пользователей необходимо завести склады.**

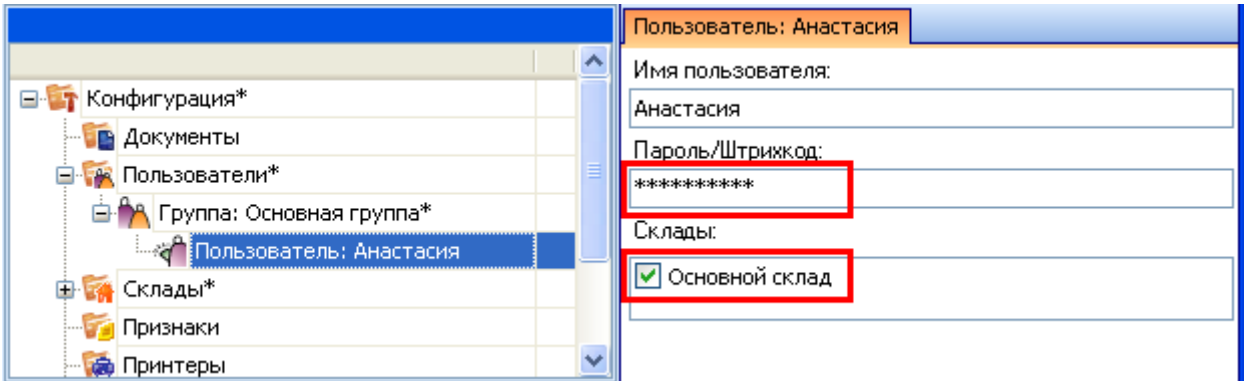








При заведении новой группы редактор ждет, что мы отметим для неё те типы документов, которые будут доступны пользователям из этой группы. Если не отметить ни одного типа документа или в конфигурации вообще нет еще ни одного типа документа, мы подучим предупреждение об ошибке.

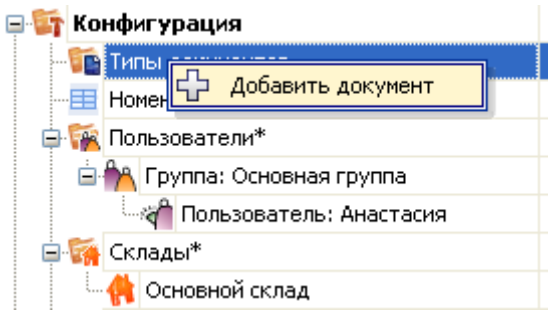


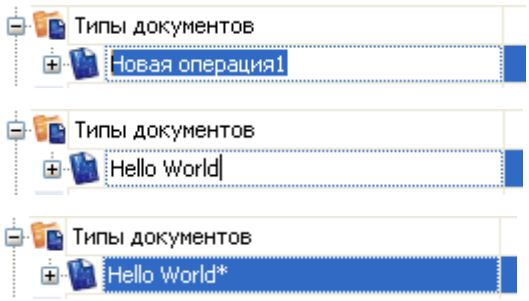
В качестве пароля/штрихкода можно вводить любую строку (в данном примере – «1»). Но если в качестве пароля ввести пробел (« »), и пользователь единственный, это будет означать, что этому единственному пользователю не нужно вводить никакого пароля. Т.е. пробел используется как бы в качестве пустого пароля.

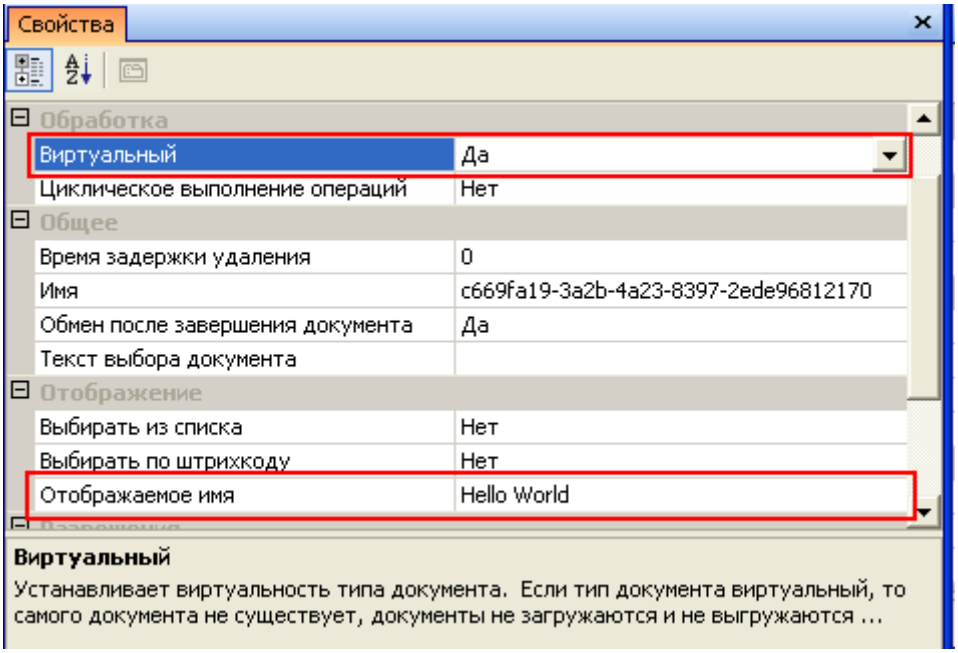
Дело в том, что пароли пользователей в Mobile SMARTS должны быть уникальны – при логине на терминале сбора данных Mobile SMARTS не спрашивает имени пользователя, а только его пароль. Соответственно, отсутствие пароля запрещено, и только один пользователь из всех может иметь пароль в виде пробела. Если в системе несколько пользователей, ему придется вводить пробел. Если это единственный пользователь, Mobile SMARTS на терминале сбора данных не будет спрашивать никакого пароля, сразу запускаясь от имени этого единственного пользователя.

Добавление новой виртуальной операции

Для нашей цели – просто показать на экране какой-нибудь текст – мы будем использовать виртуальный тип документа. «Виртуальный» означает, что никаких документов на терминале сбора данных сохраняться не будет. При нажатии кнопки с названием виртуального типа документа в основном меню Mobile SMARTS на мобильном терминале мы не увидим списка документов. Вместо этого автоматически создастся новый документ соответствующего типа, и на экране терминала мы увидим первое окно в программе обработки документов этого типа.







Основы программирования в Mobile SMARTS

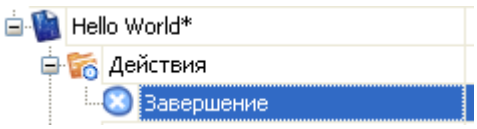
Прежде чем выводить текст на экран терминала – пару слов о программировании в Mobile SMARTS. В каждом заводимом типе документа существует узел «Действия», в котором и выполняется программирование поведения терминала сбора данных при выполнении документов такого типа.

В своей основе, программирование операции в Mobile SMARTS сводится к компоновке операции из готовых блоков в визуальном редакторе. Готовые блоки называются *действиями*, и сами по себе тоже могут настраиваться при помощи установки их свойств в различные значения.

На самом деле, одним и тем же словом «действие» в данном руководстве обозначаются две разные вещи – 1) сам тип готового блока и 2) конкретный экземпляр готового блока такого типа. Когда где-либо в тексте упоминается то или иное «действие», точный смысл слова либо понятен из контекста фразы, либо уточняется отдельно.

Узел «Действия» в любом типе документа может содержать произвольное количество экземпляров одного и того же действия, каждый из которых может отличаться от другого выставленными значениями своих свойств. Значения свойствам проставляет разработчик операции.

В примере «Hello World» только что созданный тип документа содержит всего одно действие – действие «Завершение»:

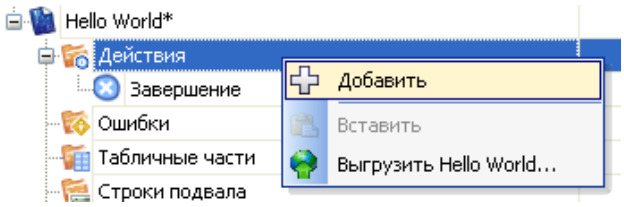


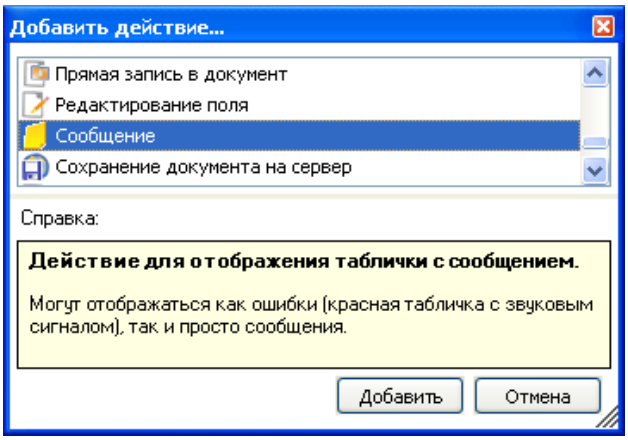
Как понятно из названия, это действие завершает работу с документом на мобильном терминале сбора данных.

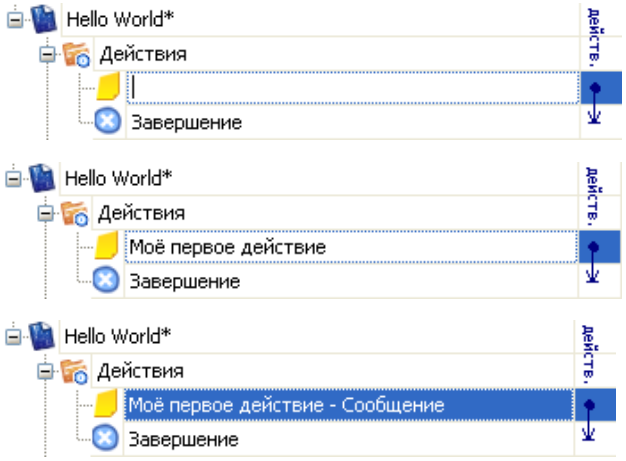
Если явно не указано иное, работа на терминале сбора данных по очереди переходит от одного действия к следующему за ним в списке до тех пор, пока работа с документом не завершится одним из предусмотренных способов.

Вывод сообщения на экран терминала сбора данных

Чтобы вывести на экран терминала сбора данных какой-нибудь текст, в виртуальную операцию «Hello World» нужно будет добавить действие вывода сообщения.

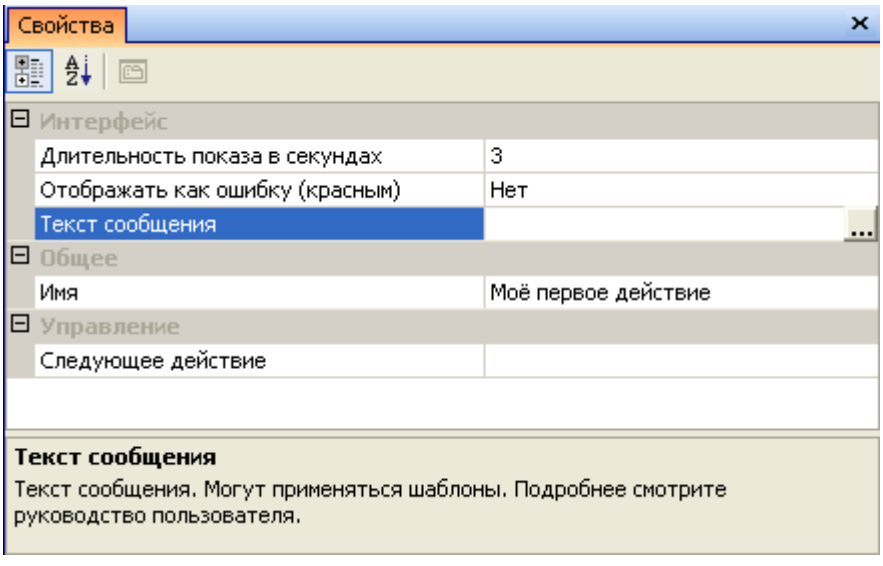




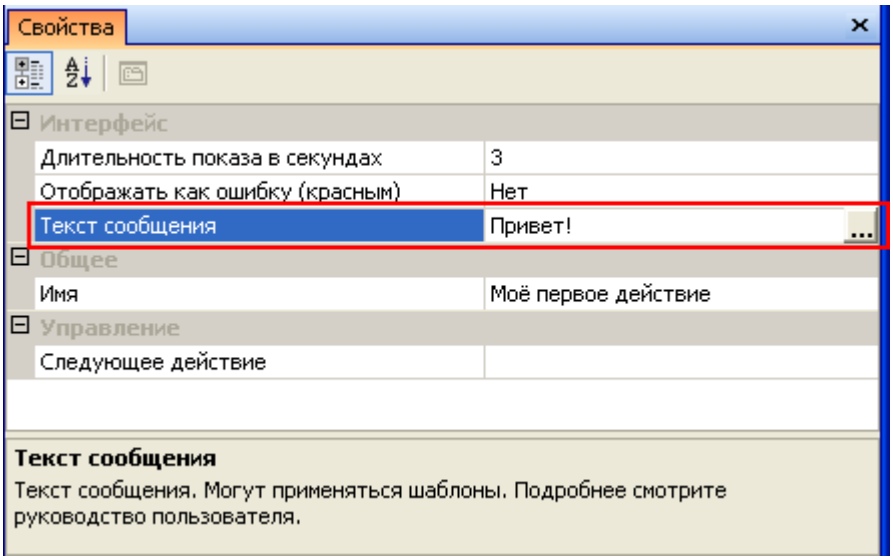


Набранный текст «Моё первое действие» не будет отображаться на экране – это не текст в окне, а имя экземпляра действия, по которому это конкретное действие будет идентифицироваться в дереве визуального редактора. Основное применение имен – адресация в условных и безусловных переходах. В сравнении с такими языками программирования, как ассемблер или бейсик, имя действия аналогично номеру строки. В действительности, никто не запрещает давать действиям такие имена как «10», «20» и т.д.

Имя – одно из свойств действия. Другие свойства отображены в редакторе свойств сбоку от дерева:



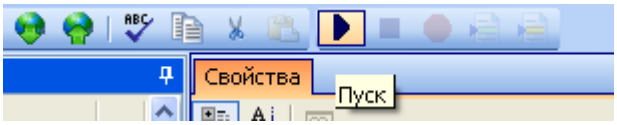
Главное для нас свойство – «Текст сообщения»:



Запуск программ на отладку

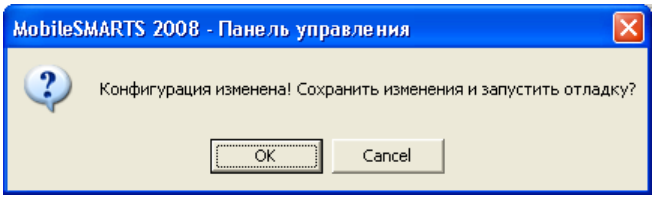
Отладчик предназначен для запуска операций на ТСД в целях тестирования процессов обработки документов. Прямо во время отладки существует возможность просмотра и изменения данных документа и сессии, а также изменения свойств действий (!!). Отладчик является частью редактора метаданных Mobile SMARTS внутри панели управления Mobile SMARTS.

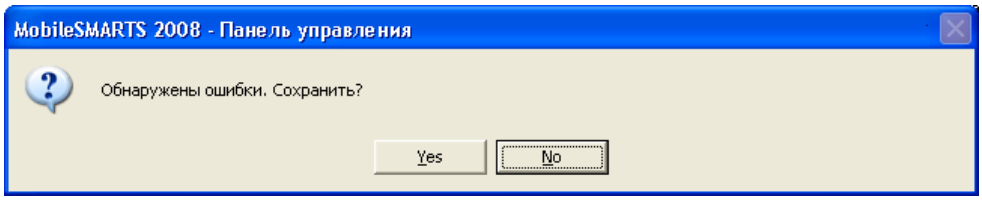
Для того, чтобы увидеть результат на экране терминала, нужно подключить терминал к компьютеру при помощи кредла/кабеля и программ ActiveSync (для XP) или Центр мобильный устройств (для Vista), дождаться установления соединения и запустить операцию на отладку:

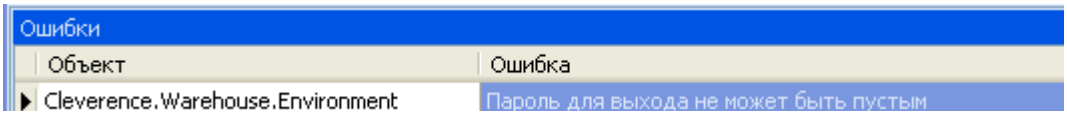


Запуск на отладку попросту означает, что на терминале, подключенном к компьютеру, запустится Mobile SMARTS, а в нем запустится та операция, тип документа для которой сейчас выделен в дереве (или выделен один из его дочерних узлов, например какое-нибудь действие).

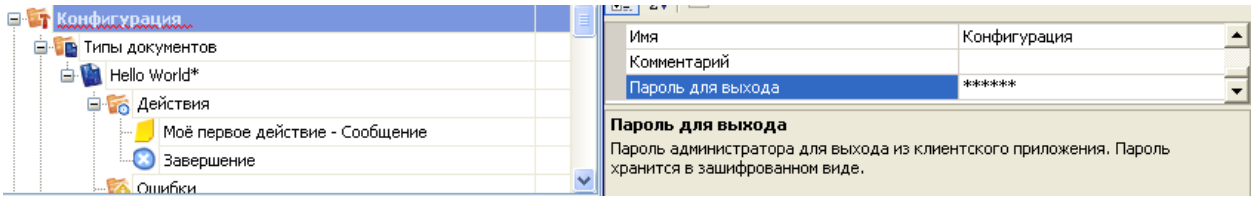
Перед запуском на отладку в самый первый раз панель управления Mobile SMARTS задаст несколько вопросов:





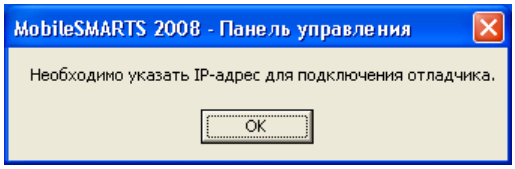


В данном случае редактор просит задать пароль администратора для выхода из Mobile SMARTS на мобильном терминале сбора данных. Пароль администратора – это свойство конфигурации, соответственно следует выделать узел «Конфигурация» в дереве и найти соответствующую строку в таблице свойств сбоку от дерева:

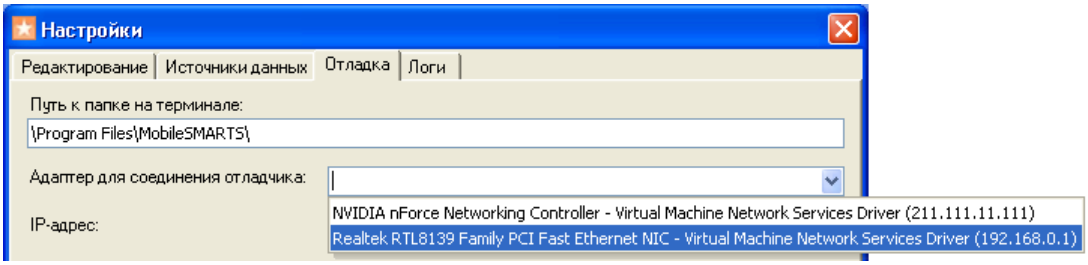


Как и в случае с паролями обычных пользователей, в качестве пустого пароля используется пробел (« »), т.е. если указать в качестве пароля просто пробел, то при попытке выйти из Mobile SMARTS на терминале сбора данных не будет запрошено никакого разрешающего это пароля.

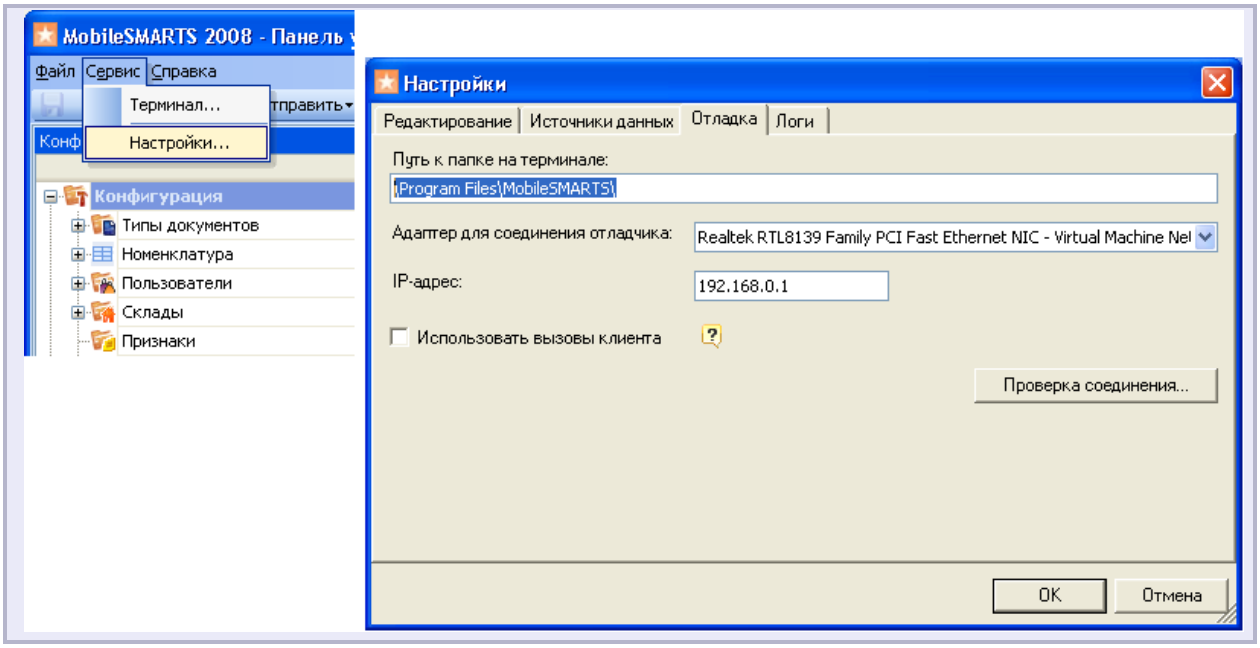
После установления пароля, сохранения и второй попытки запуска на отладку мы получим еще несколько вопросов:



Панель управления и клиент Mobile SMARTS на мобильном терминале сбора данных общаются друг с другом по сети через HTTP, и запуск на отладку требует указать IP-адрес компьютера, на котором запущена панель управления. Этот IP-адрес будет передан в Mobile SMARTS на терминале сбора данных для того, чтобы программа терминала могла отправлять отладочные данные в панель управления. Поскольку компьютер с панелью может иметь несколько сетевых адаптеров с несколько IP-адресов, некоторые из которых могут быть недоступны с терминала сбора данных (например, находиться в другой подсети), отладка Mobile SMARTS не может сама принять решения о том, какой IP правильный, и просит пользователя сделать соответствующий выбор:

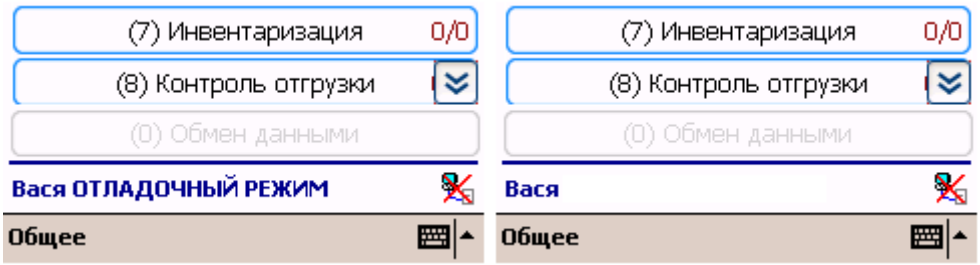


Внимание! Если при установке клиента Mobile SMARTS была изменена папка, в которую выполняется установка по умолчанию, перед запуском отладки следует зайти в настройки Редактора метаданных Mobile SMARTS и в поле **Путь к папке на терминале** ввести название папки, в которую был установлен клиент Mobile SMARTS на ТСД:

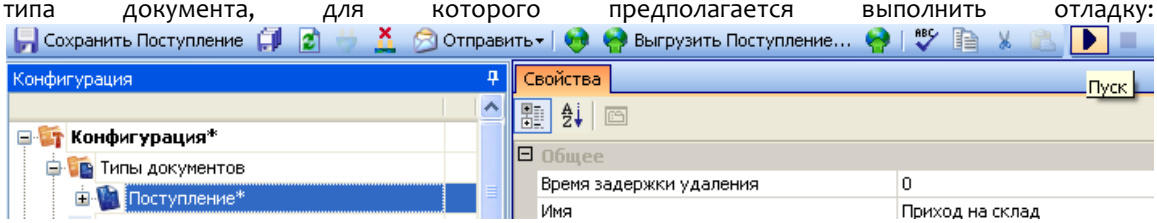


Для запуска отладки необходимо выполнить следующие шаги:

1. Установить связь между ТСД (либо эмулятором Windows CE) и компьютером через AcrtiveSync или Центр мобильных устройств. На ТСД предварительно должен быть установлен клиент Mobile SMARTS. Подробно данный шаг описан в Руководстве по установке клиента Mobile SMARTS. Если на ТСД уже выполняется клиент Mobile SMARTS, запущенный не в отладочном режиме, из него следует выйти. При выполнении клиента в отладочном режиме в нижней части окна выбора операции отображается надпись ОТЛАДОЧНЫЙ РЕЖИМ, иначе этой надписи нет:

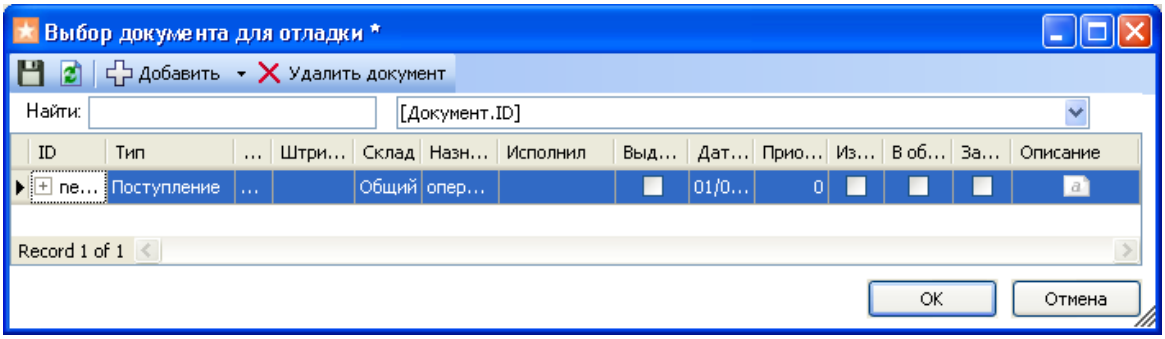


2. В редакторе метаданных Mobile SMARTS выбрать в дереве операций любой узел, дочерний узлу типа документа, для которого предполагается выполнить отладку:



и панели инструментов нажать кнопку **Пуск** ();

3. Появится окно со списком документов выбранного типа. Необходимо выбрать существующий документ или добавить новый, выбрать его и нажать ОК:



4. Должен произойти запуск клиента Mobile SMARTS и начаться обработка выбранного документа.

Для того, чтобы запустить на отладку этот же или другой документ еще раз, нужно просто нажать кнопку **Пуск**. Выходить из клиента Mobile SMARTS на терминале сбора данных нет необходимости. Весь смысл механизма отладки в том, чтобы не тратить время на повторные запуски Mobile SMARTS на ТСД, а запускать сразу интересующую операцию в одном и том же уже запущенном клиенте Mobile SMARTS.

Выполнение отладки

Отладка заключается в выполнении процесса обработки документа Mobile SMARTS на ТСД, при этом есть возможность просмотра данных сессии, данных самого документа, лога клиента Mobile SMARTS, нажатых на ТСД клавиш, отправки событий нажатия клавиш на ТСД из отладчика, приостановки отладки в произвольный момент времени. Окно редактора метаданных Mobile SMARTS в режиме отладки приведено.

Тип документа, для которого выполняется в данный момент отладка, подкрашен розово-красным цветом, действие, выполняемое в данный момент на ТСД – песочно-желтым.

Для управления процессом отладки предназначены следующие кнопки:

- **Пуск.** Запуск отладки;
- **Шаг.** Запуск отладки по шагам/выполнение шага. При этом клиент Mobile SMARTS будет выполнять действия по одному, ожидая нажатия клавиш **Шаг** или **Пуск** для перехода к следующему действию;
- **Шаг без захода.** Запуск отладки по шагам/выполнение шага без захода в группы действий.
- **Стоп.** Прерывает процесс выполнения документа клиентом Mobile SMARTS (Рис 7). Выполняется переход к окну выбора операций.

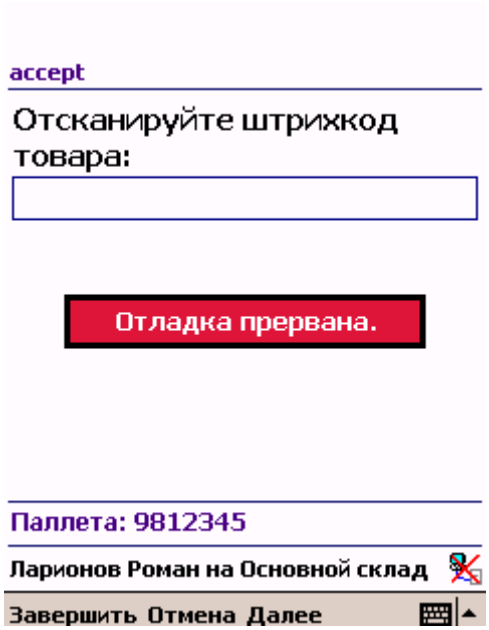


Рис. 7

- **Установить/снять точку останова**. Устанавливает/снимает точку останова на выбранном в дереве действии. Выбранное действие подкрашено синим (Рис. 8).

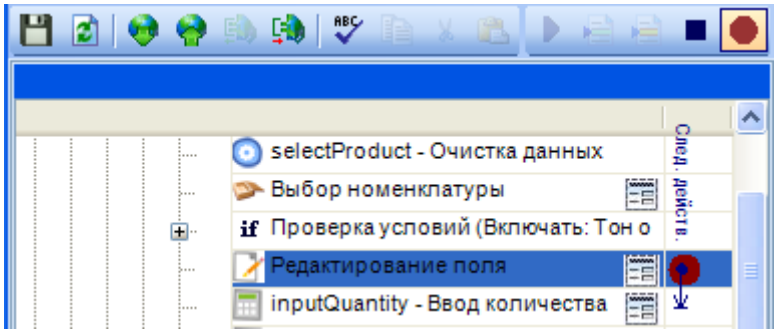


Рис. 8

Выполнение документа клиентом Mobile SMARTS всегда приостанавливается на действии, для которого стоит точка останова. Выполнение и переход к следующему действию не выполняется до нажатия на одну из клавиш: **Пуск, Шаг, Шаг без захода**.

В процессе отладки возможно изменение свойств действий, в том числе, выполняющегося в данный момент действия, при этом изменения автоматически передаются клиенту Mobile SMARTS.

Назначение закладок:

- **Лог терминала**. В поле **Терминал** отображается адрес соединения с клиентом Mobile SMARTS на ТСД, в поле **Лог** выводятся сообщения, попадающие в лог-файл клиента Mobile SMARTS во время отладки;
- **Сессия** (Рис. 9). В виде таблицы «Ключ-Значение» отображаются данные сессии в текущий момент. Есть возможность изменения данных, добавления новых значений и удаления имеющихся. Для того чтобы сделанные изменения попали в клиент Mobile SMARTS на ТСД, следует нажать кнопку **Выгрузить**;

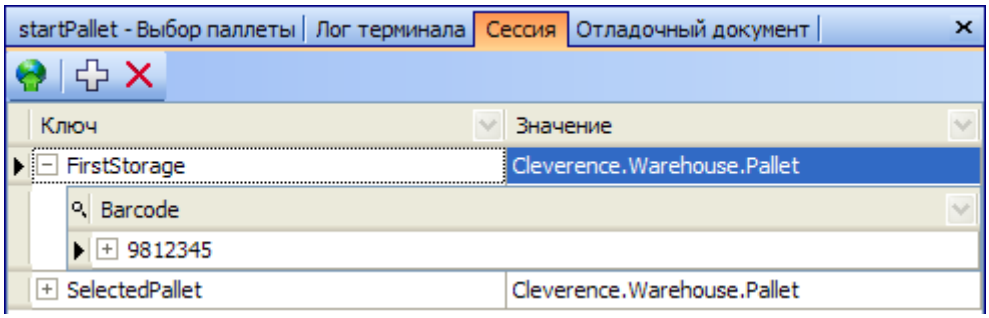
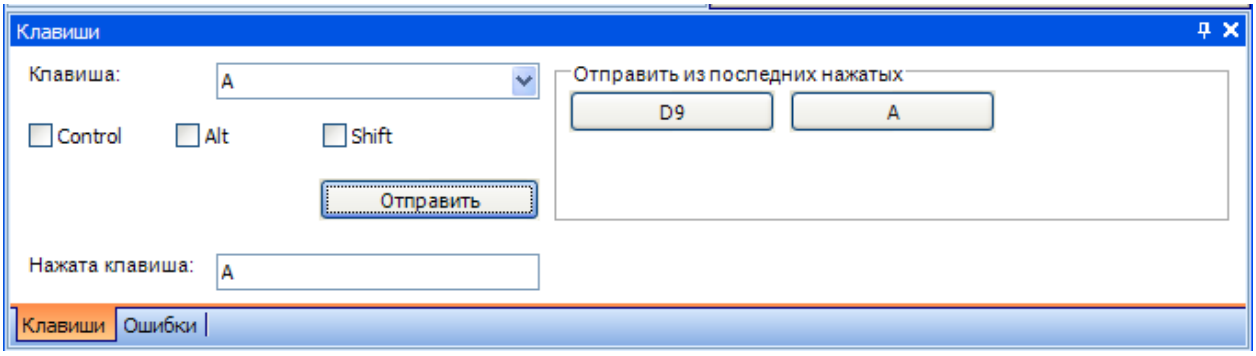


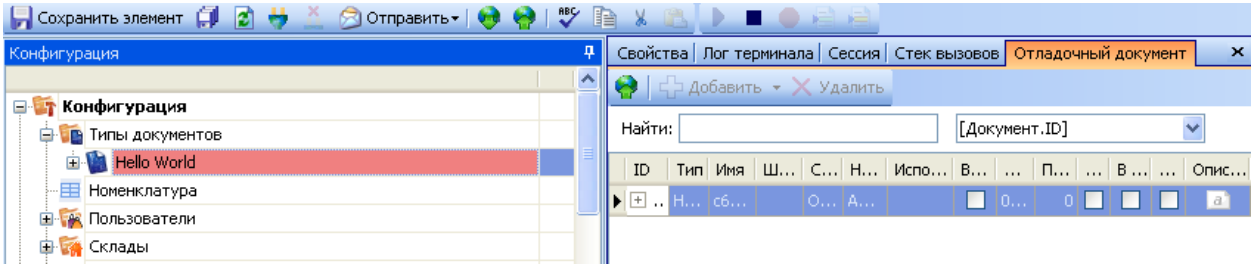
Рис. 9

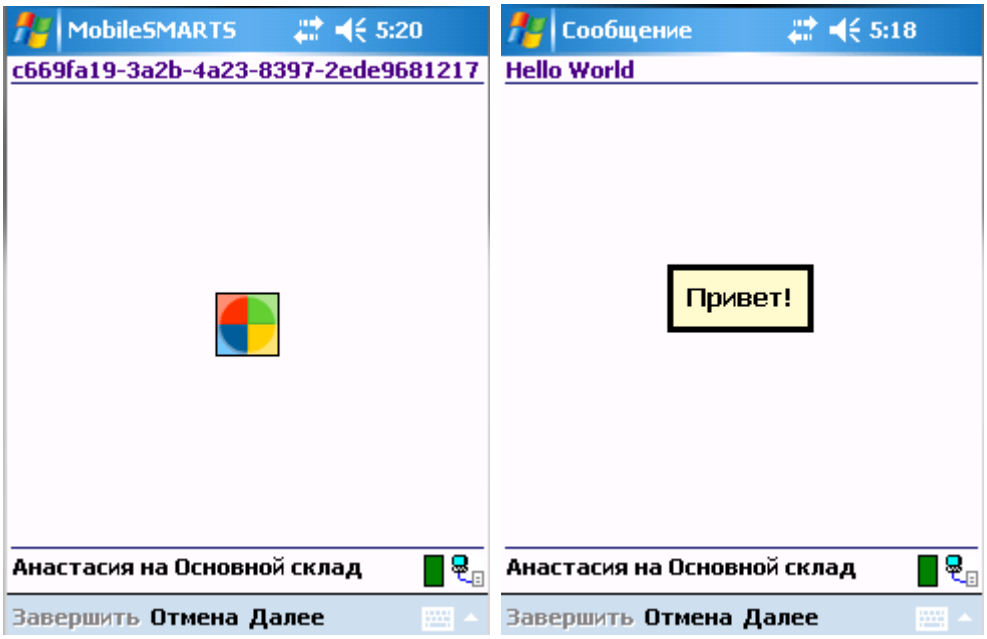
- **Отладочный документ**. В виде таблицы отображаются данные обрабатываемого в данный момент на ТСД документа. Есть возможность изменения данных документа. Для того чтобы сделанные изменения попали в клиент Mobile SMARTS на ТСД, следует нажать кнопку  **Выгрузить;**

- **Клавиши.** В поле **Нажата клавиша** отображаются нажатые на ТСД клавиши. Чтобы отправить на ТСД событие о нажатии клавиши нужно из списка **Клавиши** выбрать нужную клавишу, если вместе с требуемой клавишей должна быть нажата одна из клавиш **Control, Alt, Shft**, следует проставить соответствующий флажок, и затем нажать кнопку **Отправить.** Для ускорения доступа к последним нажатым клавишам предназначены кнопки справа от списка клавиш:



Наконец, отладка запускается:



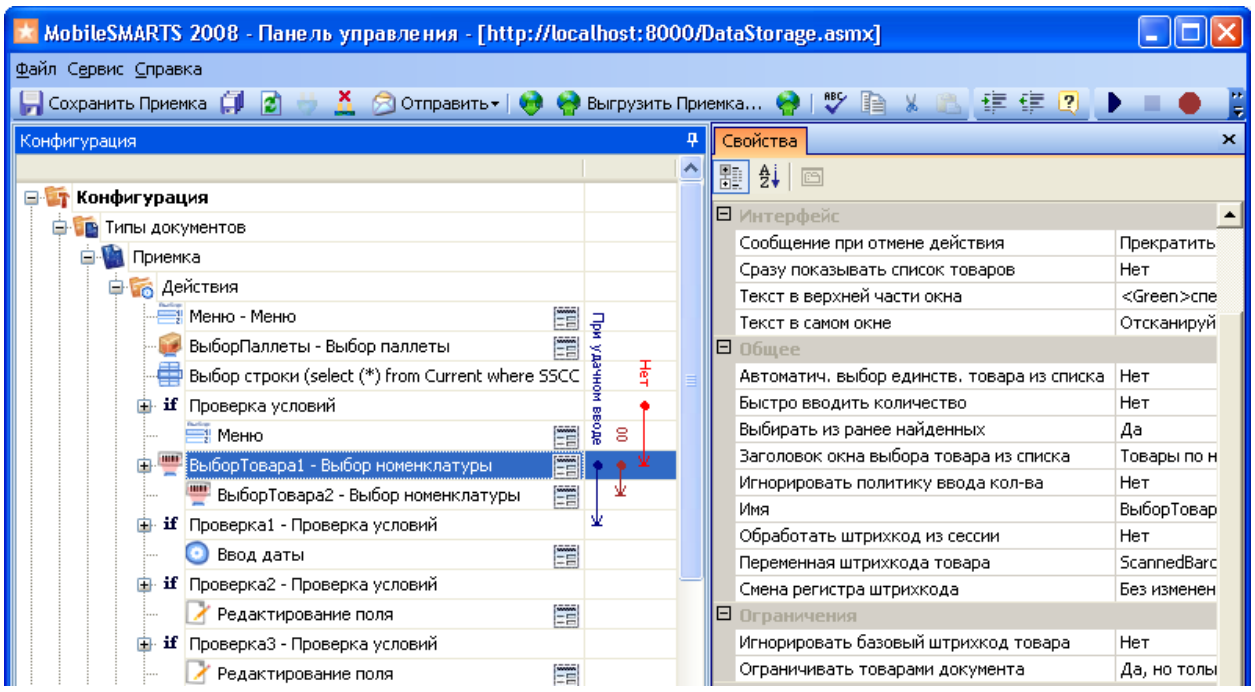


Простейшая программа для Mobile SMARTS успешно создана и запущена на исполнение.

§ 5. Настройка свойств действий в Mobile SMARTS – Часть 1

Программирование в Mobile SMARTS во многом сводится к настройке свойств действий, из которых строится программа работы на терминале сбора данных.

Визуальные и не визуальные действия



Значок «  » напротив некоторых действий означает, что такое действие является *визуальным*. Все доступные действия в Mobile SMARTS делятся на *визуальные* и *не визуальные*. Визуальные действия являются модальными, т.е. требуют от пользователя какого-то действия – ввода данных, выбора чего-то из списка и т.д. Не визуальные ничего не требуют, и выполняются одно за другим до тех пор, пока не

наступит очередь визуального действия или документ не закроется. Соответственно, визуальные и не визуальные действия по-другому можно назвать как «с окошками» и «без окошек».

Практически все визуальные действия содержат свойства, задающие пользовательский интерфейс на терминале – например, «Текст в верхней части окна», «Текст в самом окне» и т.д.

Стрелочки с подписями отражают направление перехода по событиям (таким как успешное считывание штрихкода, выбор пункта меню или нажатие определенной клавиши).

Стек действий

В процессе исполнения операций визуальные действия, а также действие группировки действий, могут складываться в *стек действий*. Смысл стека действий в следующем:

1. предоставить оператору ТСД возможность возвращаться к действиям, которые он уже выполнял (экранам, на которых он только что был), для того, чтобы что-то там поправить;
2. предоставить возможность вызова каких-то последовательностей действий как подпрограмм – например, вызов меню по клавише F1, и возврат к месту, откуда меню было вызвано, по клавише Escape.

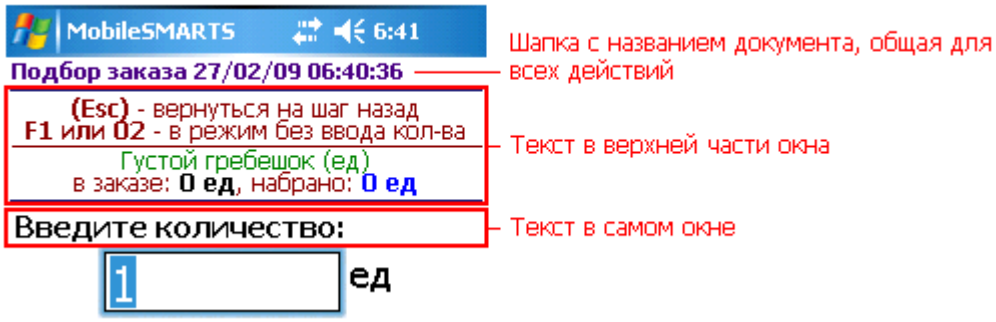
   Действие окажется на стеке в том случае, если её свойство «Запомнить на стеке» будет равно «Да». Для навигации по стеку в Mobile SMARTS предусмотрен механизм отмены действия и специальные зарезервированные команды перехода, такие как return, abort и т.д.

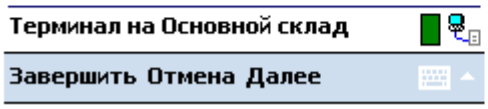
Свойства, общие для всех действий

Единственным свойством, общим для всех действий, является имя конкретного экземпляра действия в дереве.

	Имя	Имя. Имя действия используется для указания перехода на него. Например, при неудачном сканировании товара можно сделать переход на какое-либо действие. Для этого в соответствующем свойстве «При ошибке ввода» действия выбора номенклатуры нужно указать имя того действия, на которое следует переходить. Имя можно не указывать. В этом случае на такое действие нельзя задать явного перехода по какому-либо событию или условию.

Свойства, общие для всех визуальных действий



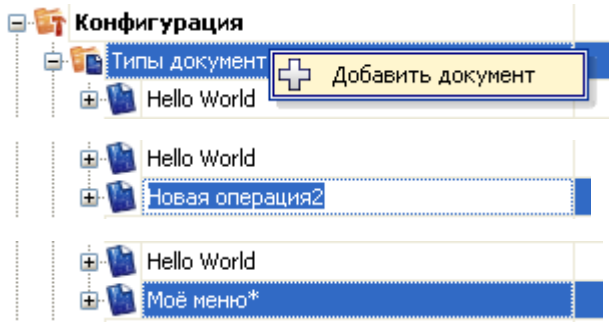


	Текст в верхней части окна	Строка. Информационный текст, отображаемый в заглавии страницы при выполнении действия. могут применяться шаблоны. Подробнее смотрите Руководство разработчика.
	Текст в самом окне	Строка. Вступительный текст.
	Полноэкранный режим	Зависит от старшего | Да | Нет. Определяет выводить ли окно действия в полноэкранном режиме (без верхних и нижних панелей, меню и т.д.). Полноэкранный режим означает, что не будут отображаться шапка с названием документа и нижние панели, которые в редакторе метаданных называются «Строки подвала». По умолчанию зависит от старшего, т.е. действия, находящиеся в группе, берут значение полноэкранного режима у группы, та у своей группы и в конечном итоге у типа документа.
	Полноэкранный режим (значение)	Да/Нет. Отображает будет ли действие выводиться в полноэкранном режиме с учетом указания на полноэкранный режим для родительских действий и типа документа.
	Запомнить на стеке	Да/Нет. Флаг, указывающий, следует ли сохранять состояние сессии перед выполнением данного действия. Позволяет организовывать правильные алгоритмы отмены выполнения действий.
	Глубина отмены действия	Нет | На одно действие | К началу | Возврат из подпрограммы | Весь документ. Тип отмены действия.
	Сообщение при отмене действия	Строка. Текст сообщения, выдаваемого на экран при отмене действия. Если не задан - сообщение не выводится.

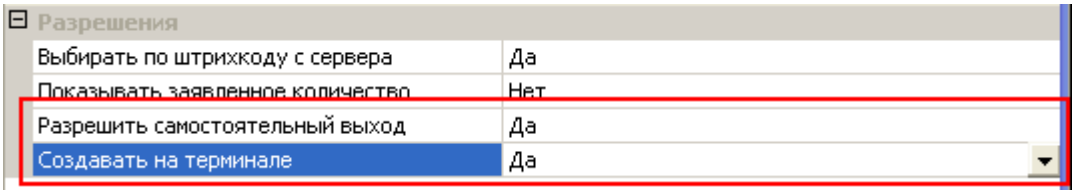
§ 6. Пример меню выбора на терминале сбора данных

Рассмотрим еще один простой пример, в котором на экране терминала сбора данных выводится некоторое меню с вариантами действия.

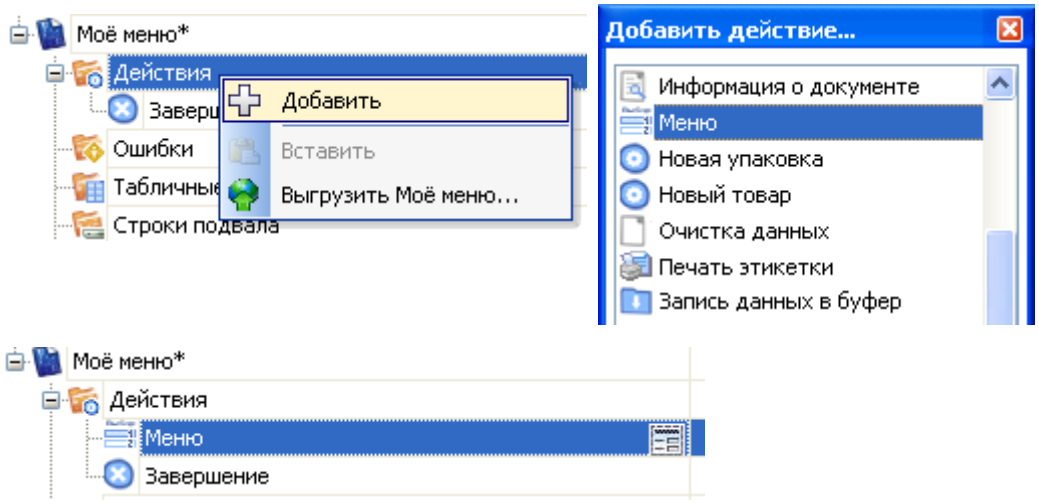
Для примера с меню создадим новый тип документа:



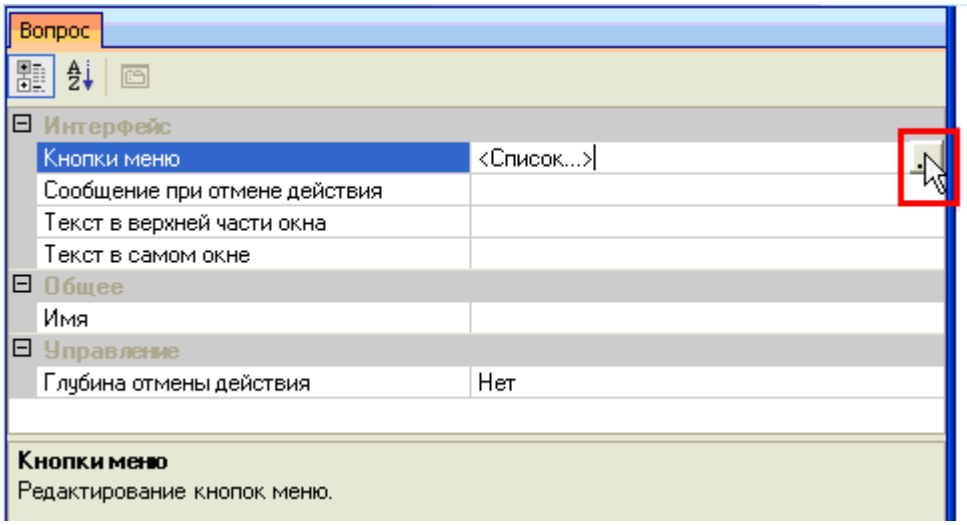
В отличие от предыдущего простого примера, этот тип документа не будет виртуальным. Соответствующие свойства типа документа должны разрешать нам не только создавать документы вручную на терминале, но и выходить из них в любой момент:



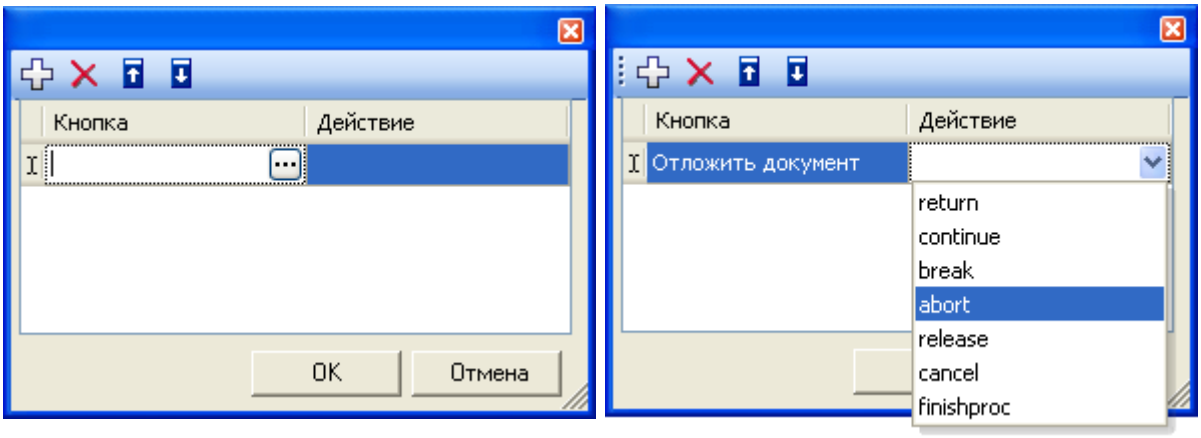
Для вывода меню в Mobile SMARTS предусмотрено специальное действие, которое так и называется – «Меню»:



Среди свойств меню есть «Кнопки действия», которое содержит список кнопок.



Для каждой кнопки в списке требуется обязательно указать текст на ней и действие, которое следует выполнить по нажатию.



Значение системных действий return, abort и т.д.

В выпадающем списке для действия кнопки отображаются возможные варианты перехода по нажатию такой кнопки. Вначале этот список заполнен системными действиями:

return – выход из документа, документ считается выполненным;

continue – начать выполнение текущей группы действий с начала;

break – прервать текущую группу действий;

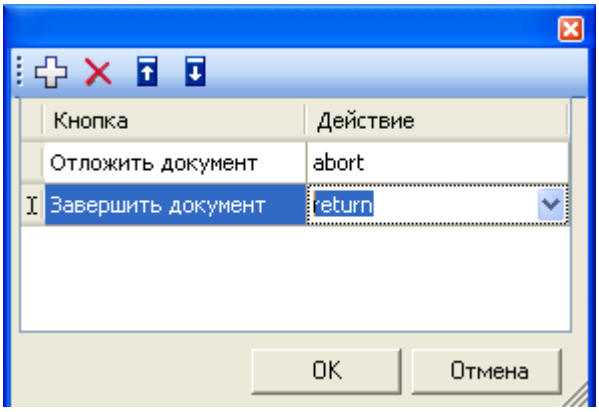
abort – выход из документа, документ считается прерванным;

cancel – отмена изменений по текущему действию и возврат к предыдущему в стеке;

finishproc – возврат к предыдущему действию в стеке без отмены произведенных изменений.

Разница между прерыванием и завершением документа состоит в том, будет ли выставляться флажок «Завершен» в документе (свойство Finished).

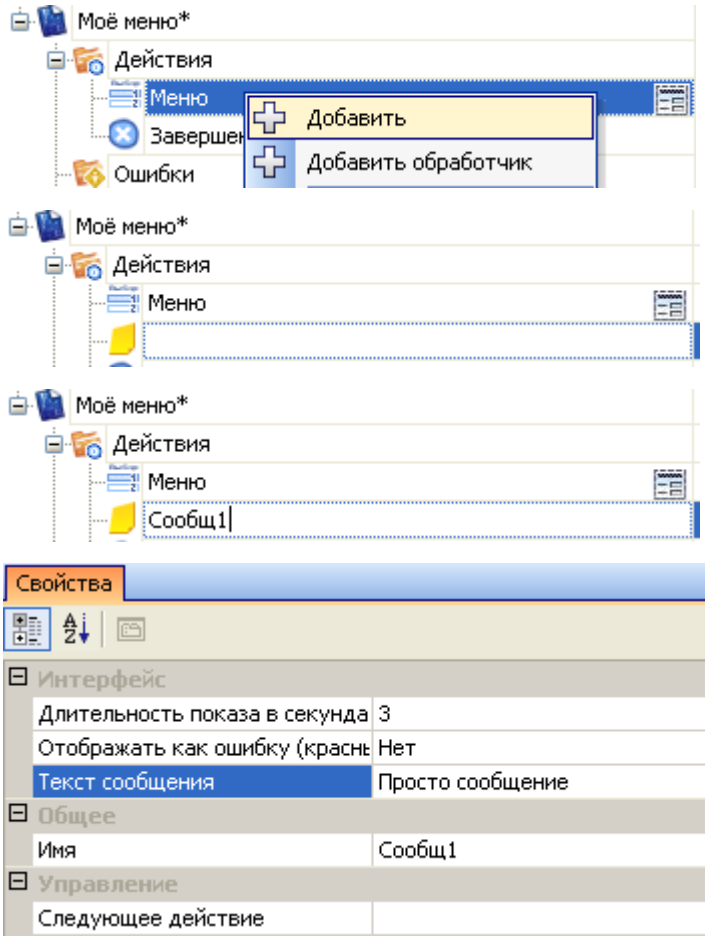
Заполним список следующим образом:



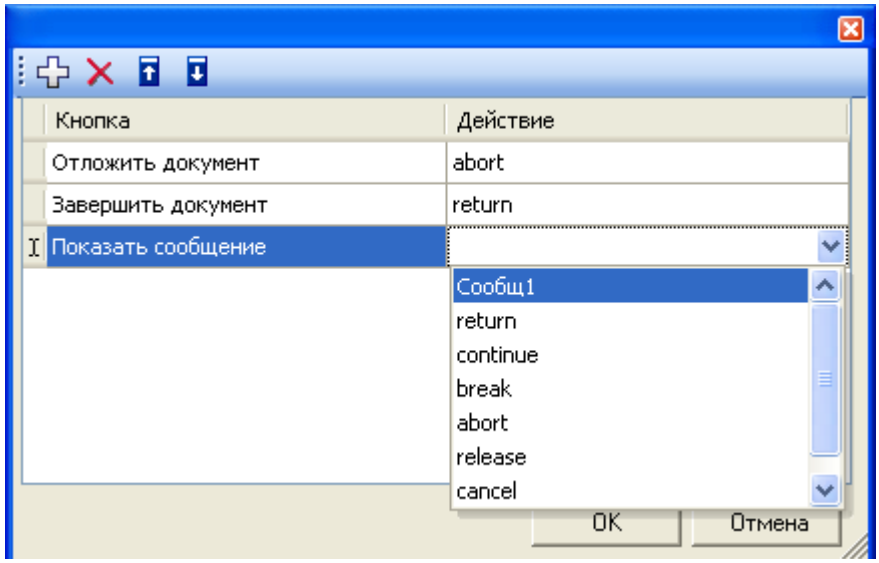
Переходы по действиям в дереве

Чтобы в списке появилось больше вариантов, необходимо назначать действиям в дереве имена. Эти имена попадут в список и, соответственно, при нажатии по кнопке будет происходить переход к именованному действию.

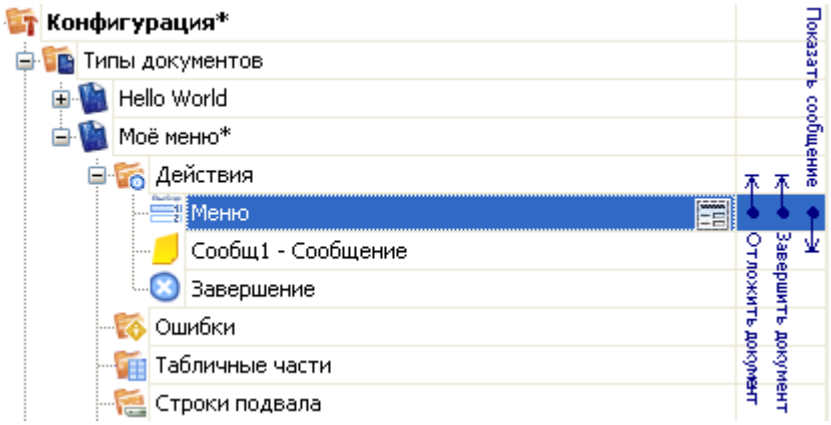
Для примера перехода к действию с именем вставим в дерево показ сообщения:



Теперь в меню можно добавить кнопку с переходом для показа сообщения:

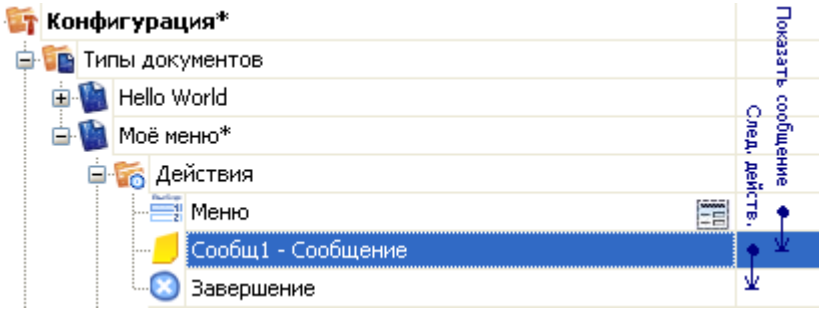


Если теперь щелкнуть в дереве по действию «Меню», можно увидеть стрелочки, объясняющие работу кнопок:

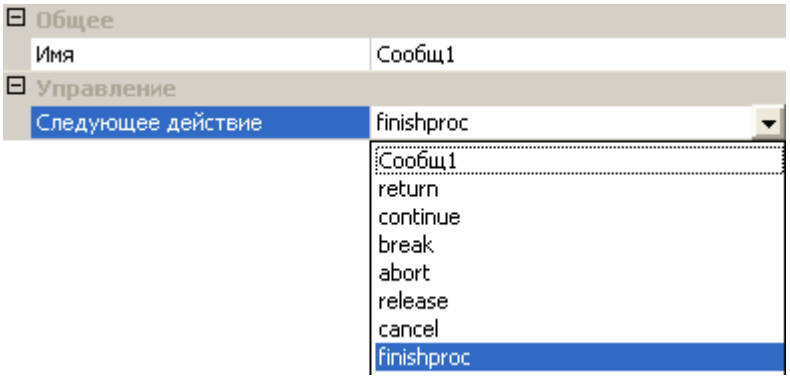


Эти стрелочки всегда показывают, откуда и куда совершаются переходы между действиями в дереве, помогая ориентироваться в работе программы обработки документа.

Если следом щелкнуть по действию «Сообщ1», то будет видно, что в сообщение мы попадаем из меню по кнопке «Показать сообщение», а после показа сообщения исполнение перейдет к действию «Завершение» и документ будет завершен:

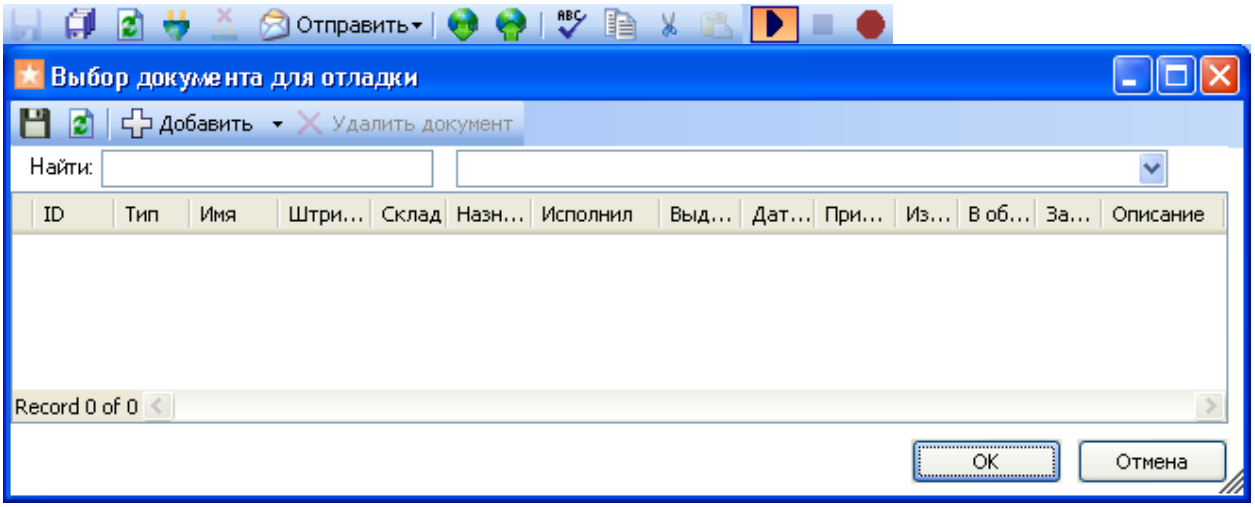


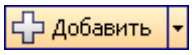
Если удалить теперь действие «Завершение», то после показа сообщения документ всё равно будет завершен, т.к. обработка дойдет до конца дерева. Чтобы показ сообщения возвращался в меню, следующим действием для «Сообщ1» следует указать «cancel» или «finishproc»:



Запуск меню на отладку

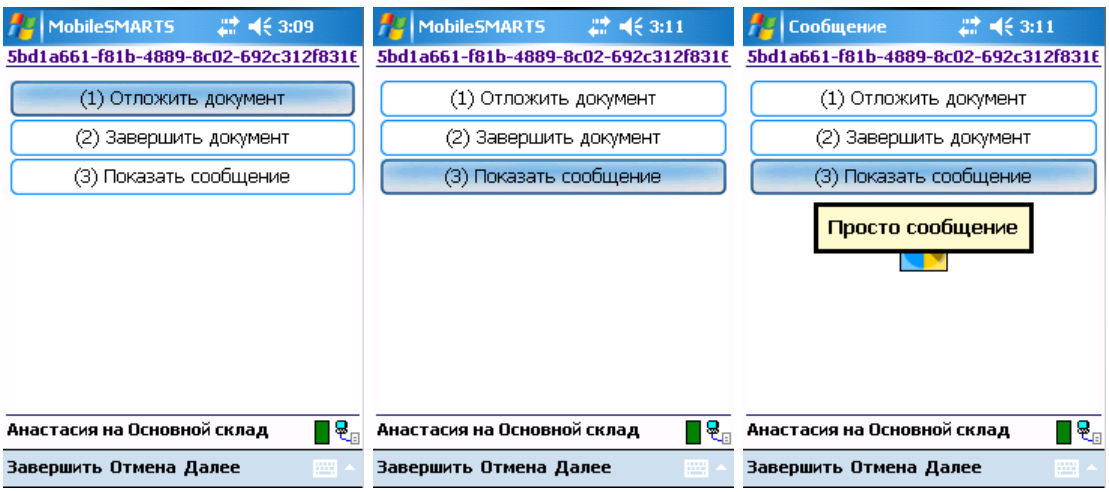
При запуске нового типа документа на отладку панель управления попросит выбрать экземпляр документа, на котором будет выполняться отладка:



Документов в списке нет, и следует 

новый.

После запуска программы видно, как клиент Mobile SMARTS на терминале сбора данных отобразил меню из кнопок. Кнопки выровнены вертикально, как и в списке в панели управления, занимают по ширине почти всё окно терминала. В скобках сбоку от надписи показаны автоматически назначенные горячие клавиши – цифры от 1 и далее, которые позволяют нажать кнопку с цифровой клавиатуры терминала сбора данных, не выбирая её в списке стрелочками и не используя стилус:



Если выбрать «Отложить документ» или «Завершить документ», то будет проиграна мелодия окончания обработки документа и документ будет закрыт, а отладка завершена.

§ 7. Пример сканирования товара с суммированием

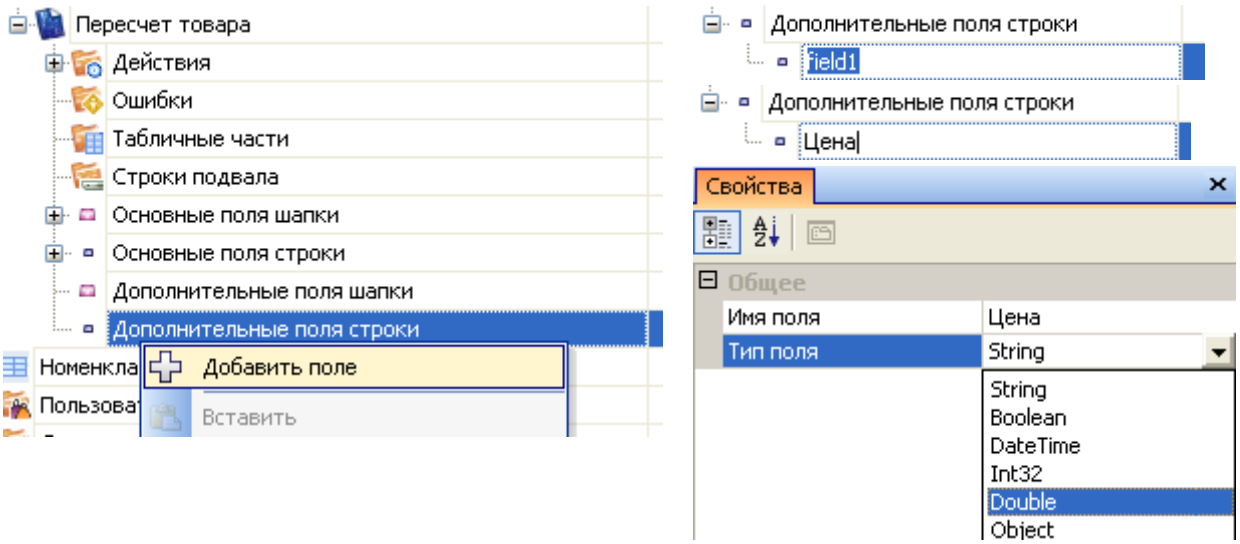
Рассмотрим следующий пример, в котором сканируемые штрихкоды попадают в документ, а на экране терминала сбора данных выводится общая стоимость сканированных позиций.

Для создания примера добавим новый тип документа «Пересчет товара»:

Общая стоимость пересчитанного товара слагается из стоимостей отдельных позиций. Для начала реализуем самый простой пример, в котором не предусмотрено справочника цен, соответствующих тем или иным штрихкодам, т.е. терминал сбора данных должен будет спрашивать цену каждой сканируемой позиции. Введенная цена будет сохраняться в соответствующей колонке документа.

Добавление в документ новых колонок и строк

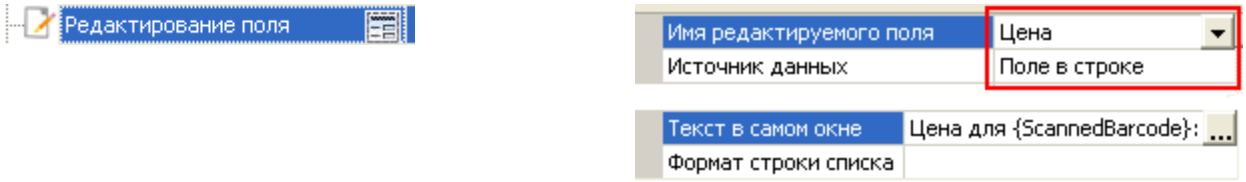
Документ в Mobile SMARTS по умолчанию содержит две табличные части одинакового формата. Каждая табличная часть состоит из строк, соответственно колонки называются *полями строки документа*. Цена сканированной позиции будет являться дополнительным полем строки и иметь тип Double (т.е. вещественное число с двойной точностью):



В Mobile SMARTS для всех случаев сканирования, выбора из списка или любого другого ввода товара в систему следует использовать действие «Выбор номенклатуры», поскольку оно заполняет

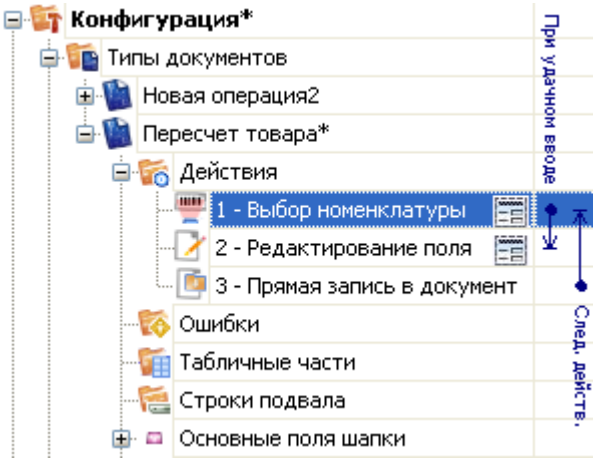
переменную сессии «SelectedProduct», которая (переменная) необходима для занесения строки в документ. В этой переменной содержится своеобразный «проект» новой строки.

Вторым действием следует ввести цену позиции при помощи действия «Редактирование поля»:



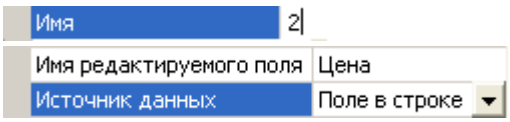
Следующим действием после выбора номенклатуры следует добавить действие «Прямая запись в документ».

Действие «Завершение» лучше удалить. Хотя действие завершения удалено, после выполнения действия «Прямая запись…» программа обработки документа всё равно будет завершена, т.к. выполнение достигнет последнего действия в дереве. Чтобы документ не завершался сам собой, нужно назначить действиям имена и переходы. В результате должно получиться следующее дерево:

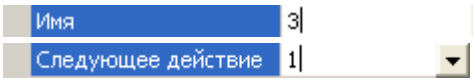


Для выбора номенклатуры:

Для редактирования поля:



Для записи в документ:



Стрелки в дереве показывают, в каких вариантах, откуда и куда будут совершаться переходы для каждого конкретного выделенного действия – в данном случае для выбора номенклатуры.

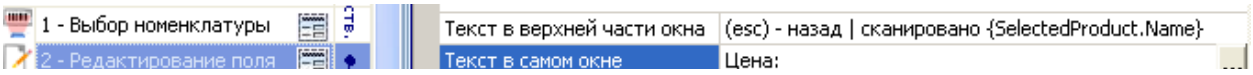
Отображение информации в окнах ввода данных

Было бы полезно, если бы в окне ввода цены можно было увидеть, для какого распознанного товара или штрихкода вводится эта цена.

Программирование вывода информации в Mobile SMARTS максимально упрощено. Для этого очередные так называемые *шаблоны*, в которых всё, что заключено в фигурные скобки «{}», обрабатывается специальным образом. Подробнее об этих шаблонах можно почитать в разделе «Шаблоны текстов и математических выражений».

Например, если в тексте указано «Сканировано {ScannedBarcode}», то часть «{ScannedBarcode}» всегда будет заменяться значением соответствующей переменной сессии. Переменная ScannedBarcode используется действием выбора номенклатуры для хранения сканированного штрихкода. Таким образом, на экране отобразится только что сканированный штрихкод, допустим так: «Сканировано 9771729040004».

Для текстов в окне редактирования цены можно использовать следующий подход: в заголовке окна (мелким шрифтом) дать справочную информацию, а в самом окне (крупным шрифтом) – указания по текущему этапу операции. Например, так:

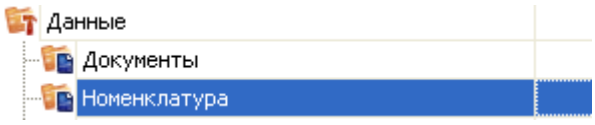


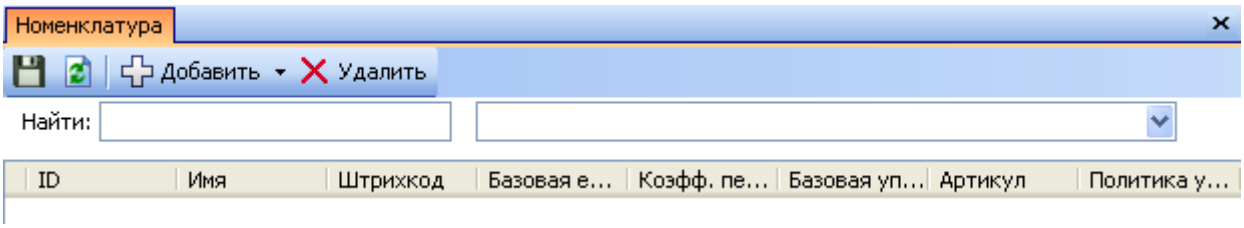
Текст «(esc) - назад» напоминает, что по нажатию кнопки Escape можно вернуться к предыдущему действию, на сканирование штрихкода. Часть «{SelectedProduct.Name}» каждый раз будет заменяться на название только что сканированного товара. В итоге будет получаться, например, «(esc) – назад | сканировано Миксер BINATONE HM 212,6 скор. 150вт».

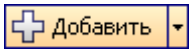
Редактирование справочника номенклатуры в панели управления

По умолчанию действие выбора номенклатуры не пропускает штрихкоды, не зарегистрированные в справочнике номенклатуры. Если встречается такой штрихкод, действие пытается вывести сообщение об ошибке и перейти к действию, обрабатывающему ошибку сканирования. Текст сообщения берется из свойства «При ошибке ввода» группы «Тексты сообщений об ошибках», а действие перехода берется из одноименного свойства «При ошибке ввода» группы «Управление». Если текст пуст, сообщения не выводится, а если пуст адрес перехода, то управление возвращается к выбору номенклатуры, именно поэтому по умолчанию неизвестные штрихкоды и не попадают в документ.

Чтобы иметь возможность добавлять в документ данные о сканированных штрихкодах, в Mobile SMARTS необходимо завести номенклатуру. Для этого в панели управления предусмотрен специальный редактор:





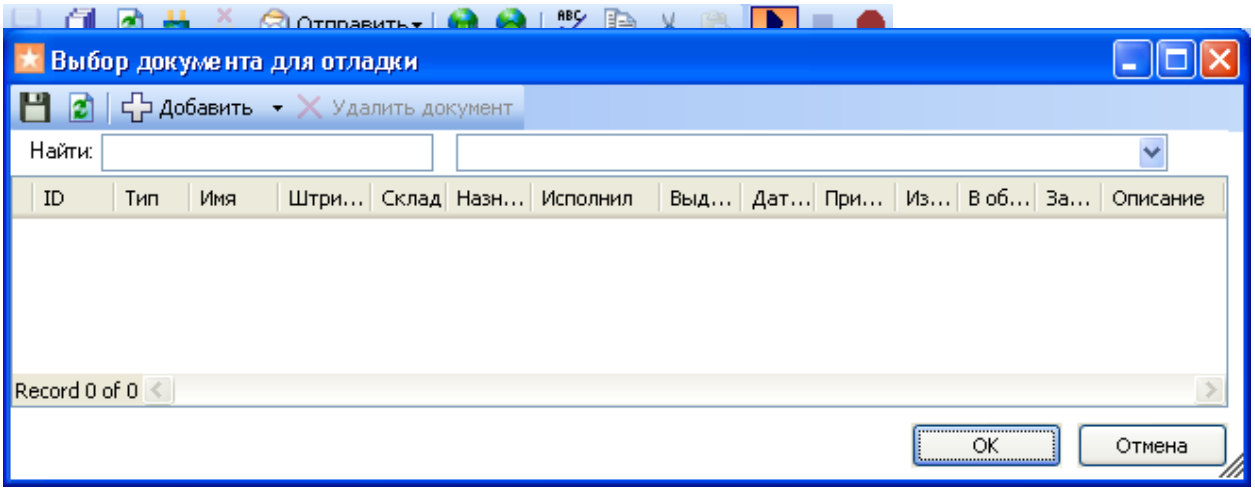
В нём следует 

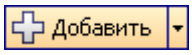
новую номенклатуру и указать штрихкоды.

Для каждой номенклатуры обязательно должна существовать как минимум одна упаковка номенклатуры, которая отражает собственно вариант упаковки с весом и коэффициентами. Штрихкод можно указать как для номенклатуры, так и для упаковки. Одна из упаковок обязательно указывается как *базовая упаковка*. Каждая строка документа обязана содержать и код товара, и код упаковки (либо ни того, ни другого) – поэтому при сканировании штрихкода, который указан для самой номенклатуры, выбранной упаковкой считается базовая.

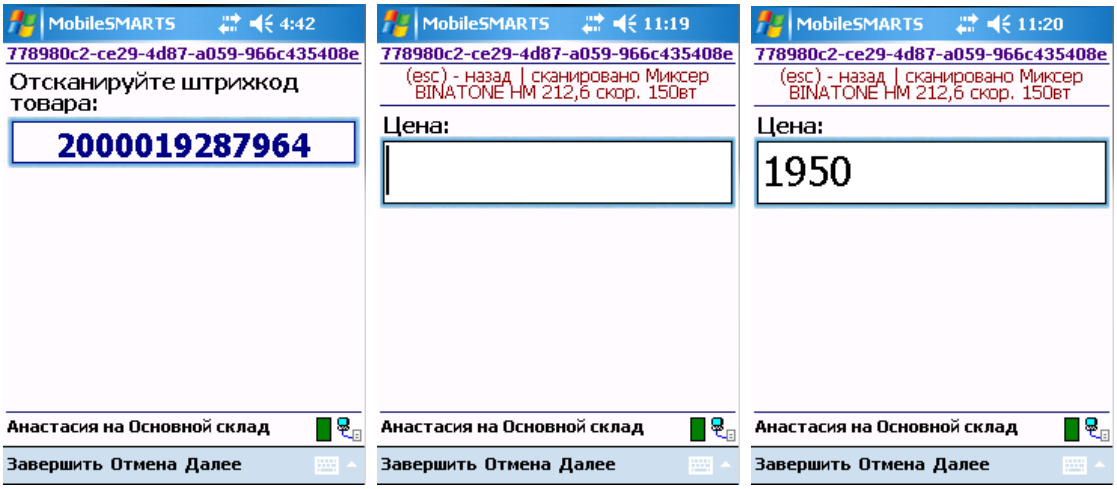
Запуск пересчета товаров на отладку

При запуске нового типа документа на отладку панель управления попросит выбрать экземпляр документа, на котором будет выполняться отладка:



Документов в списке нет, и следует  новый.

После запуска на терминале сбора данных получается следующее:



Примечание: Благодаря отладчику Mobile SMARTS любые ошибки и недоделки в настройках действий можно исправлять прямо во время исполнения программы на терминале сбора данных и тут же видеть результат. Например, изменить текст в заголовке и тут же увидеть, как он поменяется на экране.

Обсуждение необходимых улучшений программы

Полученная программа работает непонятно – штрихкоды сканируются, на неизвестные выдается звук ошибки, на известные – звук успеха и окно ввода цены. Однако в целом непонятно, добавляются ли строки. Кроме того, название товара в шапке окна выглядит невыразительно.

Добавляются ли строки можно увидеть в окне отладчика на закладке «Отладочный документ», однако для превращения программы из примера в практически полезную операцию, требуется внести в неё как минимум следующие изменения:

— сделать просмотр отсканированных строк;

— при сканировании позиции показывать, сколько уже отсканировано такого же товара и на какую сумму;

— показывать, сколько вообще уже отсканировано товаров и на какую сумму.

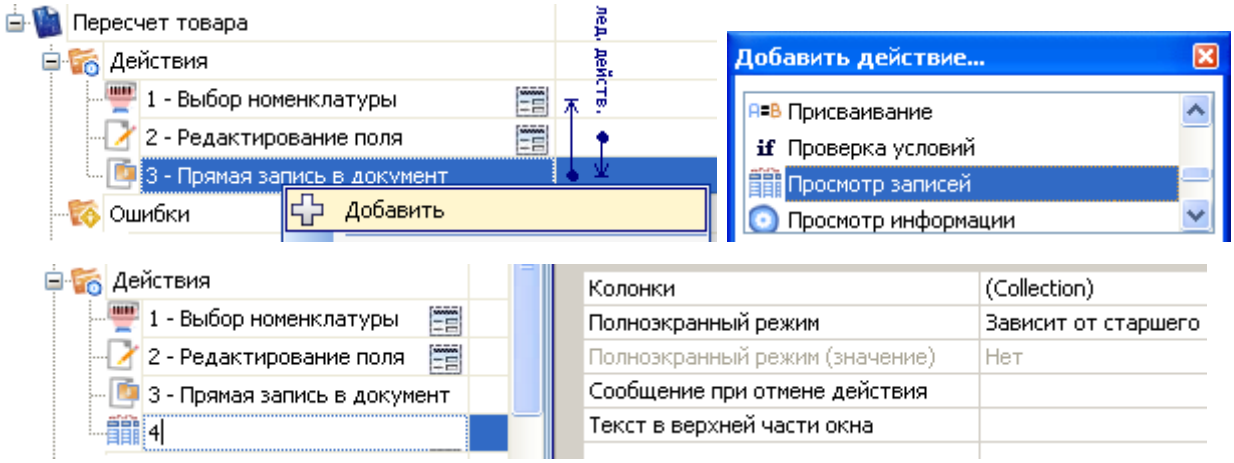
Каждое улучшение требует отдельного рассмотрения, т.к. всё это часто используемые доработки практически любой складской операции, и объяснение способов реализации их в Mobile SMARTS позволяет лучше понять логику системы.

Добавление просмотра строк документа

Для просмотра строк чего-нибудь в Mobile SMARTS существует специальное действие, которое почти так и называется – «Просмотр записей».

Настройкой свойств «Просмотра записей» можно указать источник данных для просмотра, отфильтровать и отсортировать этот источник, а также задать внешний вид таблицы для отображения, включая формат каждой строки.

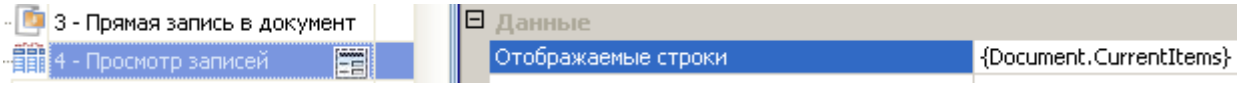
Добавим «Просмотр записей» в самый конец дерева действий:



В свойствах «Просмотра записей» можно найти настройки как источника данных, так и параметров их отображения. Первые объединены в группу «Данные», а вторые – в группу «Интерфейс».

Набранные строки оказываются в коллекции CurrentItems документа. Существует два способа задать источник данных для отображения строк: напрямую или через select-запрос. Напрямую данные указываются в свойстве «Отображаемые строки», а select-запрос – в свойстве «Запрос». Соответственно, чтобы задать строки CurrentItems в качестве источника данных, нужно указать «{Document.CurrentItems}» либо в свойстве «Отображаемые строки» действия, либо в свойстве «From» запроса. Если нет планов фильтровать или суммировать что-то в строках перед отображением их в окне, запросом пользоваться не обязательно.

Укажем в свойстве «Отображаемые строки» значение «{Document.CurrentItems}»:

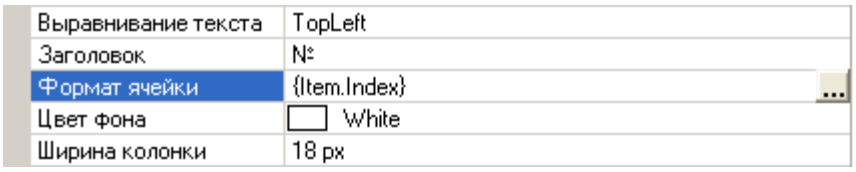


Настройки самих окошек и отображения задаются свойствами из группы «Интерфейс». Помимо стандартных для визуального действия свойств «Текст в верхней части окна» и «Текст в самом окне», у действия отображения записей есть свойство «Колонки». Это те колонки, которые будут визуально формировать таблицу.

Для каждой колонки таблицы можно задать заголовок, рамки, цвета и другие параметры отображения. Ширина колонки может быть задана либо фиксировано в пикселах, либо в процентах от той ширины окна, которая останется не занятой колонками с фиксированной шириной.

Самое главное свойство колонки – «Формат ячейки». Оно содержит текстовый шаблон, по которому для каждой отображаемой строки в таблице будет формироваться содержимое ячеек по данной колонке. Заполнение таблицы под отображение происходит следующим образом: пробегаясь по объектам из источника данных (например, по строкам из табличной части документа), действие «Отображение записей» кладет в сессию каждый по очереди под именем «Item», затем для каждой ячейки в строке обрабатывается текстовый шаблон соответствующей ей колонки и полученный текст попадает в ячейку. Соответственно, чтобы шаблон выбирал данные именно из текущей строки, он должен начинаться на «{Item.».

Создадим следующие колонки:

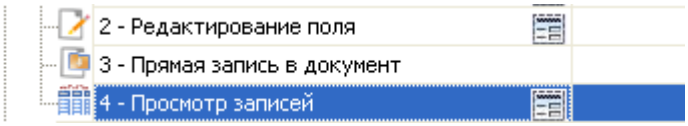
1я – № п/п:	 Выравнивание текста = TopLeft Заголовок = № Формат ячейки = {Item.Index} Ширина колонки = 18px
2я – описание:	Выравнивание текста = TopLeft Заголовок = «Наименование» Формат ячейки = «{Item.Packing.Barcode} - {Item.Product.Name}» Ширина колонки = 100%
3я – количество:	Выравнивание текста = TopCenter Заголовок = «Наименование» Формат ячейки = «{Item.CurrentQuantityString}» Ширина колонки = 40px
4я – стоимость:	Выравнивание текста = TopCenter Заголовок = «Итог» Формат ячейки = «{Item.Цена:(0:0.00)р.} » Ширина колонки = 60px

Как видно, первые две колонки выровнены по левому краю, количество – по центру, а колонка стоимости – по правому. Кроме того, в шаблонах присутствует форматирование размера шрифта, а также «» и цвет, чтобы сделать отображение **жирным и цветным**.

Минус « - » перед строкой «{Item.Product.Name}» в шаблоне для наименования – это не минус, а просто черточка. Наименование товара не будет вычитаться из штрихкода, а просто они будут показаны одно друг за другом: жирный штрихкод, черточка и нежирное наименование.

Зато для цены указан не просто путь к значению, «Item.Цена», но также двоеточие и строка форматирования «(0:0.00)р.». Строка форматирования гласит: «Возьми строку 0.00, отдай её значению пути в качестве .NET формата, а потом замени (0:0.00) тем, что получится на выходе». Например, для суммы 440 результатом будет «440.00р.», а если после первого двоеточия в шаблоне написать не «(0:0.00)р.», а «привет! (0:00.0)», то результатом будет «привет! 440.00». Подробнее о шаблонах следует читать в соответствующем разделе документации.

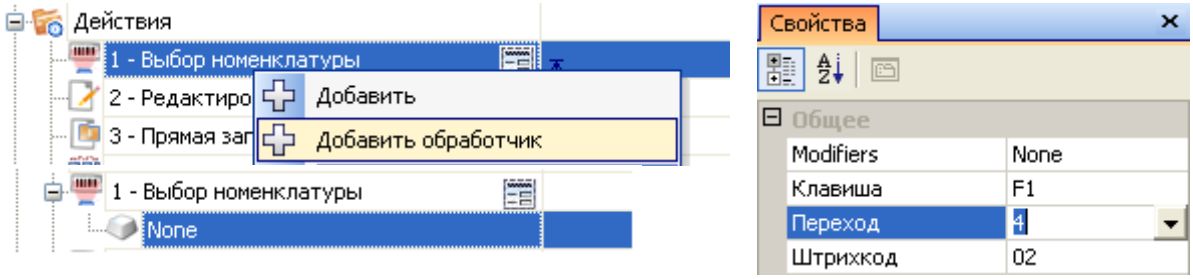
Запускать операцию на отладку пока рано, ведь в дереве нет ни одного перехода на новое действие, и до него никогда не дойдет очередь исполнения. Это хорошо видно, если выделить в дереве новое действие и посмотреть на стрелочки перехода:



Ни одной стрелочки нет, соответственно не и переходов.

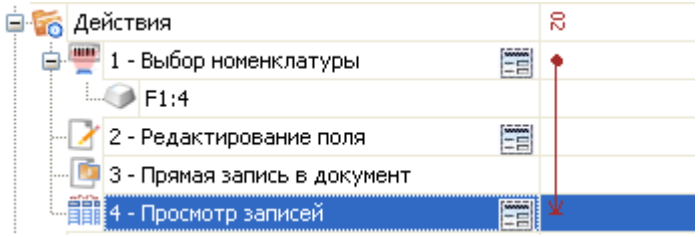
Будет разумно, если список сканированных позиций будет показывать не всегда, а только по нажатию какой-нибудь клавиши терминала сбора данных или по вводу какого-нибудь зарезервированного штрихкода. Для этого в Mobile SMARTS предусмотрены обработчики клавиш и штрихкодов к визуальным действиям.

Добавим такой обработчик к действию выбора номенклатуры, чтобы в окне сканирования штрихкодов товара иметь возможность просмотреть список уже отсканированного:

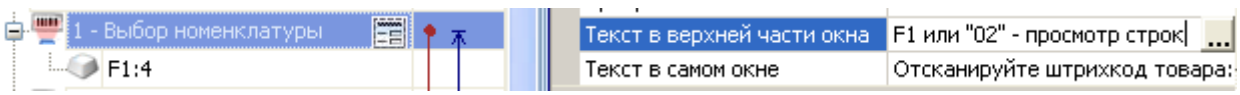


В данном случае настройки свойств обработчика говорят, что в окне выбора номенклатуры (сканирования товара) по нажатию клавиши «F1» или вводе в качестве штрихкода строки «02» (с клавиатуры вручную или действительно сканированием штрихкода «02») программа терминала должна перейти к исполнению действия с именем «4», а именно в действию просмотра строк документа.

После того, как клавиша, штрихкод и переход на действие с именем «4» заданы, при щелчке на «Просмотр записей» в дереве видна стрелка перехода (с подписью «02»):

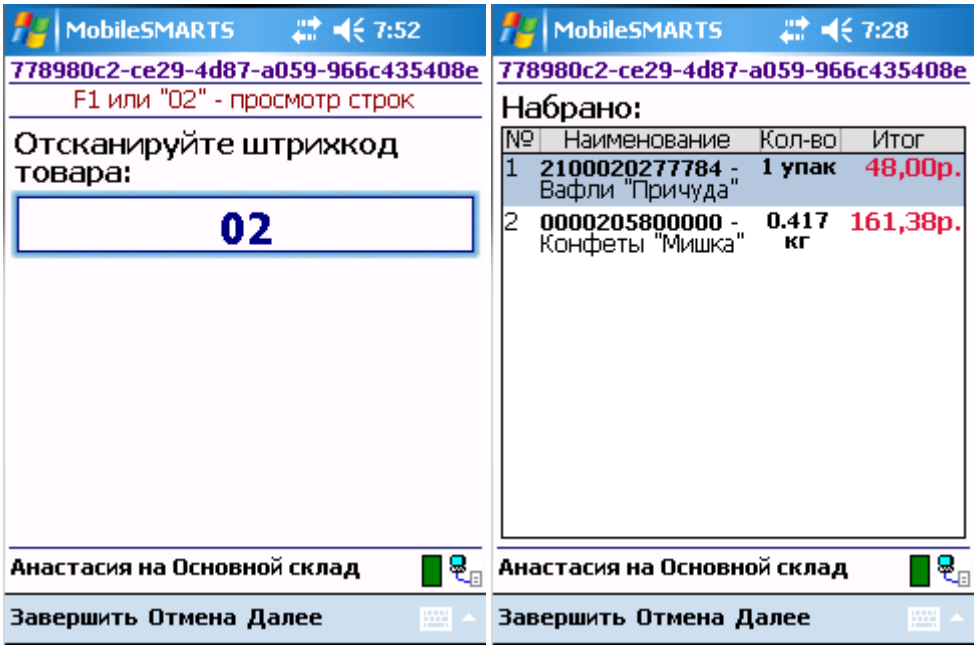


Также будет разумно оставить в окне сканирования товара визуальное напоминание о том, какой штрихкод и какая клавиша были назначены для показа списка. Это делается в свойстве «Текст в верхней части окна»:



Действие «Просмотр записей» оказалось последним в дереве и, соответственно, после него документ будет завершен. По нажатию Escape будет идти возврат к сканированию, т.к. у просмотра записей указано «Глубина отмены действия = На одно действие», однако если случайно нажать Enter, то документ завершится. Чтобы этого не случалось и после просмотра списка происходил бы возврат к сканированию, следует указать следующим действием просмотра записей служебное слово «finishproc»:

После запуска на отладку и сканирования нескольких строк, получится следующее:

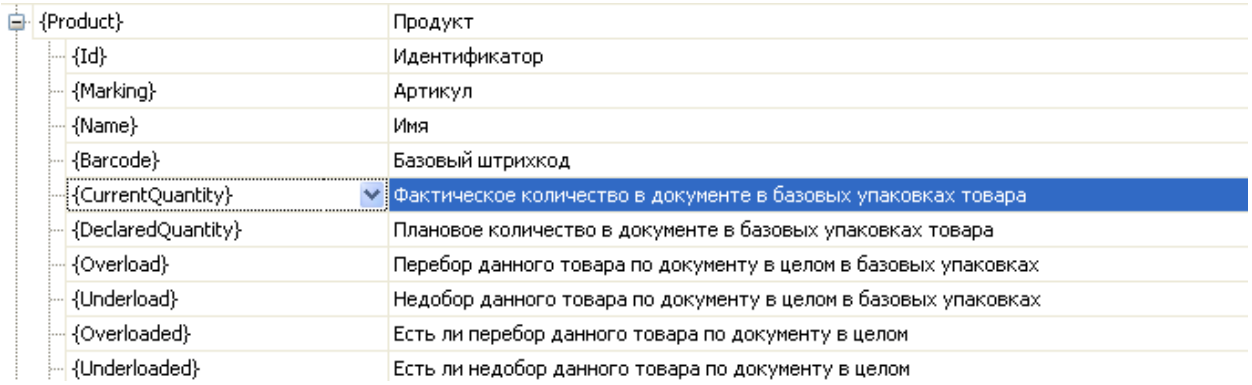


Примечание: Благодаря отладчику Mobile SMARTS ошибки и недоделки в настройках действий можно исправлять прямо во время исполнения программы на терминале сбора данных. Например, изменить текст в заголовке или состав колонок и тут же увидеть, как он поменяется на экране.

Отображение частичных итогов по сканированному товару

Для получения сумм и итогов, которые обычно требуют циклов по строкам, с условиями и т.п., в Mobile SMARTS предусмотрены более короткие явные методы.

Например, в объектах номенклатуры предусмотрены свойства, которые возвращают различные итоги по текущему документу. Все такие свойства доступны в панели управления при редактировании текстов в окнах действий, для чего предусмотрен специальный редактор с подсказкой:



ПРИМЕЧАНИЕ: указанные свойства доступны только на мобильном терминале и отсутствуют в COM-компоненте доступа к серверу Mobile SMARTS.

Пользуясь этим, можно вывести в окне ввода цены общее количество товара, штрихкод которого был только что сканирован:

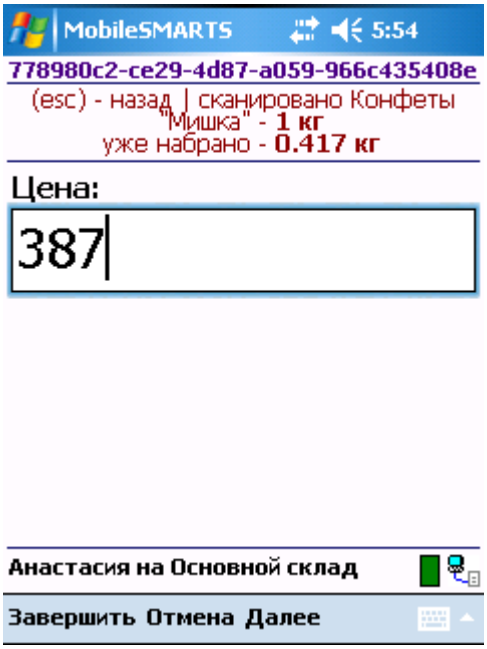
{SelectedProduct.Product.CurrentQuantity}

В данном случае «SelectedProduct» – это переменная сессии, в которую действие сканировании товара («Выбор номенклатуры») положило проект строки для занесения в документ. «Product» – свойство проекта строки, содержащее объект с номенклатурой, а «CurrentQuantity» – соответствующее правильно суммированное количество товара по документу, с учетом пересчета упаковок в базовые.

Укажем в качестве значения свойства «Текс в верхней части окна» действия ввода цены следующие две строки:

(esc) - назад | сканировано {SelectedProduct.Name} - {SelectedProduct.QuantityString}

уже набрано - {SelectedProduct.Product.CurrentQuantity} {SelectedProduct.Product.BasePacking.Name}



Отображение полных итогов по набранному товару

Для получения сумм и итогов, которые обычно требуют циклов по строкам, с условиями и т.п., в Mobile SMARTS предусмотрены более короткие явные методы.

Полная стоимость и количество набранного товара можно посчитать несколькими способами. Рассмотрим по очереди несколько вариантов.

Просто сложить количества и цены

Если сканируются просто коробки или веса, суммирование их количества имеет смысл. А в том случае, если вводимая цена – это стоимость сканированной позиции, а не цена единицы – то вывод итогов сводится просто к выводу суммы по колонкам количества и цены.

Табличные части документа – {Document.CurrentItems} и {Document.DeclaredItems} – позволяют работать с суммами по колонкам без циклов. Допустим, если в документе есть колонка «Количество2»,

то сумма по колонке во всех строках табличной части может быть получена при помощи «{Document.CurrentItems.Количество2}», об этом можно подробнее почитать в соответствующем разделе о документах. Таким образом, вывод сумм сводится к указанию следующих текстовых шаблонов в заголовке, например, действия сканирования товара:

Итого: {Document.CurrentItems.CurrentQuantity} на {Document.CurrentItems.Цена:(0:0.00)р.}.

Просто сложить количества, цены взять из поля номенклатуры или упаковки

Как и в предыдущем варианте, если сканируются только коробки или веса, то простое суммирование их количества имеет смысл. Для того, чтобы суммировать стоимость строк, придется воспользоваться специальными методами суммирования.

Все коллекции и массивы чего-либо в Mobile SMARTS, а в частности табличные части документа – {Document.CurrentItems} и {Document.DeclaredItems}, – позволяют суммировать по свойствам содержащихся в них объектах с применением арифметики при помощи специального метода Sum. Например, указание «{Document.CurrentItems.Sum("{Item.CurrentQuantity}")}» эквивалентно указанию «{Document.CurrentItems.CurrentQuantity}» из предыдущего примера. Как видно, Sum принимает строку с шаблоном, в котором указывается формула расчета слагаемых; при этом текущий объект коллекции или массива (можно сказать, «переменная цикла») упоминаются как «Item».

Таким образом, вывод сумм сводится к указанию следующих текстовых шаблонов в заголовке, например, действия сканирования товара.

Если цена хранится в товаре и требуется пересчет количества в строке документа (в каких-то там единицах) а базовые упаковки (основные единицы измерения товара):

Итого: {Document.CurrentItems.CurrentQuantity} на
{Document.CurrentItems.Sum("{Item.CurrentQuantityInUnits * Item.Product.Цена}"):(0:0.00)р.}.

Если цена хранится в упаковке товара и ничего пересчитывать не надо:

Итого: {Document.CurrentItems.CurrentQuantity} на
{Document.CurrentItems.Sum("{Item.CurrentQuantity * Item.Packing.Цена}"):(0:0.00)р.}.

§ 8. Пример реальной складской операции

Рассмотрим простейший пример схемы метаданных для процесса *поступления товара на склад*, целиком исполняемого одним сотрудником.

Итак, необходимо выполнить при помощи терминала сбора данных (ТСД) следующие операции:

1. Выгрузка товара из транспортного средства;
2. Проверка по накладной;
3. Расфасовка по паллетам;
4. Ввод данных о каждой паллете;
5. Размещение в места постоянного хранения.

Необходимо создать один единственный тип документа Mobile SMARTS и настроить схему его обработки. Схема обработки будет определять порядок действий сотрудника при выполнении документов типа «Поступление».

Составление плана работы

Продемонстрируем составление плана работы на простом примере для операции поступления товара. Требуется составить небольшой план того, как сотрудник будет пользоваться ТСД, обрабатывая поступление.

Рассмотрим такой план по шагам. Допустим, выгрузка товара начинается с ввода номера пломбы на воротах транспортного средства. Итак, первое, что должен увидеть человек, – окно ввода номера пломбы (1). Теперь необходимо сверить товар по накладной с реальностью и с данными заявки и затем расфасовать приход по паллетам. Будем рассматривать вариант, когда товар пришел навалом, и вся передняя стенка заставлена коробками. Необходимо взять свободный поддон и набрать на него коробок одноименного товара. Допустим, коробки обматываются пленкой. Теперь на сцену выходит штриховое кодирование, а именно *рулон* заранее распечатанных *этикеток паллет*.

> Клеить заранее распечатанные безымянные этикетки намного быстрее, чем печатать более информативные этикетки по ходу приемки. Сотрудник берет готовый рулон и налепляет по одной этикетке на каждый паллет.

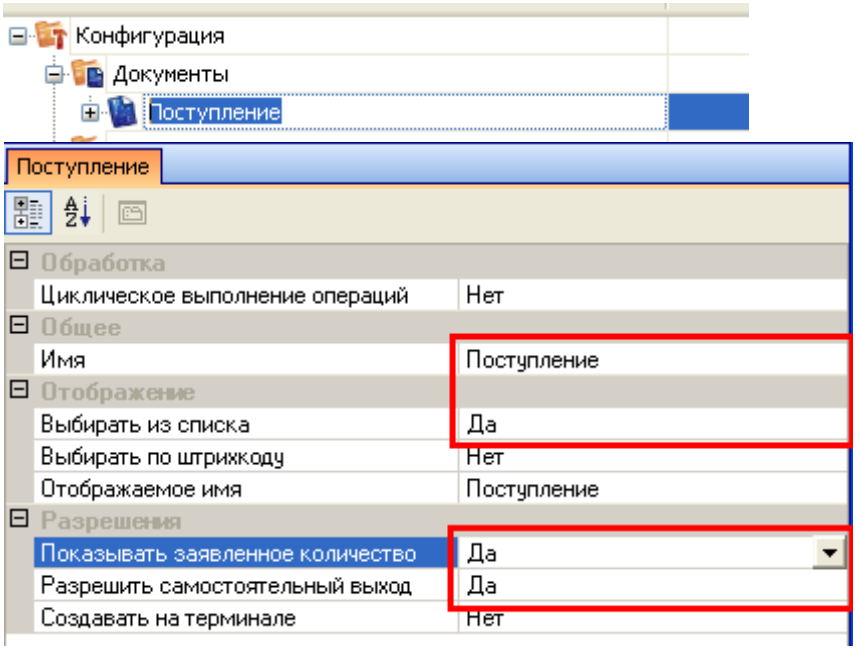
Наступило время ввода данных. Необходимо сканировать этикетку и ввести данные о коробках. Итак, третье – сканирование штрихкода паллеты. Четвертое – ввод номенклатуры товара (сканирование штрихкода коробки или выбор товара из списка). Пятое – ввод количества. Данные введены и нужно переходить к следующей паллете, т.е. на шаг 3. А если не нужно больше паллет? Для этого шестое – спросить, пора ли уже заканчивать приемку.

Реализация плана работы в виде схемы обработки документа

Теперь, когда у нас есть план работы сотрудника с мобильным терминалом, его необходимо переложить на язык схемы обработки, понятный Mobile SMARTS. Рассмотрим пример создания такой схемы «с нуля». По шагам:

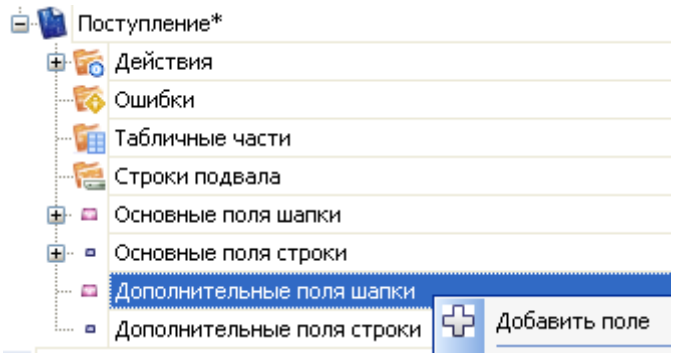
1. Создание пустой конфигурации(см. «Hello World» для Mobile SMARTS);

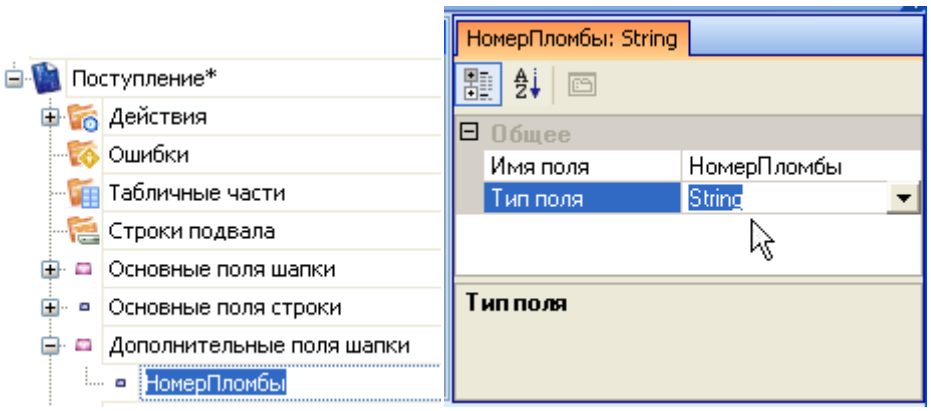
2. Добавляем в узел «Документы» новый тип документа и переименовываем его в «Поступление» (сразу, по F2 или через свойство «Отображаемое имя»):



Свойства, значения которых не равны значениям по умолчанию (т.е. требующие изменения), выделены красной рамкой.

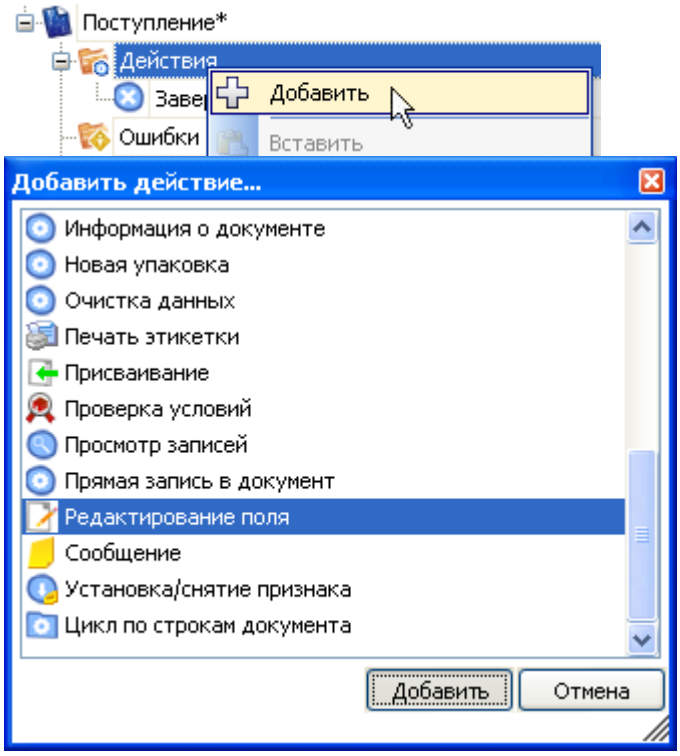
3. Добавить новое поле «Номер пломбы» в шапку документа

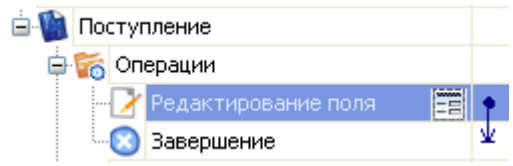


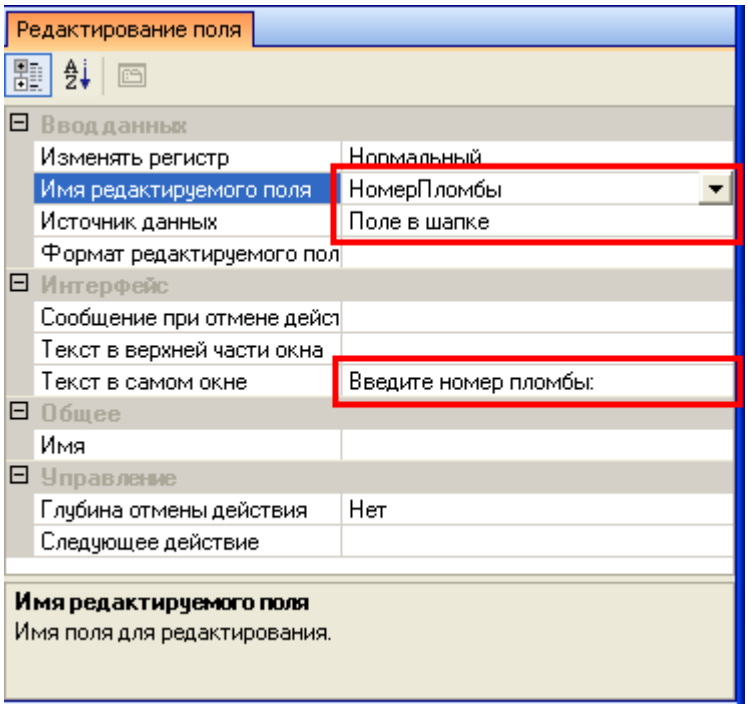


4. Последовательно наполняем узел «Действия» следующим набором:

1) Редактирование поля

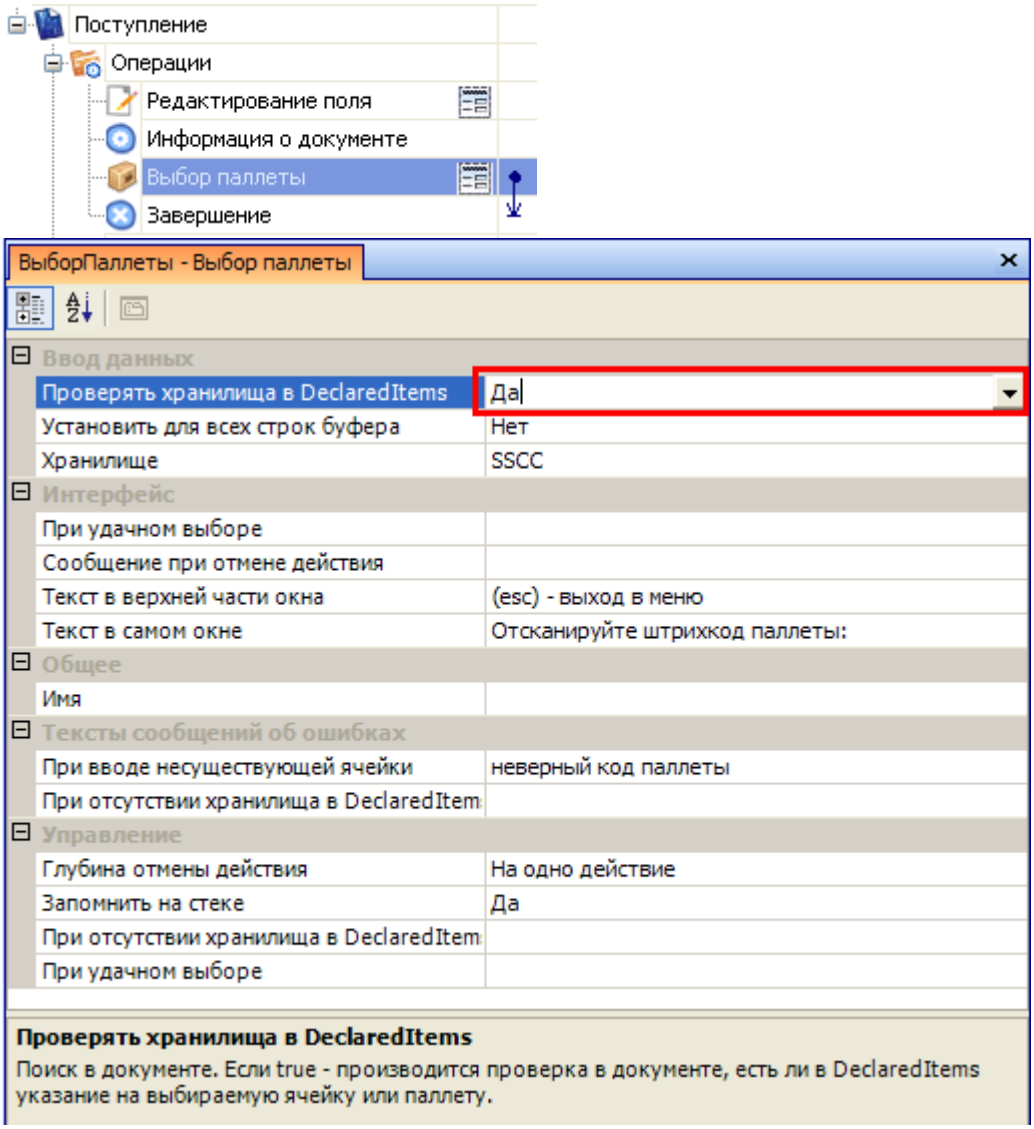






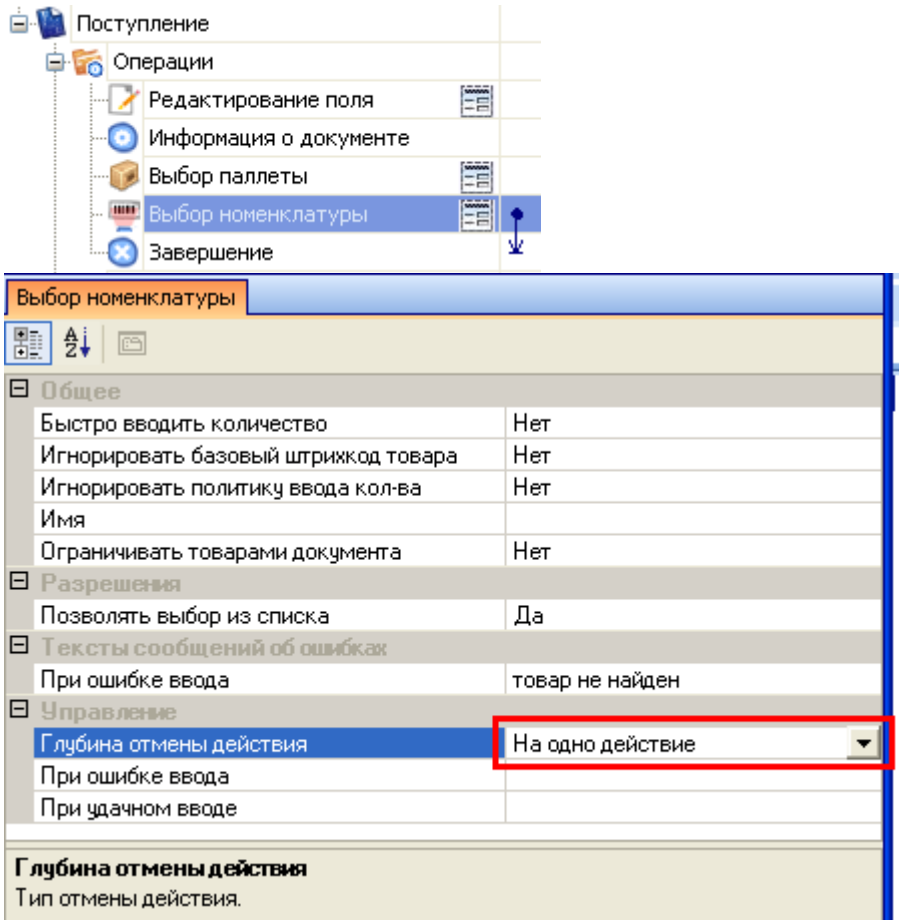
2) Выбор паллеты:

(вызов контекстного меню и выбор из списка «Добавить действие...» опущены)

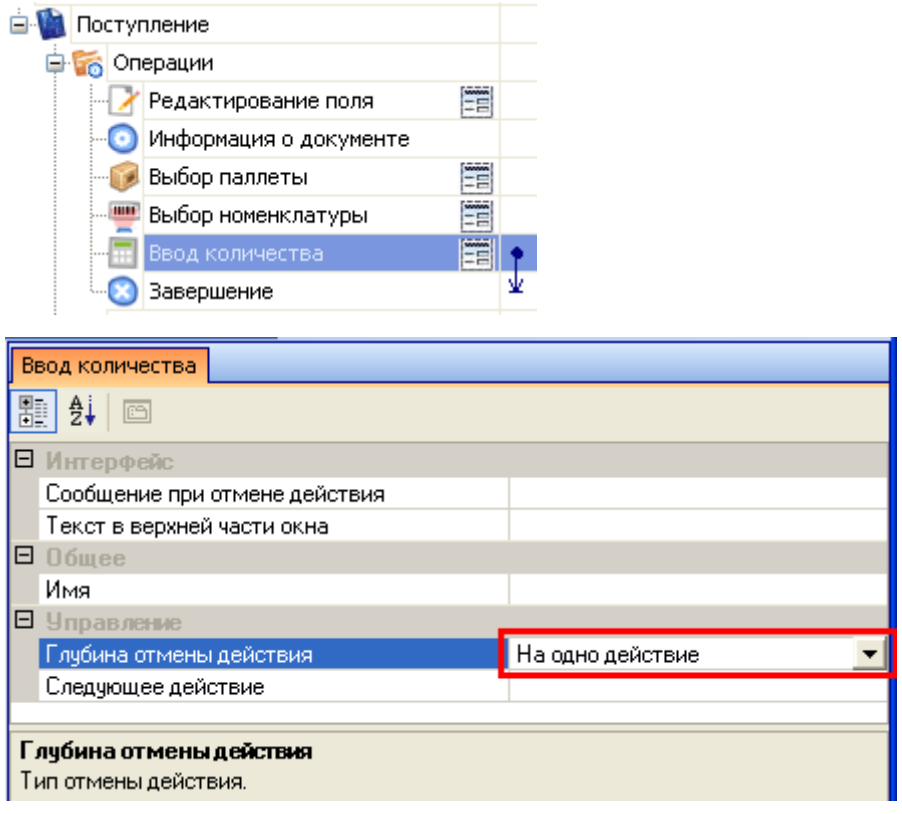


Один-единственный флажок «Да» позволяет включить проверку соответствия прихода накладной.

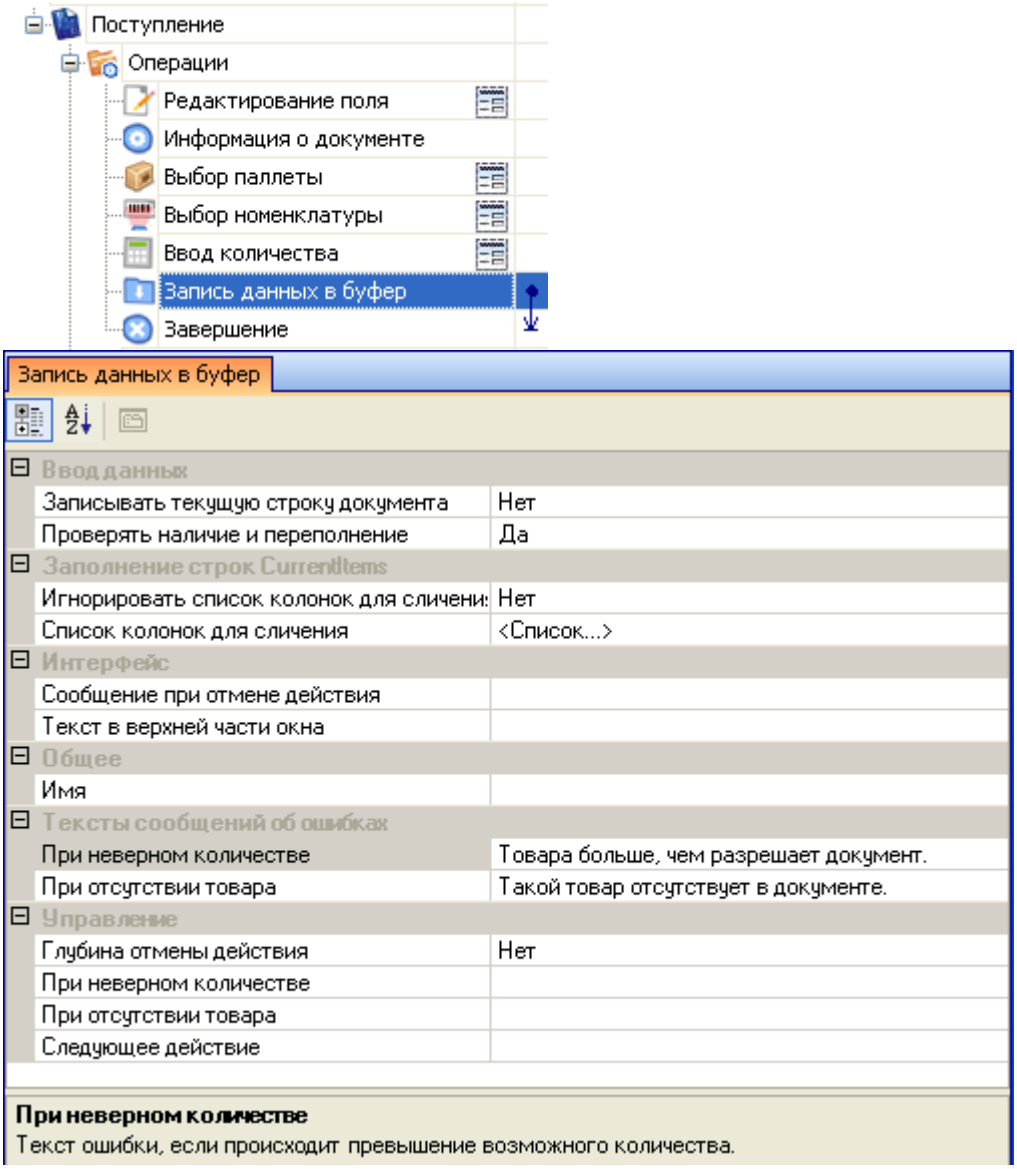
3) Выбор номенклатуры



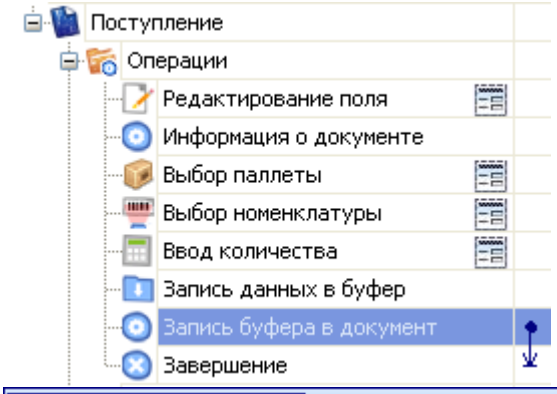
4) Ввод количества

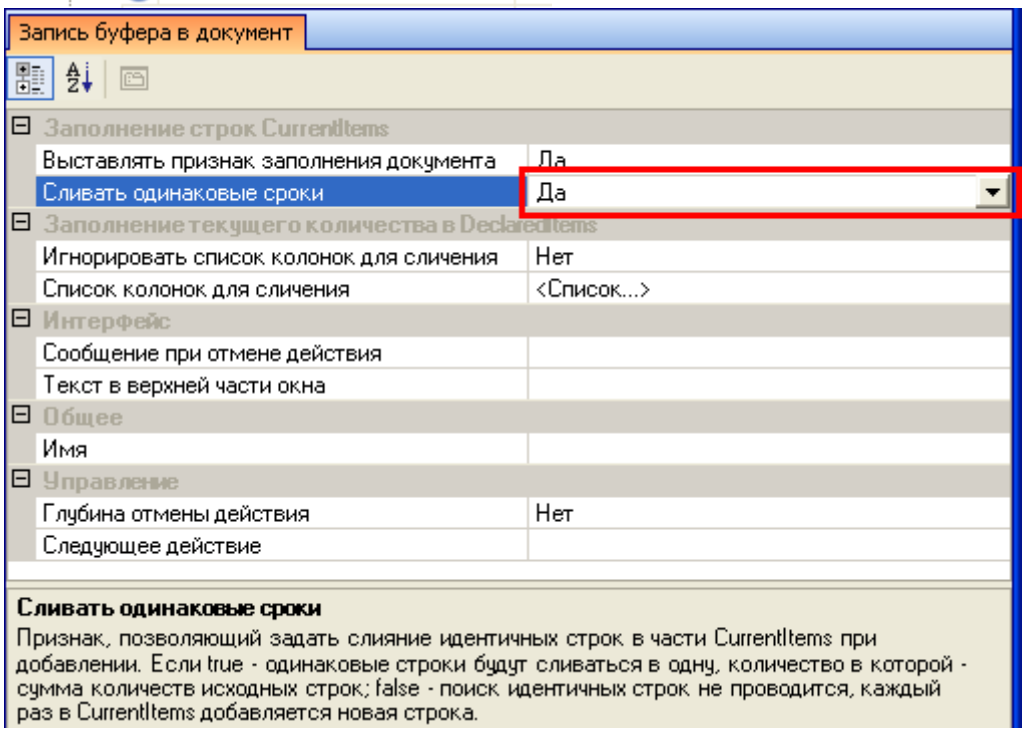


5) Запись данных в буфер

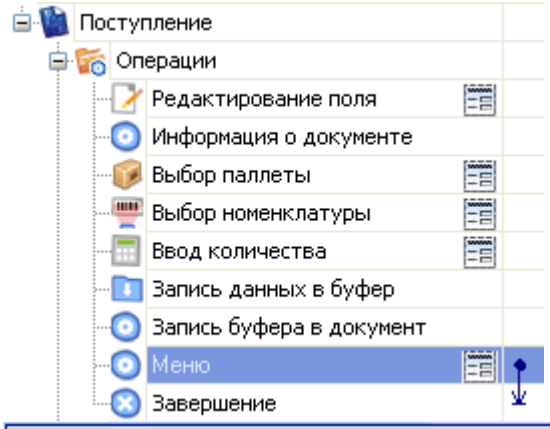


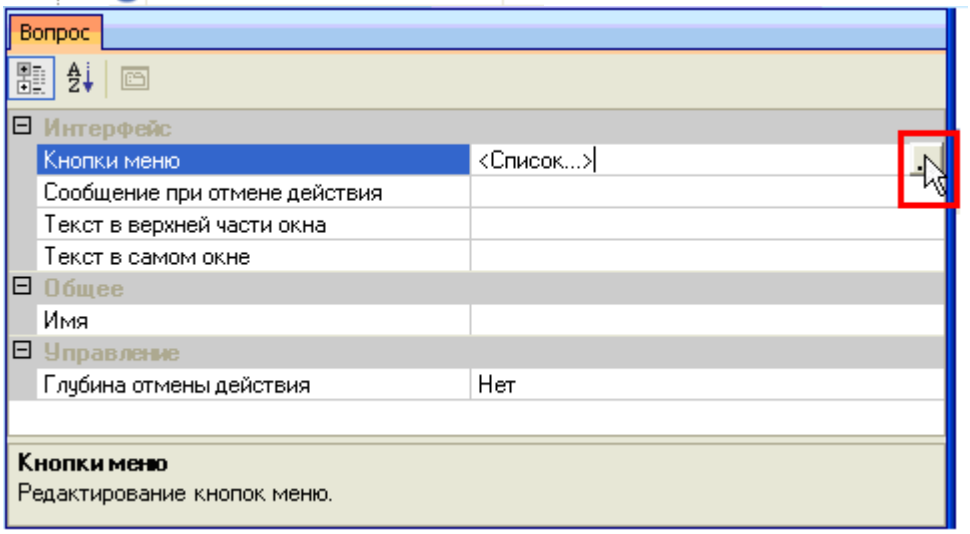
6) Запись данных в документ

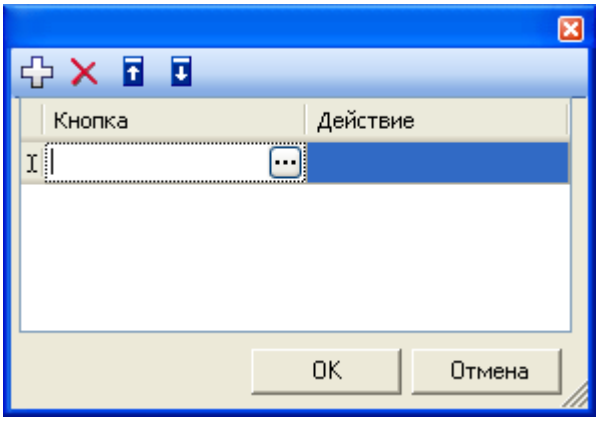


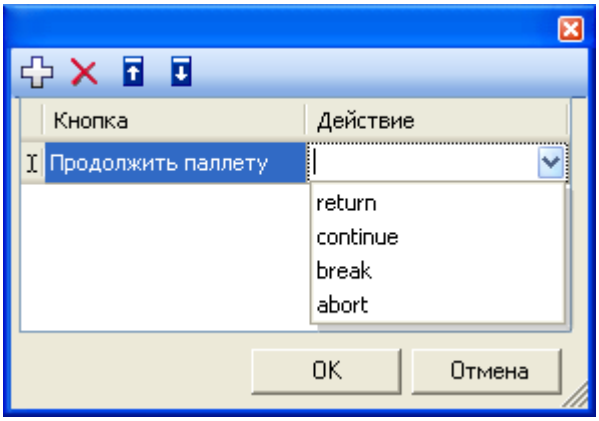


7) Вопрос



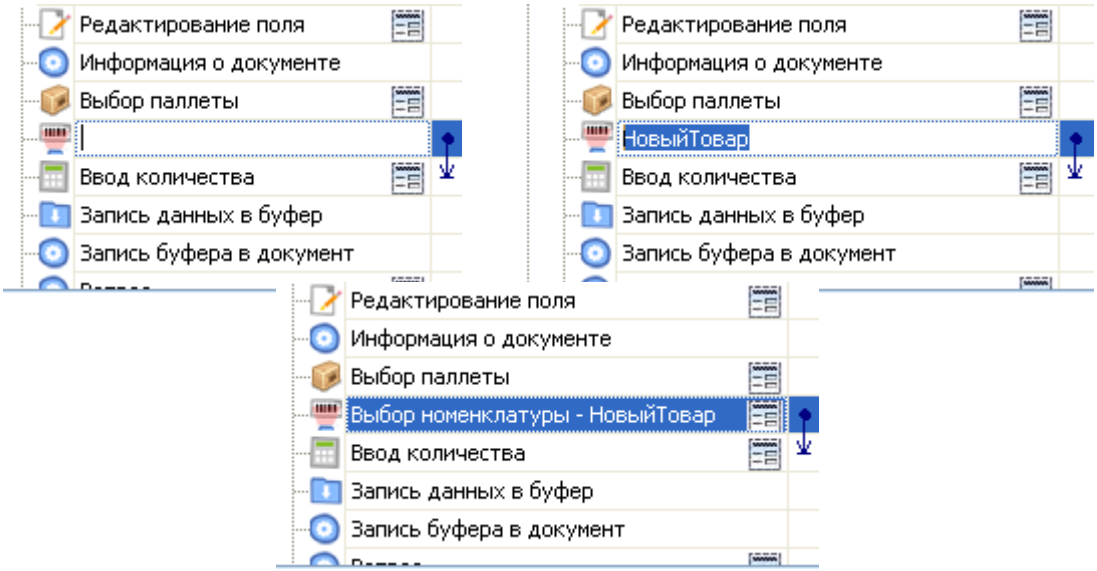




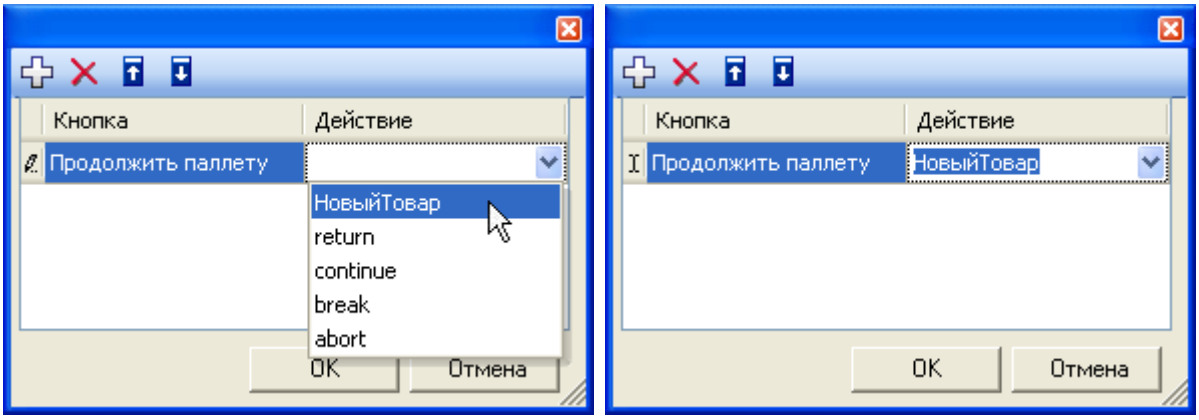


В выпадающем списке напротив «Продолжить паллету» отображаются возможные варианты перехода по нажатию такой кнопки. Относительно назначения служебных слов «return» и т.п. см. параграф о примере с меню

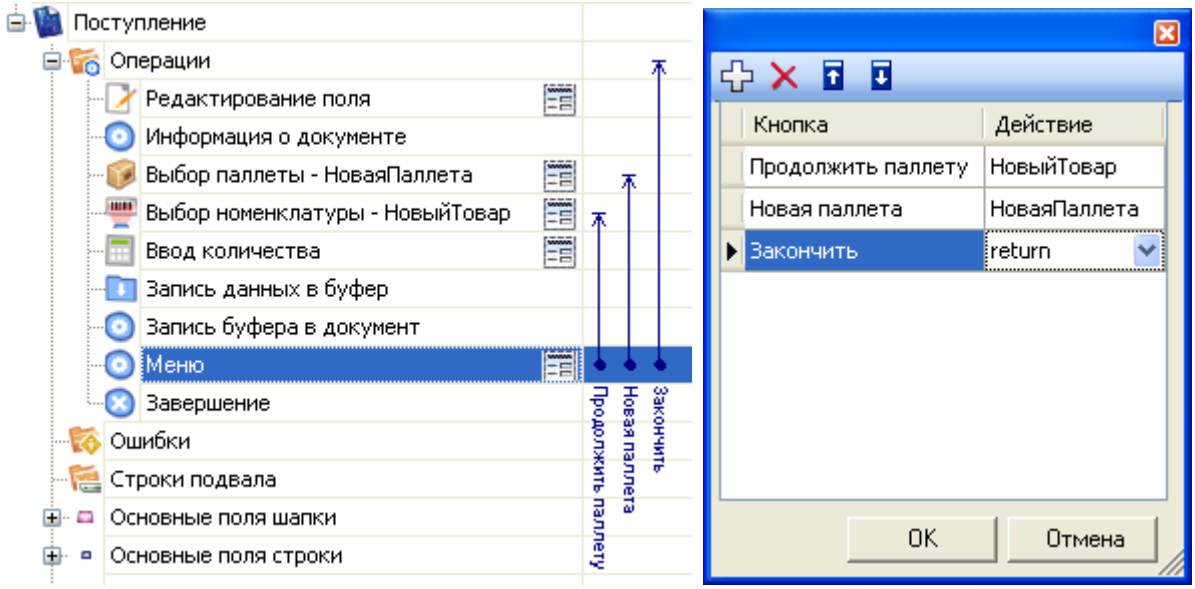
В списке выбора нет перехода на ввод нового товара. Чтобы он там появился, дать задать действию выбора новой номенклатуры конкретное имя:



Теперь необходимо снова вернуться в диалог настройки списка кнопок меню и выбрать переход на «НовыйТовар»:



Таким же образом следует поступить с двумя оставшимися пунктами меню:



Если пользователи и склады уже были выгружены в вашу конфигурацию, можно проводить испытание. Если пользователей и складов еще нет, заводим их самостоятельно (см. «Заведение новых складов» и «Склады необходимо заводить ДО заведения пользователей.

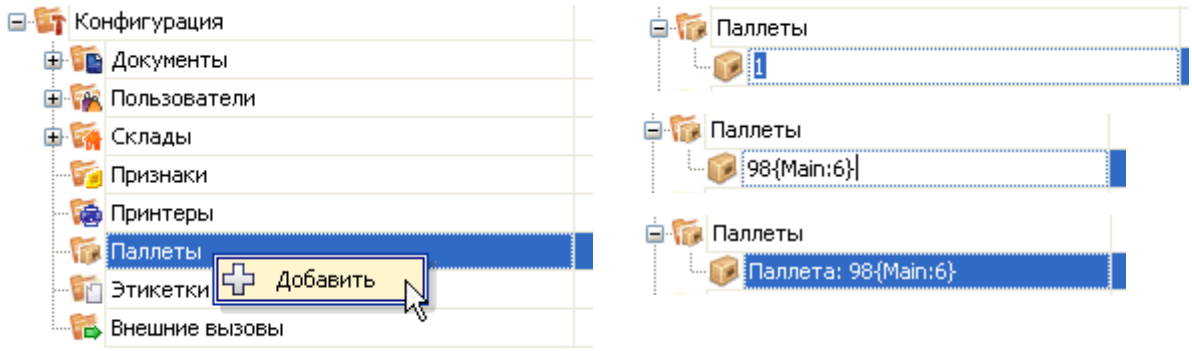
Заведение новых пользователей и групп пользователей»).

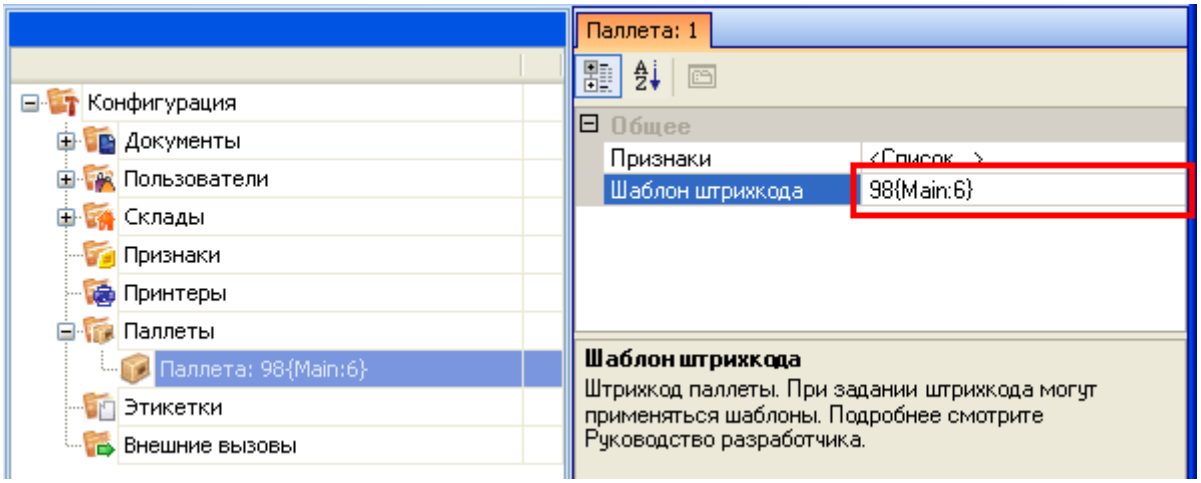
Настройка штрихкодов паллет

Для проведения тестирования, в конфигурации обязательно должны быть заведены склады, пользователи и паллеты! Также должен быть выгружен справочник товаров (или взят из тестового примера). Для теста также потребуется документ накладной на поступление с заполненными ожидаемыми строка поступления (или воспользуйтесь накладной из тестового примера).

На шаге 3) нашего простейшего примера происходит сканирование штрихкода паллеты. Каким образом терминал определяет, что это именно паллета, а не что-то другое? За идентификацию паллет ответственен узел «Паллеты» редактора метаданных. В нем регистрируются общие шаблоны, под которые должны подходить использующиеся этикетки паллет. Наиболее подробно всё это рассмотрено в разделе «Шаблоны штрихкода».

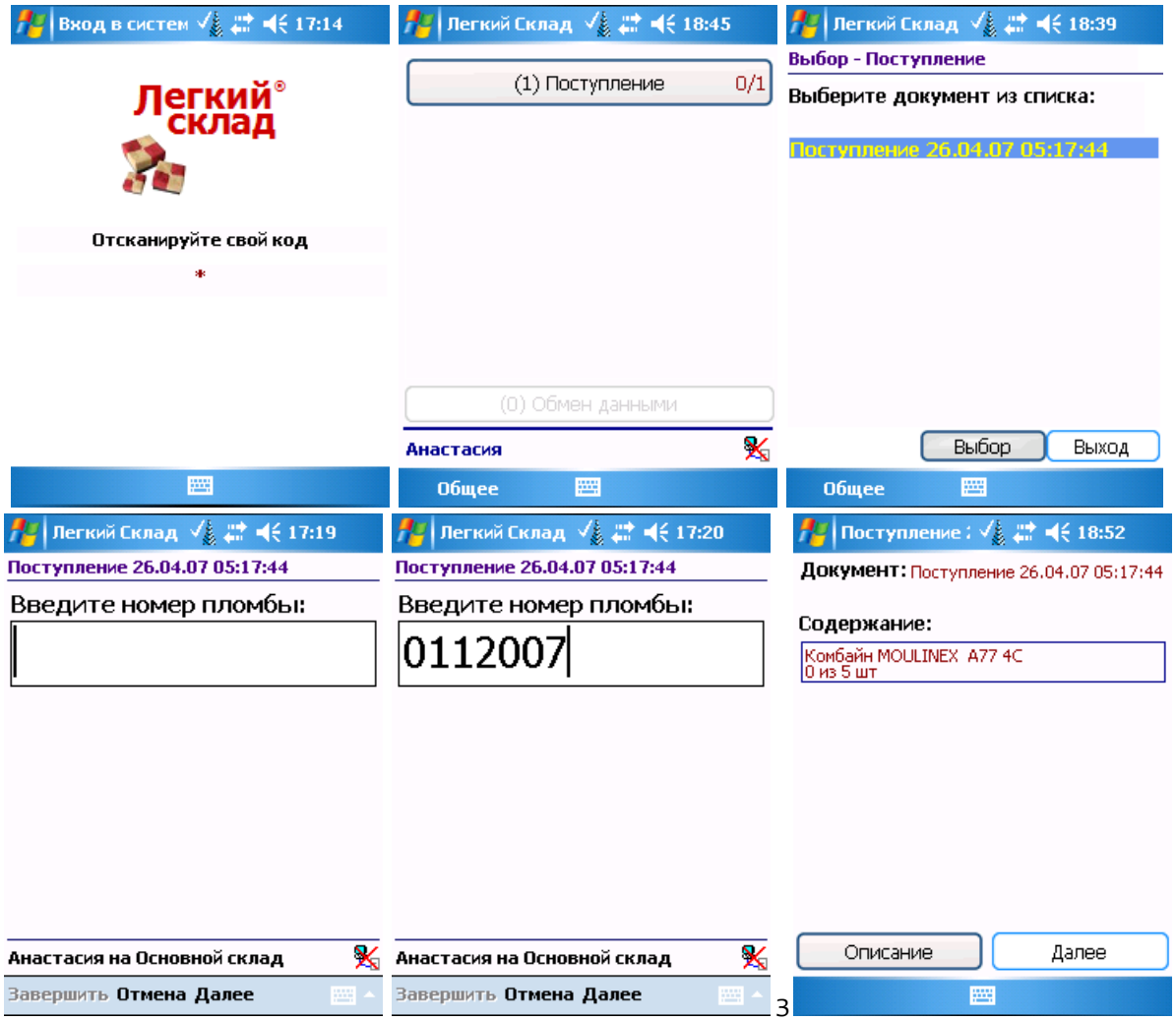
Для простейшего примера достаточно зарегистрировать один-единственный шаблон паллеты примерно такого вида: «паллетой считается любой штрихкод, который начинается с цифр “98” и дальше любые 6 символов». В редакторе это задается следующим образом:

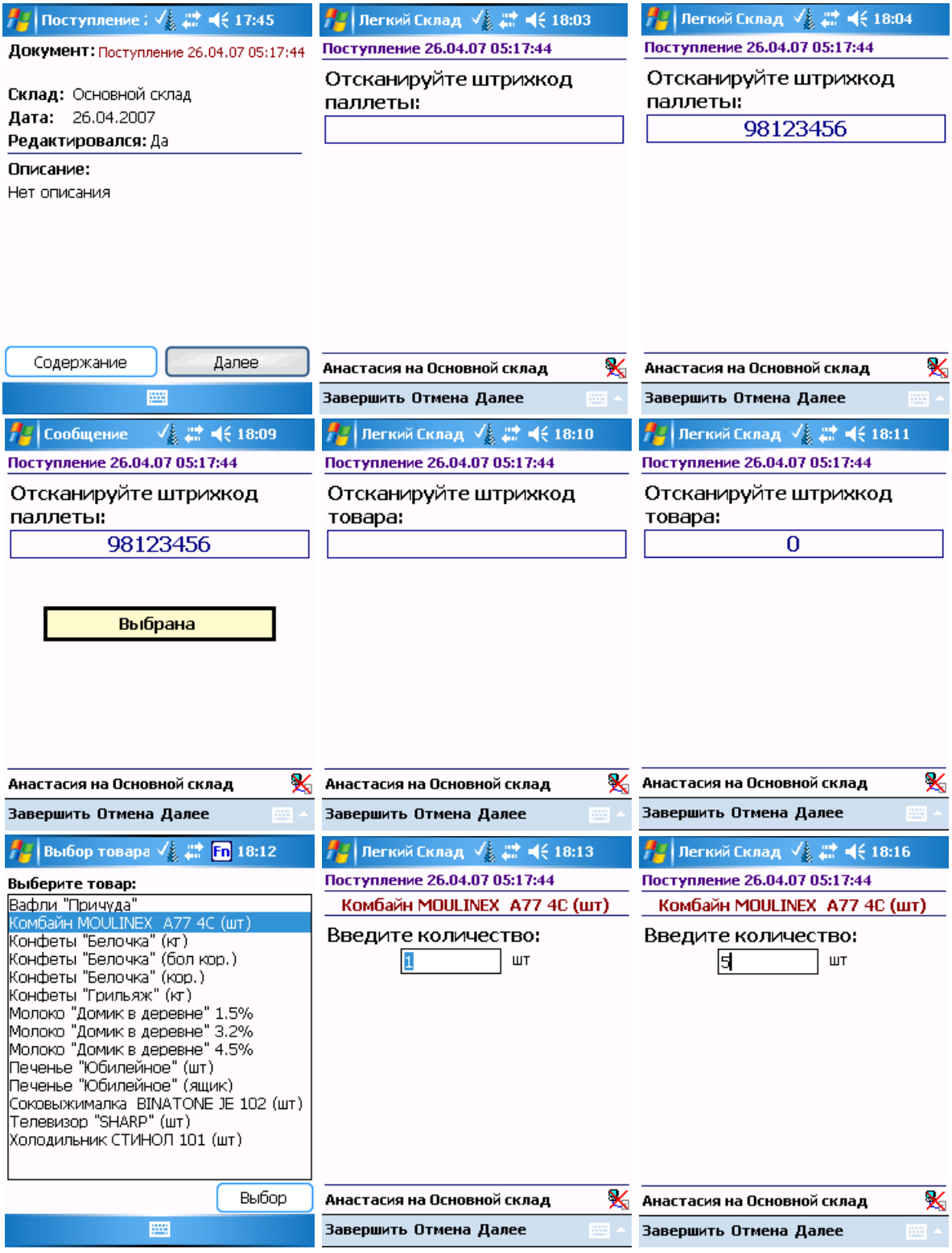


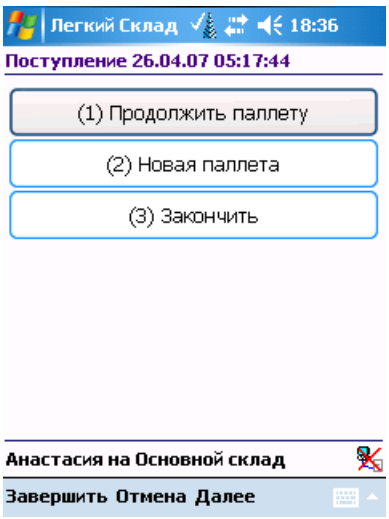


Результат

В результате мы должны получить на терминале следующую операцию:



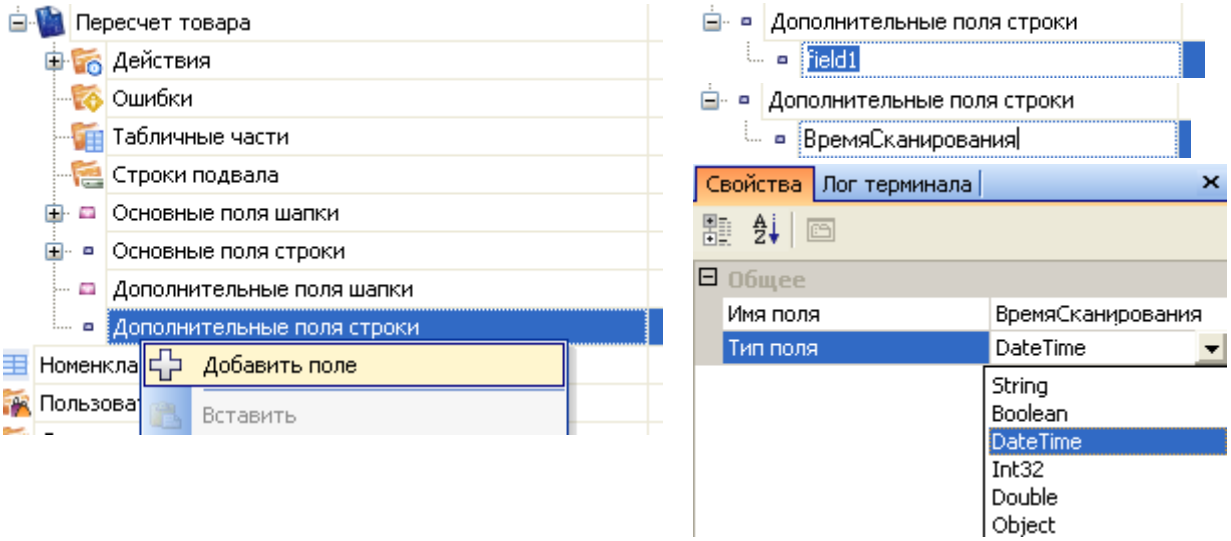

Поступление : 17:45
Документ: Поступление 26.04.07 05:17:44
Склад: Основной склад
Дата: 26.04.2007
Редактировался: Да
Описание:
Нет описания
Содержание
Далее
Легкий Склад 18:03
Поступление 26.04.07 05:17:44
Отсканируйте штрихкод паллеты:
Анастасия на Основной склад
Завершить Отмена Далее
Легкий Склад 18:04
Поступление 26.04.07 05:17:44
Отсканируйте штрихкод паллеты:
98123456
Анастасия на Основной склад
Завершить Отмена Далее
Сообщение 18:09
Поступление 26.04.07 05:17:44
Отсканируйте штрихкод паллеты:
98123456
Выбрана
Анастасия на Основной склад
Завершить Отмена Далее
Легкий Склад 18:10
Поступление 26.04.07 05:17:44
Отсканируйте штрихкод товара:
Анастасия на Основной склад
Завершить Отмена Далее
Легкий Склад 18:11
Поступление 26.04.07 05:17:44
Отсканируйте штрихкод товара:
0
Анастасия на Основной склад
Завершить Отмена Далее
Выбор товара Fn 18:12
Выберите товар:
Вафли "Причуда"
Комбайн MOULINEX A77 4C (шт)
Конфеты "Белочка" (кг)
Конфеты "Белочка" (бол кор.)
Конфеты "Белочка" (кор.)
Конфеты "Грильяж" (кг)
Молоко "Домик в деревне" 1.5%
Молоко "Домик в деревне" 3.2%
Молоко "Домик в деревне" 4.5%
Печенье "Юбилейное" (шт)
Печенье "Юбилейное" (ящик)
Соковыжималка BINATONE JE 102 (шт)
Телевизор "SHARP" (шт)
Холодильник СТИНОЛ 101 (шт)
Выбор
Анастасия на Основной склад
Завершить Отмена Далее
Легкий Склад 18:13
Поступление 26.04.07 05:17:44
Комбайн MOULINEX A77 4C (шт)
Введите количество:
1
шт
Анастасия на Основной склад
Завершить Отмена Далее
Легкий Склад 18:16
Поступление 26.04.07 05:17:44
Комбайн MOULINEX A77 4C (шт)
Введите количество:
5
шт
Анастасия на Основной склад
Завершить Отмена Далее



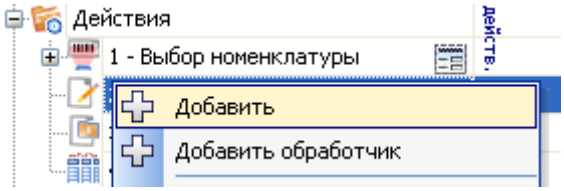
§ 9. Контроль времени выполнения операций на ТСД

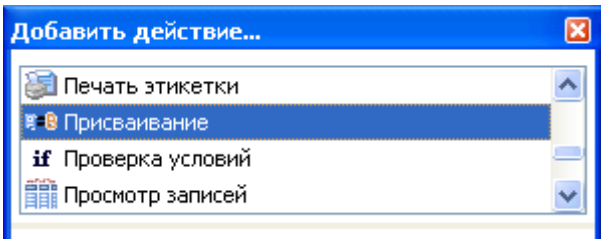
Во многих случая оказывается полезным знать время выполнения кладовщиком тех или иных операций на терминале сбора данных.

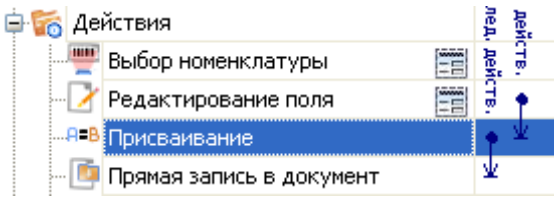
В .NET нет отдельных типов данных для времени или для даты, а есть один общий для даты и времени (DateTime). Для хранения времени следует добавить в документ соответствующие колонки. Документ в Mobile SMARTS по умолчанию содержит две табличные части одинакового формата. Каждая табличная часть состоит из строк, соответственно колонки называются *полями строки документа*. Любая дата или время сканированной позиции будет являться дополнительным полем строки и иметь тип DateTime (т.е. дата и время):

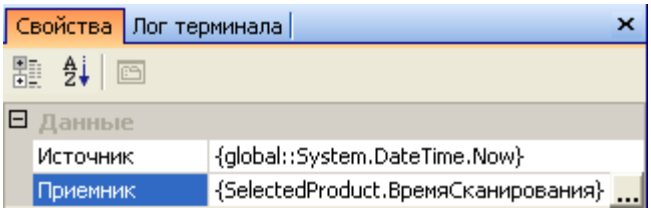


Получить текущее время на терминале сбора данных можно при помощи шаблона «{global::System.DateTime.Now}». Для того, чтобы время попало в документ, нужно скопировать его в «{SelectedProduct.ВремяСканирования}». Действие «Выбор номенклатуры», обязательное к использованию, заполняет переменную сессии «SelectedProduct», которая (переменная) необходима для занесения строки в документ. В этой переменной содержится своеобразный «проект» новой строки. Именно туда и следует заносить время, чтобы затем при выполнении занесения новой строки в документ, время оказалось в этой новой строке.









При этом следует всегда помнить, что в документе окажется не время занесения строки в документ, а время выполнения вот этой вот операции присваивания.

§ 10. Настройка свойств действий в Mobile SMARTS – Часть 2

Действие «Выбор номенклатуры»

Действие с многозначительным названием «Выбор номенклатуры» выполняет очень важную функцию: указать программе тот самый товар, для которого будут вводиться количества и пр. Одно и то же действие отвечает за любой способ выбора товара. Способы могут быть следующими:

1. Сканирование ШК товара;
2. Ввод ШК товара вручную;
3. Ввод кода или артикула товара;
4. Выбор товара из списка;
5. Поиск товара по базе на основании ШК, кода или артикула, указанного в переменной сессии.

Действие выбора номенклатуры поддерживает наиболее вероятные сценарии выбора товара. В нем есть настройки отображения списка для выбора товара из списка. Есть настройки для реакции на неверный ввод. Полный список свойств данного действия следующий:

	Автоматический выбор единственного товара из списка	Да/Нет. Определяет, следует ли автоматически выбирать товар из списка товаров (при вводе в качестве штрихкода '0'), если в списке всего один товар. Да - выбрать и перейти к следующему действию. Нет - вывести список с одним товаром для подтверждения выбора.
	Автоматический выбор первого товара из списка	Да/Нет. Определяет, следует ли автоматически выбирать первый же товар из списка товаров (при вводе в качестве штрихкода '0'). Да - выбрать и перейти к следующему действию. Нет - вывести список товаров для ручного выбора.
	Смена регистра штрихкода	Без изменения | К верхнему регистру | К нижнему регистру. Задает тип изменения регистра символов введенного штрихкода.

	Фильтр товаров для поиска по присутствию признака	Список идентификаторов. Позволяет указать фильтрацию товаров для поиска по присутствию признака. Поиск будет производиться только среди тех товаров, у которых стоят все указанные признаки. Можно указывать идентификатор признака, идентификатор типа признака или путь к переменной, содержащей признак, тип признака или их идентификатор.
	Фильтр товаров для поиска по отсутствию признака	Список идентификаторов. Позволяет указать фильтрацию товаров для поиска по отсутствию признака. Поиск будет производиться только среди тех товаров, у которых нет ни одного из указанных признаков. Можно указывать идентификатор признака, идентификатор типа признака или путь к переменной, содержащей признак, тип признака или их идентификатор.
	Заголовок окна выбора товара из списка	Строка. Текст, выводимый в заголовке окна со списком товаров.
	Игнорировать политику ввода кол-ва	Да/Нет. Игнорировать политику учета товара, даже если она у товара выставлена.
	Быстро вводить количество	Да/Нет. Быстрый выбор продукта. Товар выбирается в базовом типе упаковки с количеством = 1.
	Типы признаков для построения дерева	Список идентификаторов. Позволяет задать вывод справочника товаров в виде дерева. Указывается список из типов признаков, которые будут использоваться для построения уровней вложенности дерева.
	Переменная хранилища	Строка. Имя переменной, которая будет использована для хранения сканированной ячейки или паллеты (см. "Выбирать ячейку сканированием" и "Выбирать паллету сканированием").
	Сразу показывать список товаров	Да/Нет. Включает режим выбора продукта без штрихкода, из списка. Для вывода онлайн справочника ДОЛЖНО быть выставлено древовидное отображение товаров.
	Выбирать из ранее найденных	Да/Нет. Включает режим для случая, когда, при нахождении по штрихкоду нескольких товаров, производится поиск в ранее выбранных и его автоматический выбор.
	Переменная штрихкода товара	Строка. Имя переменной сессии, используемой для хранения введенного штрихкода товара. По-умолчанию 'ScannedBarcode', если задать пустое значение (null), штрихкод в сессии сохранен не будет.
	Обработать штрихкод из сессии	Да/Нет. Определяет, использовать ли переменную для хранения штрихкода не для того, чтобы занести туда результат, а для того, чтобы взять оттуда штрихкод на распознавание. Эта возможность позволяет выбрать известный товар в автоматическом режиме без запроса к пользователю.
	При удачном вводе	Действие, на который произойдет переход в случае удачного выбора товара.

	Источник списка товаров	Строка. Имя переменной, в которой содержится список товаров для выбора и поиска, либо пусто для поиска в общем списке.
	Игнорировать базовый штрихкод товара	Да/Нет. Игнорировать базовые штрихкоды товаров. Если флаг выставлен, поиск будет учитывать только штрихкоды упаковок, но не штрихкоды товаров.
	При ошибке ввода	Строка. Текст ошибки на случай, если товар не найден.
	Позволять выбор из списка	Да/Нет. Признак того, разрешено ли отображение списка товаров для выбора с помощью введения "0" в качестве штрихкода.
	Ограничивать товарами документа	Да/Нет. Признак, того, что справочник продуктов при поиске и отображении ограничивается позициями, присутствующими в документе.
	При ошибке ввода	Действие, на которое произойдет переход в случае, если товар не найден.
	Выбирать паллету сканированием	Да/Нет. Может ли по введенному штрихкоду, помимо поиска товара, производиться и поиск паллеты.
	Текст выбора паллеты	Строка. Текст сообщения, отображаемый в том случае, если сканированный штрихкод оказался штрихкодом паллеты.
	Перейти по выбору паллеты	Действие, на который произойдет переход в том случае, если сканированный штрихкод оказался штрихкодом паллеты.
	Выбирать ячейку сканированием	Да/Нет. Может ли по введенному штрихкоду, помимо поиска товара, производиться и поиск ячейки.
	Текст выбора ячейки	Строка. Тест сообщения, отображаемого при удачном распознавании штрихкода как штрихкода ячейки (см. Выбирать ячейку сканированием).
	Перейти по выбору ячейки	Строка. Действие, на который произойдет переход в том случае, если сканированный штрихкод оказался штрихкодом ячейки.
	Формат позиций окна выбора из списка	Строка. Текстовый шаблон для вывода строк в просмотре справочника товаров (списке выбора товара при нажатии «0»).

Рассмотрим несколько сценариев использования действия выбора номенклатуры:

Сканирование штрихкода товара и возможность ввода с клавиатуры

По умолчанию действие выбора номенклатуры показывает на экране мобильного терминала большое поле ввода для сканирования штрихкода, текст в заголовке окна и текст прямо над полем ввода. Это позволяет сканировать и распознавать штрихкоды, вводить тот же самый штрихкод с клавиатуры, а также вводить артикул или код товара, по которому он будет идентифицирован.

Выбор товара по ручному вводу артикула

По умолчанию действие выбора номенклатуры распознает товары не только по штрихкоду, но также и по артикулу и уникальному коду товара (например, коду товара в 1С). Для того, чтобы выбрать товар путем

ручного ввода артикула на терминале сбора данных ничего дополнительно программировать или настраивать не нужно.

Выбор товара из списка

По умолчанию свойство «Позволять выбор из списка» выставлено в «Да», поэтому если ввести вместо штрихкода «0» и Enter, то на экран ТСД будет выведен список всех товаров по базе.

В том случае, если где-то требуется вывести только список, следует выставить в «Да» свойство «Сразу показывать список товаров». Как только исполнение операции достигнет такого действия, на экране сразу же будет выведен список товаров для выбора.

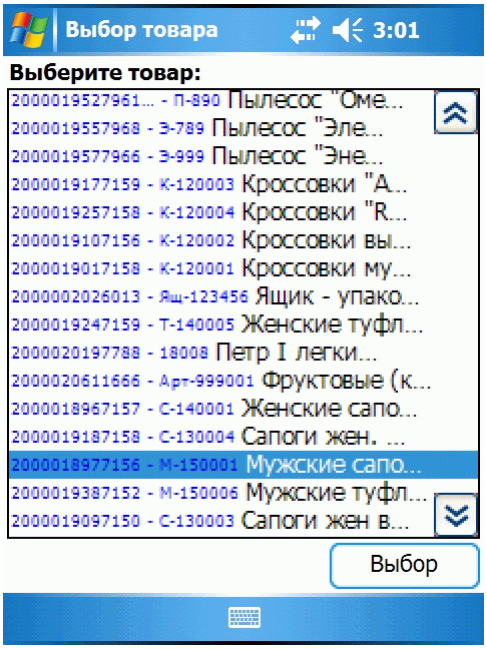
Состав выводимого списка управляется свойствами из группы «Данные» и свойством «Ограничивать товарами документа» из группы «Ограничения».

Два самых распространенных вариантов списка – это полный список всех товаров и список только тех товаров, что присутствуют в документе. Обоих можно добиться одним только свойством «Ограничивать товарами документа». Если оно выставлено в «Нет», то список окажется полным, в противном случае в списке будут только товары по накладной.

Для настройки отображения списка существует свойство «Формат позиций окна выбора из списка». Свойство принимает текстовый шаблон отображения позиции номенклатуры. Каждая позиция в списке – это пара «товар + упаковка», поэтому для обращения к номенклатуре и её упаковке в шаблоне используются пути «Item.Product» и «Item.Packing». Например, шаблон

«<Blue>{Item.Packing.Barcode} – {Item.Packing.Marking}</Blue> {Item.Product.Name:(0:E12)}»

выведет на экран следующее:



Подробнее о шаблонах см. «§ 11 Шаблоны текстов и математических выражений».

Распознавание товара по штрихкоду, артикулу или коду

Иногда необходимо найти в справочнике товар по заранее известному коду или артикулу, который указан где-то в документе, в переменной или просто равен «12345», допустим.

Для решения этой задачи в действии выбора номенклатуры предусмотрены свойства «Переменная штрихкода товара» и «Обработать штрихкод из сессии». «Переменная штрихкода товара» используется для хранения штрихкода. При сканировании товара туда попадает сканированный штрихкод. При выборе товара из списка туда чаще всего попадает «0» (т.к. для вызова списка в окне штрихкода вводится «0»). Это происходит по факту выбора, и если ничего выбрано или сканировано не было, то переменная не трогается.

Однако, если выставить в «Да» свойство «Обработать штрихкод из сессии», то действие выбора номенклатуры заглянет в переменную еще до начала выбора. Вместо ожидания ввода пользователя будет произведен автоматический поиск номенклатуры на совпадение штрихкода, кода или артикула со строкой, хранящейся в указанной переменной.

Например, можно поместить в переменную «МойТовар» значение «001» и затем выставить «Переменная штрихкода товара» = «{МойТовар}», «Обработать штрихкод из сессии» = «Да». В этом случае действие выбора номенклатуры будет искать в справочнике товар, соответствующий строке «001». Как и при обычном сканировании товара, результат поиска может быть таким:

1. Товар не был найден – издается громкий звук ошибки, на экран выводится текст из свойства «При ошибке ввода»;
2. Был найден единственный товар – он кладется в сессию в переменную «SelectedProduct», а исполнение переходит либо к действию, указанному в свойстве «При удачном выборе», либо просто к следующему действию в дереве (если свойство пустое);
3. Найдено несколько товаров – в зависимости от значения свойства «Автоматический выбор первого товара из списка» на экран либо выводится список найденных, либо выбирается первый же найденный товар и всё происходит как в пункте 2.

Следовательно, для полностью автоматического выбора заранее известного товара по его штрихкоду, коду или артикулу, необходимо выставить «Переменная штрихкода товара» = «{<Какая-то моя переменная>}», «Обработать штрихкод из сессии» = «Да», «Автоматический выбор первого товара из списка» = «Да».

Действие «Сообщение»

Действие позволяет выводить на экран сообщения для пользователя.

Отображаемый в сообщении текст задается с помощью свойства «Текст сообщения» и поддерживает применение текстовых шаблонов (подробнее о шаблонах см. «§ 11 Шаблоны текстов и математических выражений»).

Для визуального отображения является ли текст сообщением о какой-либо ошибке или нет, применяется «Отображать как ошибку». Сообщения ошибки отображаются белым текстом на красном фоне. Обычные сообщения – черным текстом на бежевом фоне.

Время отображения сообщения задается в секундах с помощью свойства «Время показа в секундах». Если время задано равным нулю – сообщение будет отображаться до нажатия пользователем любой клавиши.

Действие очистки данных

Действие, позволяющее удалить данные из сессии.

Иногда требуется удалить из сессии ранее введенные данные, чтобы они не нарушали процесс работы идущих далее действий. Для этих целей следует применять данное действие.

Действие «Новая упаковка»

Действие, предназначенное для добавления нового типа упаковки непосредственно с терминала.

Иногда необходимо присваивать новые штрихкоды для товаров непосредственно с терминала. В рамках системы такая возможность реализуется с помощью добавления новых упаковок номенклатуры.

Действие позволяет настроить автоматическое задание параметров упаковки с помощью свойств NewPackingAction.DefaultPackingName, NewPackingAction.DefaultQuantity и NewPackingAction. UseDefaultParams, либо позволить пользователю вводить параметры вручную.

Следует обратить внимание, что работа самого действия ограничивается заведением новой упаковки только на используемом мобильном терминале. Для корректной работы всей процедуры создания новой упаковки необходимо реализовать следующий порядок обработки и обмена данных:

1. Заведение упаковки для товара на мобильном терминале с помощью действия Cleverence.Warehouse.NewPackingAction. Все созданные упаковки хранятся на терминале и выгружаются на Сервер только во время процедуры обмена данными.
2. Загрузка созданных упаковок в учетную систему. Для этих целей используется функция StorageConnector.GetUserAddedProducts(), которая возвращает коллекцию продуктов с новыми упаковками.
3. После занесения новых упаковок в учетную систему следует провести процедуру выгрузки изменившихся позиций номенклатуры на Сервер.
4. Сразу же после выгрузки изменившихся товаров, необходимо передать Серверу соответствия между упаковками, заведенными на мобильных терминалах, и упаковками, выгруженными из учетной системы. Это позволит Серверу переопределить хранимые в нем документы, соответствующей заменой идентификаторов упаковок. Для этого используется функция StorageConnector. SetReplacementsForUserAddedProduct(). Ей передается коллекция объектов типа Cleverence.Warehouse.IdReplacement, содержащих соответствия между идентификаторами упаковок.
5. После этого Сервер уже автоматически сообщит на мобильные терминалы о сделанной подмене, что позволит им удалить у себя ставшие уже лишними упаковки.

В общем случае же рекомендуется избегать применения описанного выше функционала, так как вся процедура имеет достаточно много узких мест и потенциально может привести к ошибкам. Например, одна и та же физическая упаковка (или штрихкод) может быть заведена на двух терминалах сразу, что требует дополнительной обработки.

§ 11. Шаблоны текстов и математических выражений

Практически все тексты в окнах, любой расчет значений и обработчик шаблонов этикеток используют специальный синтаксис вычислимых выражений для вставки данных. Вычислимым выражением является всё, помещенное в фигурные скобки. Синтаксис выражения следующий:

{путь к значению:формат}

Шаблон текста для отображения в окне может содержать любое количество таких выражений вперемешку с обычным текстом. Например, «Выбрано: {ScanedBarcode} - {SelectedProduct}». Такая строка позволит выводить на экран сообщения вида «Выбрано: 2345069545 – картофель мороженый (пачка)», где ScanedBarcode и SelectedProduct заменяются соответствующими значениями, хранящимися в сессии.

Всё до первого двоеточия является указанием на то, что выводить, всё после – форматом вывода значения в строку. Формат зависит от типа значения – строка это, число, дата или что-то другое. Для правильного использования формата нужно знать, каков будет тип значения. Двоеточия может и не быть, т.е. часть с форматом необязательная – если формат не указан, то используется формат по умолчанию для соответствующего типа значений.

И синтаксис пути к значению, и синтаксис формата взяты из языка C# и спецификаций библиотеки Microsoft .NET. В .NET и, соответственно, в Mobile SMARTS всё является *объектами* – и строки, и числа, и более сложные вещи, такие как номенклатура, штрихкоды или складские документы. Источником данных для выражений является сессия исполнения документа, которая представляет собой просто набор переменных. Системные переменные и все переменные, заведенные разработчиком операции, складываются в одну кучу – сессию. В сессии лежит и сам обрабатываемый документ.

Поля в объекте, такие как номер документа или набор его строк, в .NET называются *свойствами объекта*. У каждого такого свойства есть своё имя. Зная имя свойства можно получить значение этого свойства у конкретного объекта. Это значение также будет являться объектом, у которого есть свои свойства и т.д., при этом одно имя свойства отделяется от другого точкой. Кроме свойств у объектов есть *методы* – это функции, значение которых вычисляет сам объект. Синтаксис вызова метода – имя метода с круглыми скобками «()», в которых указываются передаваемые параметры. Таким образом можно строить цепочки из обращения к свойствам и методам, чтобы получить из объекта нужные данные.

Соответственно, самый простой пример пути к значению – строка из имен свойств и методов со скобками, разделенных точками. Кроме того выражения могут содержать арифметические операции, т.е. не просто значение свойства или вызов метода, но и сумма значений двух свойств или произведение результата вызова метода на число + sin(10).

Следует всегда иметь в виду, что результатом обработки шаблона является текст. Поэтому, если a = 7 и b = 1, например, то не стоит путать следующие шаблоны:
«{a} – {b}» даст в результате «7 - 1», и только шаблон
«{a – b}» даст «6».

Другой из примеров пути к значению – указание специального объекта «global::», после которого идет указание на вызов статического метода или взятие значения статического свойства (.NET), а далее можно указать путь от полученного результата.

В том случае, если система не может вычислить значения или оно равно (null), ничего выведено не будет.

При задании шаблона можно указывать шаблон вывода значения. Например, «{SelectedProduct.ExpireDate: Срок годности – (0)}» отобразит что-то вроде «Срок годности – 01 мая 2009». А «{SelectedProduct.ExpireDate: Срок годности – (0:MM$$yyyy)}» отобразит что-то вроде «Срок годности – 05$$2009».

Если срок годности не задан (т.е. если поле не существует или равно null), то не будет выведено ничего: ни даты, ни текста «Срок годности – ». Это очень удобно для условного вывода значений. Можно указать целую страницу шаблонов для вывода различных возможных свойств номенклатуры, например, а

на экране получить строки только для тех свойств, которые реально присутствуют. Не существующие или не проставленные свойства, а также их «окаймление» в виде разных подписей и пояснений, просто не будут отображены.

Вот несколько примеров с пояснениями:

Текст в шаблоне	Пример результата и описание
Накладная №{Document.Id}	Накладная №1742 Выражение {Document.Id} было заменено на значение свойства с именем {Id} текущего документа, лежащего в сессии под именем «Document».
Длина номера: {Document.Id.Length} цифр	Длина номера: 4 цифр Выражение {Document.Id.Length} было заменено на значение свойства «Length» объекта (текстовой строки в данном случае), полученного как значение свойства с именем «Id» у документа.
Итого строк: {Document.CurrentItems.Count}	Итого строк: 16 Выражение {Document.CurrentItems.Count} было заменено на значение свойства «Count» табличной части фактических строк документа.
{Document.CurrentItems.Count:Итого строк: (0)}	Итого строк: 16 То же самое. Первое двоеточие – отделяет путь от формата. Второе – просто двоеточие. «(0)» в формате было заменено на результат выражения.
Результат: {Document.abcdefgh()} строк	Результат: строк У документа не существует метода abcdefgh, поэтому результатом выражения будет пустая строка.
{Document.abcdefgh():Результат: (0) строк} Дата: {global::System.DateTime.Today}	У документа не существует метода abcdefgh, поэтому результатом всего выражения в скобках будет пустая строка. Дата: 20.04.2009 У класса System.DateTime (дата и время) есть статическое свойство Today, которое всегда возвращает текущую дату (сегодняшнюю дату, которая сегодня).
{global::System.DateTime.Today.Day} !	4 ! У класса System.DateTime (дата и время) есть статическое свойство Today, которое всегда возвращает текущую дату. У текущей даты взято значение свойства Day (номер дня), а потом ко всему этому прибавились пробел и восклицательный знак.

Если при разборе строки из имен свойств и методов будет обнаружено, что таких свойств или методов нет или нет объектов, у которых их следует брать, в качестве результата выражения ничего не будет отображено.

По ссылкам ниже приведены описание доступных методов по работе с числами, строками и датами в .NET (на русском):

http://msdn.microsoft.com/ru-ru/library/system.string_members.aspx

http://msdn.microsoft.com/ru-ru/library/system.int32_members.aspx

http://msdn.microsoft.com/ru-ru/library/system.double_members.aspx

http://msdn.microsoft.com/ru-ru/library/system.decimal_members.aspx

http://msdn.microsoft.com/ru-ru/library/system.datetime_members.aspx

Форматы для вывода чисел

В таблице приведены только наиболее практичные форматы из спецификации .NET (http://msdn.microsoft.com/ru-ru/library/dwhawy9k.aspx и http://msdn.microsoft.com/ru-ru/library/0c899ak8.aspx) плюс специальные форматы, существующие только в Mobile SMARTS. Некоторые форматы используют числовые параметры – в тех местах, где можно вставить число, в таблице ниже используется *.

Формат	Описание	Примеры
N* или n*	Дробное число с ограничением на число знаков после запятой. Если ограничение не указано, то используется 2 знака после запятой.	{1000:N3} = 1000,000 {12.519:N} = {12.519:N2} = 12,52 {7:N} = {7:N2} = 7,00
C или c	Сумма в рублях	{100:c} = 100,00р. {1200.12:c} = 1 200,12р.
Набор из «0#.,»	Фиксированный формат вывода числа. «0» – обязательная цифра, «#» – необязательная цифра, «.» – десятичная точка, «,» – разделитель числовых разрядов. Все остальные символы ничего не означают и просто копируются в результат. Можно задать до трех секций формата, разделенных точкой с запятой. Первая секция – для положительных чисел, вторая – для отрицательных, третья – для ноля.	{2.5:0.00} = 2,50 {2.5:0.0#} = 2,5 {2.527:0.0#} = 2,53 {2.49:0.0;<red>-0.0</red>;ноль} = 2,5 {-2.49:0.0;<red>-0.0</red>;ноль} = -2,5 {0:0.0;<red>-0.0</red>;ноль} = ноль {17:0.#} = 17 {17.2:0.#} = 17,2 {17.2:000.00} = 017,20 {17.2:###.##} = 17,2 {10000:00,00.00} = 1 00 00,00
W или w	Количество прописью. W – с большой буквы, w – с маленькой	{2:W} = Два {100.24:w} = сто {1341:W} = Одна тысяча триста сорок один
Wf или wf	Количество прописью женского рода. W – с большой буквы, w – с маленькой	{2:W} = Две {100.24:w} = сто

		{1341:W} = Одна тысяча триста сорок одна
WRUR или wRUR	Сумма в рублях прописью. Если ноль копеек, то копейки не выводятся. W – с большой буквы, w – с маленькой	{2:WRUR} = Два рубля {100.247:wRUR} = сто рублей 25 копеек {1341:WRUR} = Одна тысяча триста сорок один рубль
Wrur или wrur	Сумма в рублях прописью с копейками. W – с большой буквы, w – с маленькой	{2:Wrur} = Два рубля 00 копеек {100.207:wrur} = сто рублей 21 копейка {1341:Wrur} = Одна тысяча триста сорок один рубль 00 копеек
WUSD, wUSD, Wusd или wusd	То же самое, что и с рублями RUR, но для долларов.	{2:WUSD} = Два доллара {100.207:wUSD} = сто долларов 21 цент {1341:Wusd} = Одна тысяча триста сорок один доллар 00 центов
RURc	Количество копеек (не округленно, а обрезано)	{100.207:RURc} = 20
USDc	Количество центов (не округленно, а обрезано)	{100.207:USDc} = 20

Форматы для вывода строк

В спецификации .NET у строк нет форматов вывода, но они есть в «Легком складе 3». Форматы используют числовые параметры – в тех местах, где можно вставить число, в таблице ниже используется *.

Формат	Описание	Примеры
T*	Обрезает строку до * символов	{«ABCD»:T3} = ABC {«ABCD»:T8} = ABC
E*	Обрезает строку до * символов и добавляет троеточие (…)	{«ABCD»:E3} = ABC… {«ABCD»:E8} = ABC

Форматы для вывода дат и времени

В .NET не существует отдельного типа «дата» и отдельного типа «время». Есть один общий тип «дата и время», поэтому везде, где упоминается дата, на самом деле имеется в виду дата и время, даже для даты документа. В таблице приведены наиболее практичные форматы из спецификации .NET (http://msdn.microsoft.com/ru-ru/library/az4se3k1.aspx и http://msdn.microsoft.com/ru-ru/library/8kb3ddd4.aspx).

Формат	Описание	Примеры
d	Короткий формат даты	{дата:d} = 20.04.2009
D	Длинный формат даты	{дата:D} = Понедельник, 20 Апреля 2009
ddd	День недели	{дата:ddd} = Понедельник
g	Короткий формат даты + короткий времени	{дата:g} = 20.04.2009 10:07

t	Короткий формат времени	{дата:t} = 10:07
T	Полный формат времени (с секундами)	{дата:T} = 10:07:12
Набор из «mMhHys»	Конкретный формат вывода даты и времени, составленный из следующих специальных обозначений и любых других символов:	
	«y» – последняя цифра года,	{дата:y абв} = 9 абв
	«yy» – последние две цифры года,	{дата:(0:yy)} = 09,
	«yyy» или «yyyy» – все цифры года,	{дата:yyyy} = 2009
	«M» – месяц прописью и день,	{дата:M} = апрель 20
	«MM» –две цифры месяца,	{дата:MM} = 04
	«MMM» – месяц прописью сокращенно,	{дата:MMM} = апр
	«MMMM» – месяц прописью,	{дата:MMMM} = Апрель
	«dd» – две цифры дня,	{дата:dd} = 20
	«ddd» – день прописью сокращенно,	{дата:ddd} = Пн
	«dddd» – день прописью,	{дата:ddd} = Понедельник
	«m», «mm», «mmm» и т.д. – минуты,	{дата:m} = {дата:mm} = 07
	«s», «ss», «sss» и т.д. – секунды,	{дата:s} = {дата:ss} = 12
	«f», «ff», «fff» и т.д. – миллисекунды,	{дата:f} = 9, {дата:ff} = 91, {дата:fff} = 912
	Используя эти обозначения можно составлять любые нужные форматы отображения.	{дата:MM dd} = 04 19 {дата:dd-MM-yy} = 19-04-09

Форматирование текста тегами псевдо-HTML

Тексты в окнах на терминале можно форматировать при помощи тегов псевдо-HTML. Вот примеры форматирования и того, как это будет выглядеть на экране:

Формат	Результат
<Green>Зеленый текст</Green>	Зеленый текст
<Green>Зелено-<Red>красный</Red> текст</Green> текст	Зелено-красный текст текст
<Green>Жирный зеленый текст</Green>	**Жирный зеленый текст**
Жирный зеленый текст	**Жирный зеленый текст**
<Red>Красный наклонный</Red>	Красный *наклонный*
<Red size="16">Большой красный</Red>	Большой красный
<r color="#0000FF">Синий наклонный <r	Синий *наклонный* большой

size="16">большой</r></r>	
наклонный <r>обычный</r> наклонный	*наклонный* обычный *наклонный*
первая строка вторая строка	первая строка вторая строка
первая строка вторая строка	первая строка вторая строка
Заголовок подзаголовок	**Заголовок** подзаголовок

§ 12. Дополнительные таблицы данных

Кроме уже встроенных в систему справочников, система позволяет создавать и использовать дополнительные таблицы с данными. Такие таблицы могут использоваться для совершенно различных задач: выгрузка дополнительных справочников, например, контрагентов или подразделений; хранение схемы расположения номенклатуры на складе и т.п.

Дополнительные таблицы бывают трех видов: таблица документа; серверная таблица; локальная таблица, загружаемая на ТСД.

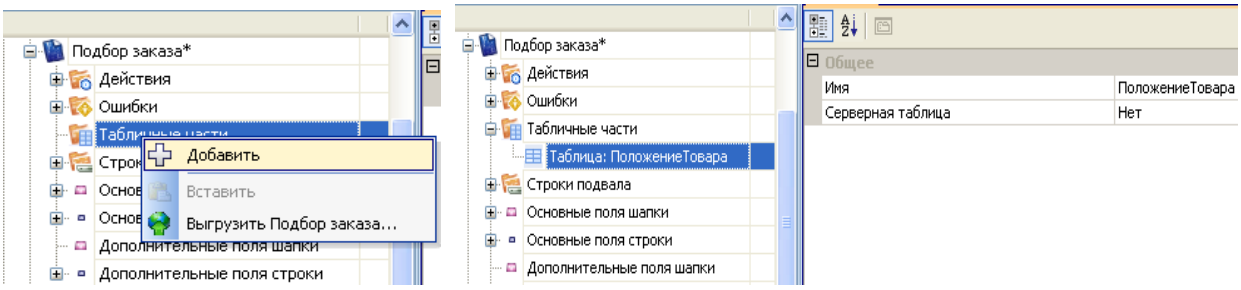
Рассмотрим все виды таблиц по порядку.

Таблицы документа

Таблица документа хранится непосредственно в самом документе и, соответственно, загружается на ТСД вместе с ним. Применять такие таблицы следует для хранения данных, используемых в контексте данного документа. К примеру, в документе подбора дополнительную таблицу можно использовать для хранения информации о расположении подбираемого товара на складе. Рассмотрим создание такой таблицы по шагам.

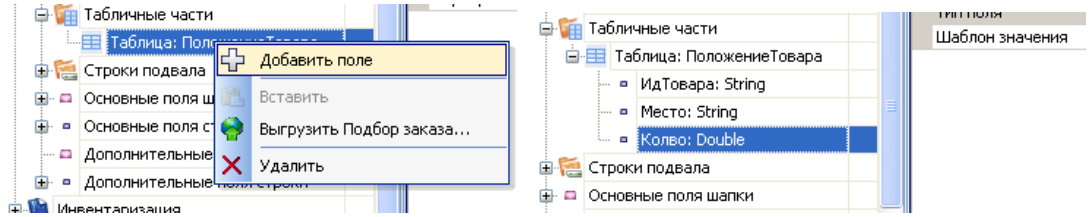
Создание таблицы

Для создания таблицы документа воспользуйтесь панелью управления:



Параметр «Серверная таблица» для таблиц документа не играет никакой роли. Они всегда загружаются на ТСД вместе с документом.

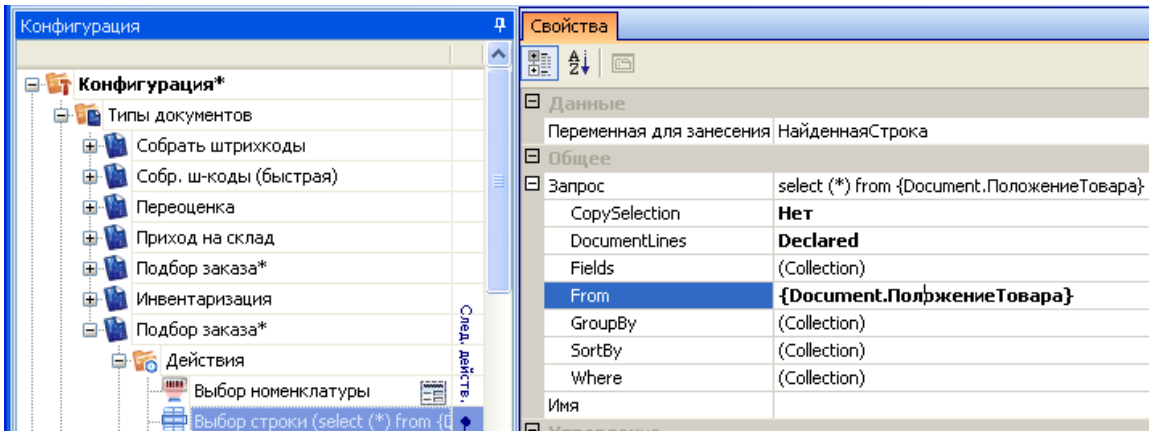
Заведите необходимые колонки таблицы и задайте им используемые типы данных:

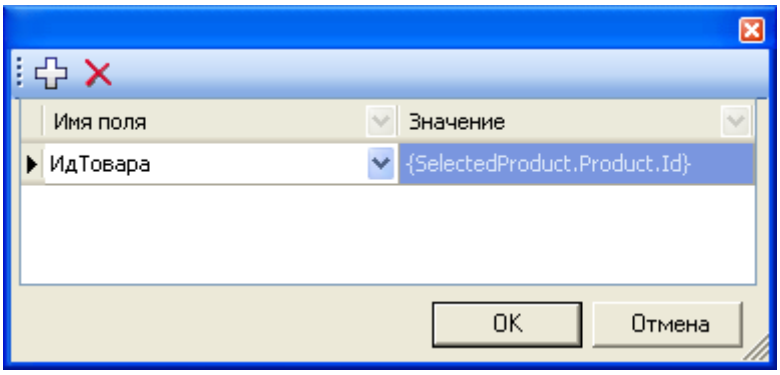


Поиск в таблице

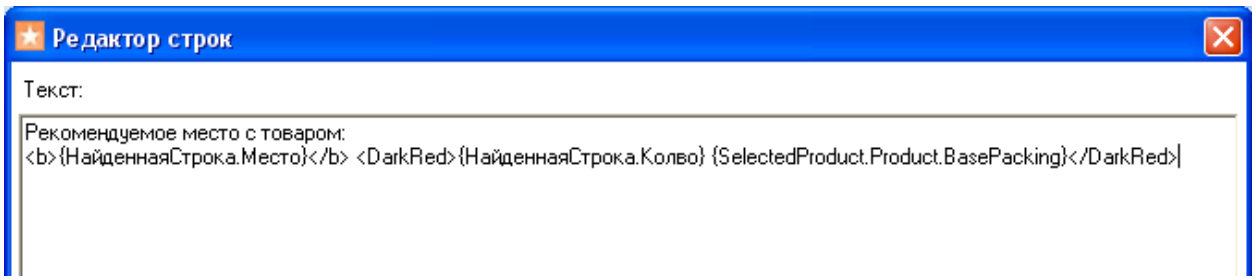
Для поиска строк в таблице следует использовать действия «Выбор строки» (для поиска одной строки) или «Выбор строк» (для выбора группы строк). В нашей задаче мы будем использовать выбор одной строки, чтобы найти возможное местоположение товара на складе.

Добавим действие «Выбор строки» после сканирования товара и зададим параметры поиска:





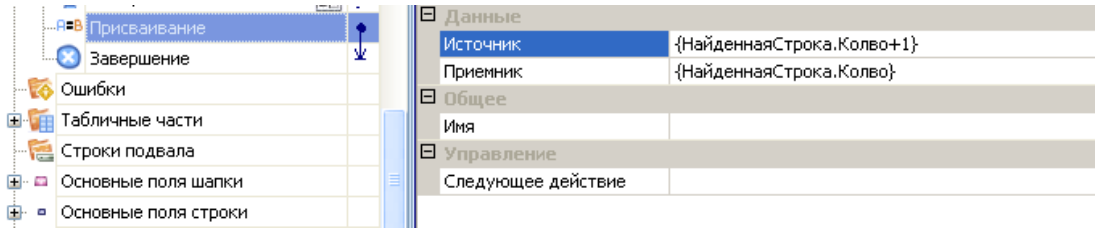
Теперь отобразим в заголовке следующего действия информацию из найденной строки:



Изменение данных в таблице

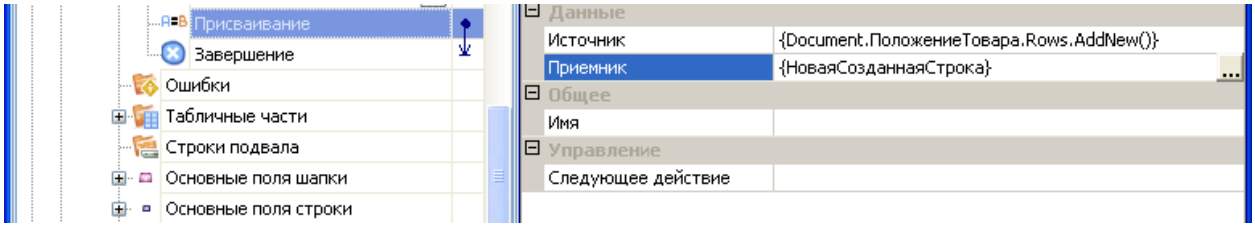
Важное отличие таблиц документа от других видов таблиц: возможность внесения изменений в строки таблицы на ТСД и добавление новых строк.

Правку таблицы можно осуществлять с помощью действия «Присваивание».



Для добавления новой строки в таблицу следует использовать вызов функции AddNew() для коллекции строк таблицы. Функция создает новую строку в таблице и возвращает ее в качестве результата.

Получение новой строки с помощью действия «Присваивание»:

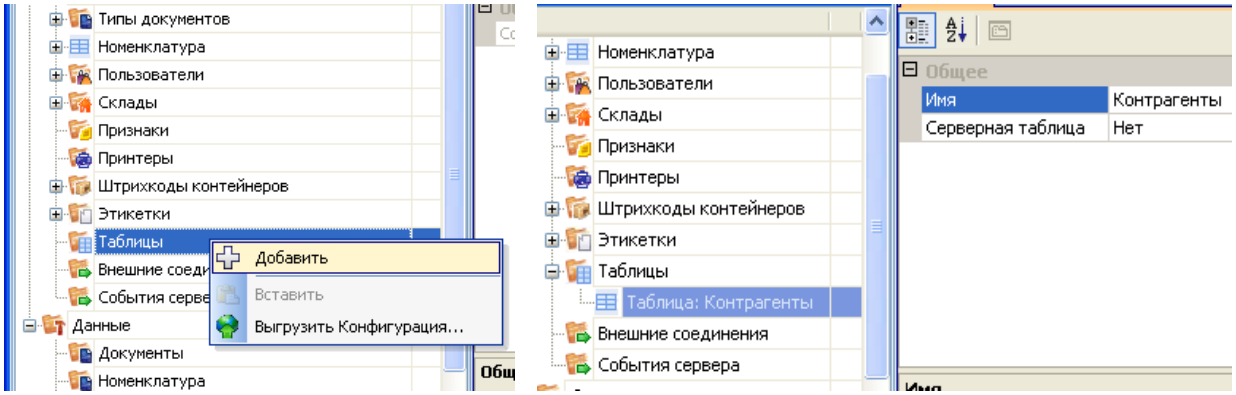


При завершении документа все измененные данные и новые строки вместе с документом попадут на сервер, и их можно будет загрузить и обработать в учетной системе.

Серверные и локальные таблицы

Серверные и локальные таблицы не привязаны к конкретному документу, доступ к их данным можно получить из любого процесса. Такие таблицы могут использоваться для хранения дополнительных справочников системы, например списка контрагентов или хранения остатков по местам хранения (не по конкретному документу, а полностью по всему складу).

Создадим таблицу для хранения контрагентов:



Параметр «Серверная таблица» определяет, будет ли таблица храниться на сервере или локально загружаться на ТСД при обмене данными. У обоих вариантов свои достоинства и недостатки: серверная таблица может использоваться только при наличии Wi-Fi сети, но при этом может быть любого размера и позволяет получать наиболее актуальные данные прямо с сервера. Локальная таблица загружается на терминал, поэтому может использоваться в batch режиме, но это накладывает ограничения на размер таблицы из-за того что она, загружаясь на терминал, занимает программную память терминала. Мы не рекомендуем использовать локальные таблицы, если количество записей в ней будет более 2000.

Поиск в таблице

Также как и в таблицах документа, для поиска в серверной или локальной таблице следует использовать действия «Выбор строки» или «Выбор строк».

В качестве параметра **From** в запросе следует указать шаблон с именем таблицы, например **{Контрагенты}**.

Изменение данных в таблице

В отличие от таблиц документа изменение данных в серверных или локальных таблицах с ТСД – **невозможно**.

Отображение данных таблиц в виде списка

Отображение данных в виде списка доступно только для локальных таблиц на ТСД или таблиц документа.

Серверные таблицы такую возможность **не поддерживают**.

Для отображения данных таблицы и выбора строки пользователем следует использовать действие «Просмотр записей», также как и для строк документа. В качестве параметра **From** в запросе следует указать шаблон с именем таблицы, например **{Контрагенты}**.

Глава 4. Интеграция с учетной системой

§ 1. Вводная в объекты Mobile SMARTS

В работе с Mobile SMARTS используется множество различных объектов, которые можно условно разделить на две большие группы: *бизнес-сущности* и *вспомогательные объекты*. Бизнес-сущности системы, такие как товар, упаковка, склад, пользователь, шаблон этикетки и т.д., представляют собой данные, хоть как-то отражающие бизнес и объекты учета учетной системы. Вспомогательные сущности, такие как StorageConnector, состояние терминала и т.д., имеют отношение только непосредственно к работе Mobile SMARTS.

Все объекты системы обладают определенным набором свойств и методов. Все доступные объекты, а также их свойства и методы, перечислены в файле справки «*Mobile SMARTS 2008 Компонента достуna.chm*».

Бизнес-сущности

Почти у всех бизнес-сущностей есть свойство (чаще всего, Id), значение которого должно быть уникально, что требуется для поддержания связей между объектами. Например, каждый товар ссылается на свой базовый тип упаковки посредством свойства Product.BasePackingId, а каждый пользователь системы (Cleverence.Warehouse.User) имеет три уникальных поля: код (User.Id), имя (User.Name) и штрихкод (User.Barcode). Первое используется для указания на конкретного пользователя (например, исполнитель в объекте документа), а второе для однозначной авторизации пользователя при входе в клиентское приложение Mobile SMARTS.

Для большинства сущностей системы предусмотрены соответствующие коллекции объектов (т.е. пополняемые списки). Например, для сущности товара (Cleverence.Warehouse.Product) существует коллекция товаров - Cleverence.Warehouse.ProductCollection. Коллекции – это не справочники, а типы данных, наподобие массивов.

Коллекции имеют функции для добавления (Add) и удаления (Remove) объектов, получения конкретного объекта по его индексу в коллекции (Item) и получения текущего количества объектов в коллекции (Count). Кроме того, коллекция позволяет искать и фильтровать содержащиеся в ней объекты. Для этих целей каждому свойству бизнес-сущности в коллекции соответствует функция вида FindBy...(...). Например, для свойства User.Name в UserCollection существует функция FindByName(string name). Если, как в данном примере, значение свойства уникально, то функция возвращает единичный объект (в данном случае User), либо пустое значение (null, т.е. объект не найден). Если же свойство не подразумевает уникальности, как например свойство артикула для товара (Product.Marking), то результатом вызова ProductCollection.FindByMarking(string marking) будет новая коллекция товаров, артикул которых соответствует параметру marking.

В таблице ниже приведен полный список всех бизнес-сущностей Mobile SMARTS:

Наименование	Перевод	Описание
Product	Товар (ТМЦ, Номенклатура)	Задает позицию справочника товаров.
Packing	Упаковка	Задает вариант упаковки товара с собственным штрихкодом, весом и т.д.

Unit	Единица измерения	«шт.», «кг», «м» и т.д.
Warehouse	Склад	Один из существующих складов Компании.
Cell	Ячейка	Описание конкретной ячейки, шаблона большой группы ячеек или зоны хранения на конкретном складе.
Pallet	Паллета	Описание конкретной паллеты либо шаблона большой группы паллет.
Document	Документ	Электронный документ, связанный с определенной складской операцией.
DocumentType	Тип документа	Задает тип операции на мобильном терминале и соответствующий ей тип электронного документа с описанием алгоритма выполнения документов такого типа на мобильном терминале.
Classifier	Признак	Задает признак, который может быть назначен объекту системы. Позволяет расширить данные о каком-то объекте, не добавляя новых полей во все объекты бизнес-сущности.
ClassifierType	Тип признака	Задает тип используемых признаков, уточняя их применение к различным бизнес-сущностям.
QuantityPolicy	Политика количества товара	Задает вариант ввода и отображения количества определенных товаров.
User	Пользователь	Пользователь Mobile SMARTS со стороны мобильных терминалов со своим именем, паролем, рабочими складами и т.д.
UserGroup	Группа пользователей	Группа пользователей Mobile SMARTS со стороны мобильных терминалов, по которой назначаются доступные операции.
Printer	Принтер	Имя и сетевой путь к принтеру.
PrinterMapping	Привязка принтера	Задает правило выбора принтера в зависимости от сочетания пользователя, склада, где он находятся, и типа документа, из обработки которого производится печать.
LabelTemplate	Шаблон этикетки	Дизайн-макет этикетки для печати на принтерах этикеток (и, вообще говоря, любых windows-принтерах) с шаблонами полей для вставки данных.

Вспомогательные объекты

Вспомогательные объекты используются в основном для обмена данными с сервером Mobile SMARTS при получении данных о терминалах, печати этикеток, вызове окон настроек и просмотра метаданных Mobile SMARTS и т.д.

В таблице ниже даны некоторые примеры вспомогательных объектов Mobile SMARTS:

Наименование	Перевод	Описание
StorageConnector	Коннектор	Содержит все методы для обмена данными с сервером Mobile SMARTS.
Label	Этикетка	Задает значения полей при печати конкретной этикетки.

§ 2. Механизм обмена данными

Для организации обмена данными между учетной системой пользователя и Mobile SMARTS применяется специальная COM компонента. После ее установки и регистрации в ОС Windows становится возможным создание и операции с объектами компоненты, непосредственно в процедурах учетной системы.

Все средства доступа к серверу Mobile SMARTS сосредоточены в одном единственном COM-компоненте Cleverence.Warehouse.StorageConnector.

Операция создания объектов, описанных в COM, специфична для каждой системы:

Псевдокод:

```
var connector = новый Cleverence.Warehouse.StorageConnector();
```

«1С:Предприятие 7»:

```
connector = СоздатьОбъект("Cleverence.Warehouse.StorageConnector");
```

«1С:Предприятие 8»:

```
connector = новый COMОбъект("Cleverence.Warehouse.StorageConnector");
```

Microsoft Dynamics AX (Axapta):

```
var connector = new COM("Cleverence.Warehouse.StorageConnector");
```

StorageConnector содержит функции для выгрузки/загрузки справочников и документов, а также для работы со специальными возможностями системы (управление терминалами, редактирование и печать этикеток и так далее). Все объекты метаданных Mobile SMARTS и прочие объекты Mobile SMARTS в рамках компоненты доступа к серверу Mobile SMARTS также представлены в виде COM-компонентов.

После создания, к COM-объекту могут применяться операции чтения и записи свойств и вызов его функций, в соответствии с синтаксисом языка разработки применяемой учетной системы.

Посмотреть все установленные в системе COM компоненты, их свойства и функции, возможно с помощью утилиты oleview.exe компании Microsoft.

Если утилита отсутствует в системе, ее можно бесплатно скачать по следующим адресам:
Для Windows 2000:

http://www.microsoft.com/windows2000/techinfo/reskit/tools/existing/oleview-o.asp
Для Windows XP и Windows Server 2003:
http://www.microsoft.com/downloads/details.aspx?FamilyID=9d467a69-57ff-4ae7-96ee-b18c4790cffd

Сервер Mobile SMARTS принимает обращения и возвращает результаты только в виде XML-документов. Формат этих документов может меняться от версии к версии, и нигде не документирован. StorageConnector облегчает работу прикладного программиста, автоматически транслируя графы объектов Mobile SMARTS в XML-документ и наоборот.

В процессе интеграции, разработчик создает код учетной системы, который создает и заполняет некоторый объект Mobile SMARTS, иногда достаточно сложный, в соответствии с данными учетной системы, и отправляет его на сервер Mobile SMARTS при помощи вызова StorageConnector. При загрузке данных происходит обратная операция – анализ объекта Mobile SMARTS и модифицирование на его основе данных учетной системы.

Доступ к серверу

После создания StorageConnector в обязательном порядке требуется провести его инициализацию с помощью функции InitializeServerConnection.

C#:

```
void InitializeServerConnection(string connectionString)
void InitializeProxyServerConnection(string connectionString,
                                     string proxyName,
                                     int port)
```

Строка соединения connectionString представляет собой url к Web-сервису сервера Mobile SMARTS и имеет следующий формат:

```
http://DNS-имя-сервера[:порт для доступа]/путь-к-странице/DataStorage.asmx
```

Самый первый шаг на пути к интеграции – это попытка выполнить следующий код:

C#:

```
Cleverence.Warehouse.StorageConnector connector =
                                    new Cleverence.Warehouse.StorageConnector();
connector.InitializeServerConnection(
      "http://server:8000/DataStorage.asmx");
```

«1С:Предприятие 7»:

```
// Создание объекта коннектора и инициализация его соединения с сервером.
connector = СоздатьОбъект("Cleverence.Warehouse.StorageConnector");
connector.InitializeServerConnection(
                     СокрЛП(Константа.ПолучитьАтрибут("СтрокаПодключения")));
```

«1С:Предприятие 8»:

```
// Создание объекта коннектора и инициализация его соединения с сервером.
connector = новый COMОбъект("Cleverence.Warehouse.StorageConnector");
connector.InitializeServerConnection(Константы.СтрокаПодключения);
```

Microsoft Dynamics AX (Axapta):

```
// Создание объекта коннектора и инициализация его соединения с сервером.
COM connector = new COM("Cleverence.Warehouse.StorageConnector");
connector.InitializeServerConnection(EasyWarehouse.GetConnectionString());
```

Если доступ к серверу возможен только через прокси, следует использовать второй вариант этой функции.

Если объект создан успешно и InitializeServerConnection выполнилась без ошибок, можно переходить к попытке выгрузить некоторые справочники.

Что могло случиться, если экземпляр StorageConnector создать не удалось:

Ошибка

Возможная причина	Диагностика	Решение
Создание экземпляра Cleverence.Warehouse.StorageConnector завершилось с ошибкой «Cleverence.Warehouse.StorageComConnector were not registered» («COM-компонент не зарегистрирован») или какой-то подобной.		
На компьютере не установлена компонента доступа.	В «Установке и удалении программ» нет записи о «Mobile SMARTS 2.x - Компонента доступа».	Найти дистрибутив Mobile SMARTS и произвести установку компоненты доступа.
Компонента установлена, но часть файлов отсутствует.	Папка установки компоненты доступа отсутствует либо пуста (и в ней нет файла Cleverence.Warehouse.StorageComConnector.dll).	Найти дистрибутив Mobile SMARTS, через «Установку и удаление программ» удалить компоненту доступа и установить ее заново.
Компонента установлена, но регистрация COM не произошла.	А) В Oleview нет компонента «Cleverence.Warehouse.StorageComConnector», либо Б) В системном реестре (regedit.exe) нет ни одной записи, содержащей строку «Cleverence.Warehouse.StorageComConnector».	Произвести регистрацию вручную, запустив из папки компоненты доступа файл «Зарегистрировать COM-компонент StorageComConnector.bat». Если файл регистрации не может найти путь «Framework. v1.1.4322» или файл «RegAsm.exe», необходимо переустановить .NET Framework 1.1.
Создание экземпляра Cleverence.Warehouse.StorageConnector завершилось с ошибкой «Cannot resolve Cleverence.Warehouse.StorageComConnector assembly or one of it's references» («Не могу найти сборку Cleverence.Warehouse.StorageComConnector или одну из используемых ею») или какой-то подобной.		
Компонента установлена, но часть файлов отсутствует.	Папка установки компоненты доступа отсутствует, либо заполнена только на половину (и в ней нет главного файла - Cleverence.Warehouse.StorageComConnector.dll).	Найти дистрибутив Mobile SMARTS, через «Установку и удаление программ» удалить компоненту доступа и установить ее заново.
Вызов метода InitializeServerConnection завершилось с ошибкой «Сервер не найден» или какой-то подобной.		
Неверно указана строка подключения.	Скопировать строку подключения в браузер и дожидаться загрузки страницы Web-сервиса. Браузер выдаст ошибку «The page cannot be displayed» («Невозможно отобразить страницу»)	Выяснить точную строку подключения. Например, на компьютере сервера запустить Internet Information Services (IIS)

	или «Server Error in '/' Application. The resource cannot be found» («Ошибка в приложении '/', ресурс не найден»).	Manager (inetmgr.exe), найти виртуальную папку Cleverence.Warehouse.DataService, найти в ней файл DataStorage.asmx и выбрать в контекстном меню команду «Browse» («Просмотр»).
Расширения ASP.NET 1.1 не были зарегистрированы в MS IIS.	Microsoft Internet Information Services (IIS) был установлен уже после установки .NET Framework 1.1. Скопировать строку подключения в браузер и дожидаться загрузки страницы Web-сервиса. Браузер выдаст ошибку «The page cannot be found» («Страница не найдена»).	Произвести регистрацию вручную, запустив из папки сервера файл «Зарегистрировать ASP.NET под MS IIS.bat». Например, на компьютере сервера запустить Internet Information Services (IIS) Manager (inetmgr.exe), найти виртуальную папку Cleverence.Warehouse.DataService и выяснить, где она расположена физически, выбрав в контекстном меню команду «Properties…» («Свойства…»). Если путь к папке выглядит как «\Cleverence.Warehouse.DataService», ее следует искать в папке «Inetpub\wwwroot» диска, на котором установлена операционная система (чаще всего, это диск C:, но может быть и по-другому).
В локальной сети установлен прокси-сервер для доступа к HTTP, а разработчик об этом не знает.	В настройке Internet Explorer, Opera, Firefox и других браузерах указаны настройки прокси. Например, в IE в закладке «Tools\Internet Options…\Connections\LAN Settings…» («Сервис\Свойства обозревателя…\Подключения\Настройка LAN…») что-то сказано про «Use a proxy server for…» («Использовать прокси-сервер для…»).	Выяснить настройки прокси-сервера и внести их в вызов метода InitializeServerConnection.
Настройки безопасности доступа к папке с Web-сервисом Mobile SMARTS настроены неверно.	Скопировать строку подключения в браузер и дожидаться загрузки страницы Web-сервиса. Браузер выдаст ошибку «You are not authorized to view this page» («Доступ к странице запрещен»).	На компьютере сервера запустить Internet Information Services (IIS) Manager (inetmgr.exe) и убедиться в правильности политики доступа к виртуальной папке Cleverence.Warehouse.DataService. На компьютере сервера найти физическую папку Cleverence.Warehouse.DataService (например, Inetpub\wwwroot\Cleverence…Dataservice) и настроить политику доступа для локальных и удаленных пользователей по всем файлам и подпапкам.
На компьютере установлен ISA-клиент.	В трее виден вот такой  или вот такой  значок.	Временно отключить Firewall Client, чтобы более точно диагностировать ошибку: Если с выключенным Firewall Client вызов работает, выяснить настройки прокси-сервера и локальных политик и внести их в вызов метода InitializeServerConnection. Если нет – искать другую причину.

Выгрузка и загрузка данных

Рассмотрим небольшой пример выгрузки данных с использованием компоненты доступа, в котором мы выгрузим на сервер Mobile SMARTS одну-единственную позицию номенклатуры «Товар1»:

C#:

```
using Cleverence.Warehouse;
…
// Создание объекта продукта «Товар1»
Product продукт = new Cleverence.Warehouse.Product();
продукт.Id = "1";
продукт.Name = "Товар1";
продукт.Marking = "АРТИКУЛ1";

// Создание объекта упаковки для товара
Packing упаковка = new Cleverence.Warehouse.Packing();
упаковка.Id = "1";
упаковка.Name = "шт";
упаковка.UnitsQuantity = 1;

продукт.Packings.Add(упаковка);
продукт.BasePackingId = "1";

// Создание коннектора, для подключения к серверу
StorageConnector connector = new Cleverence.Warehouse.StorageConnector();
connector.InitializeServerConnection("http://server:8000/DataStorage.asmx");

// Выгрузка заполненного товара на сервер
connector.SetProduct(продукт);
```

«1С:Предприятие 7»:

```
// Создание объекта продукта «Товар1»
продукт = СоздатьОбъект("Cleverence.Warehouse.Product");
продукт.Id = "1";
продукт.Name = "Товар1";
продукт.Marking = "АРТИКУЛ1";

// Создание объекта упаковки для товара
упаковка = СоздатьОбъект("Cleverence.Warehouse.Packing");
упаковка.Id = "1";
упаковка.Name = "шт";
упаковка.UnitsQuantity = 1;

продукт.Packings.Add(упаковка);
продукт.BasePackingId = "1";

// Создание коннектора, для подключения к серверу
connector = СоздатьОбъект("Cleverence.Warehouse.StorageConnector");
connector.InitializeServerConnection("http://server:8000/DataStorage.asmx");
```

```
// Выгрузка заполненного товара на сервер
connector.SetProduct(продукт);
```

«1С:Предприятие 8»:

```
// Создание объекта продукта «Товар1»
продукт = новый COMОбъект(“Cleverence.Warehouse.Product”);
продукт.Id = “1”;
продукт.Name = “Товар1”;
продукт.Marking = “АРТИКУЛ1”;

// Создание объекта упаковки для товара
упаковка = новый COMОбъект(“Cleverence.Warehouse.Packing”);
упаковка.Id = “1”;
упаковка.Name = “шт”;
упаковка.UnitsQuantity = 1;

продукт.Packings.Add(упаковка);
продукт.BasePackingId = “1”;

// Создание коннектора, для подключения к серверу
connector = новый COMОбъект("Cleverence.Warehouse.StorageConnector");
connector.InitializeServerConnection("http://server:8000/DataStorage.asmx");

// Выгрузка Выгрузка заполненного товара на сервер
connector.SetProduct(продукт);
```

Microsoft Dynamics AX (Axapta):

```
// Создание объекта коннектора
COM connector = new COM("Cleverence.Warehouse.StorageConnector");
// Создание объекта товара
COM product = new COM("Cleverence.Warehouse.Product");
// Создание объекта упаковки
COM packing = new COM("Cleverence.Warehouse.Packing");
// Дает доступ с свойству product.Packings
COM packings;

// Заполнение объекта продукта «Товар1»
product.Id(“1”);
product.Name(“Товар1”);
product.Marking(“АРТИКУЛ1”);

// Заполнение объекта упаковки
packing.Id(“1”);
packing.Name(“шт”);
packing.UnitsQuantity(1);
packings = product.Packings();
```

```
packings.Add(packing);
product.BasePackingId("1");

// Инициализация соединения с сервером.
connector.InitializeServerConnection("http://server:8000/DataStorage.asmx");
// Выгрузка заполненного товара на сервер
connector.SetProduct(product);
```

Код в примере простой, но он содержит основные элементы любой выгрузки, а именно создание коллекции под выгрузку, ее наполнение, и отправка на сервер.

Если все было сделано правильно, сервер Mobile SMARTS доступен и работает, а вызов SetProduct прошел без ошибок, поздравляем! Можно зайти в папку «C:\Program Files\Cleverence Soft\Mobile SMARTS 2008\Server\Documents\» на физическом сервере Mobile SMARTS и поискать там файл «Cleverence.Warehouse.ProductsBook.xml». Если открыть его с помошью Internet Explorer или какого-нибудь редактора XML-файлов, можно будет поближе познакомиться со структурой хранения данных Mobile SMARTS.

Что могло случиться, если функция завершилась с ошибкой:

Ошибка		
Возможная причина	Диагностика	Решение
Создание экземпляра Cleverence.Warehouse.StorageConnector завершилось с ошибкой.		
Со времен чтения раздела «Доступ к серверу» конфигурация вашего компьютера изменилась, либо это уже другой компьютер.		Вернуться к разделу «Доступ к серверу» и ошибкам подключения.

§ 3. Окружение системы

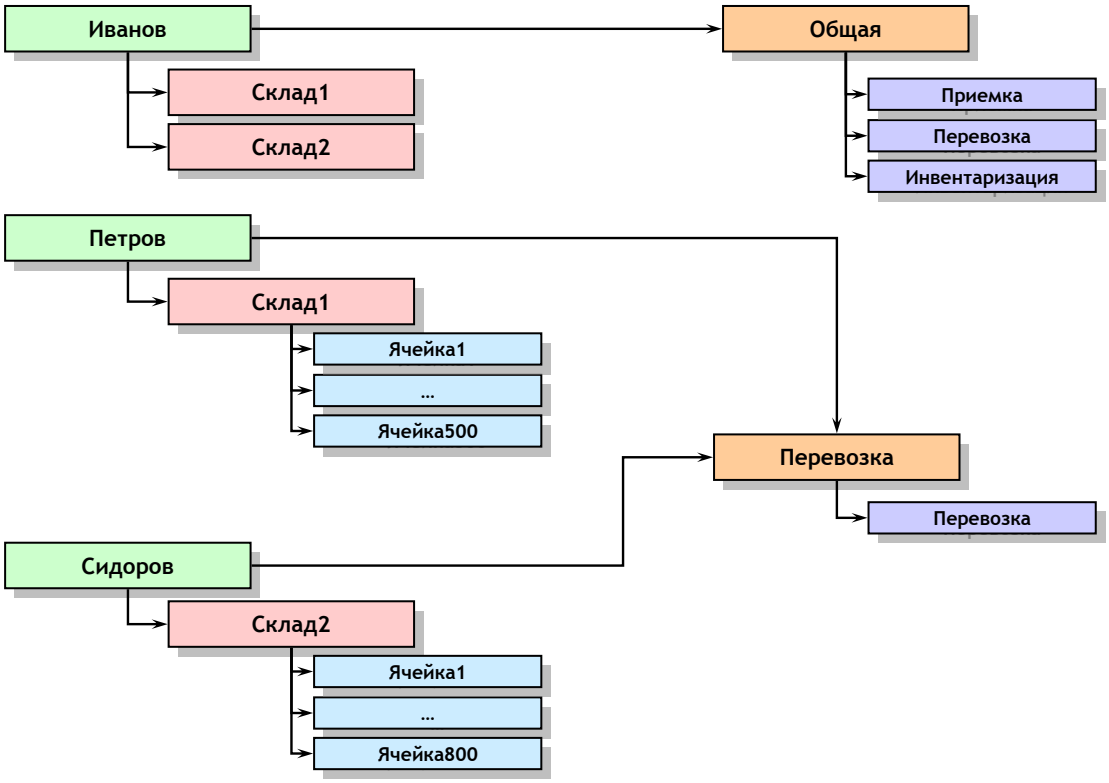
Для общего конфигурирования системы Mobile SMARTS и настройки правил ее поведения используется специальный объект – *Окружение системы* (Cleverence.Warehouse.Environment и соответствующий файл Cleverence.Warehouse.Environment.xml).

Этот объект предоставляет разработчику доступ к настройкам схемы складов и ячеек, пользователей, их групп и доступных им типов документов. Для редактирования объекта окружения системы используется визуальный редактор метаданных. Программная работа с объектом окружения напрямую через API может понадобиться в редчайших случаях, однако полезно иметь о нем хотя бы общее представление.

Mobile SMARTS поддерживает следующую схему конфигурирования:

- Все пользователи системы разбиты на группы.

- Каждая группа определяет типы документов, с которыми пользователь может работать.
- Каждому пользователю задаются склады, на которых он может работать.
- Каждый склад содержит список его ячеек.



На приведенном выше рисунке показан пример для трех пользователей. Два из них (Петров и Сидоров) входят в группу «Перевозка», и им доступна только одна соответствующая операция. Обязанности этих пользователей также ограничены складом, который к ним прикреплен. Пользователь Иванов, в отличие от них, имеет возможность работы с тремя различными операциями и на обоих складах.

Каждый пользователь (Cleverence.Warehouse.User) носит уникальное имя (User.Name), идентифицирующее его.

Клиентское приложение использует две схемы авторизации пользователя: по ручному вводу имени и пароля, либо с помощью сканирования штрихкода пользователя. Метод авторизации устанавливается для всех пользователей сразу с помощью свойства Environment.BarcodeLogin.

Группы пользователей

Как уже было сказано выше, все пользователи системы Mobile SMARTS разбиваются на группы (Cleverence.Warehouse.UserGroup).

Группа содержит список типов документов, доступных пользователям группы, тем самым, описывая стоящие перед ними задачи.

Кроме того, группа задает тип обмена данными (свойство UserGroup.BatchMode) и место расположения справочника номенклатуры (свойство UserGroup.ServerSideInventory): на Сервере, что требует постоянной связи с ним, либо на самом клиенте.

Склады и ячейки

Конфигурация складов и ячеек производится для каждого пользователя индивидуально, что позволяет пространственно ограничить доступную ему зону работы.

Каждый склад содержит коллекцию своих ячеек.

Каждая ячейка склада характеризуется индивидуальным штрихкодом и своим именем.

Для экономии ресурсов мобильного терминала вместо выгрузки большого числа ячеек система позволяет применять шаблоны штрихкодов, аналогичные шаблонам, используемым в упаковках и паллетах.

К примеру, можно добавить ячейку со следующими параметрами:

```
cell.Barcode = "8{a:2}{b:2}{c:1}";
cell.Name = "ряд - {a}, позиция - {b}, этаж - {c}";
```

Такой объект будет описывать группу ячеек со штрихкодом длиной 6 символов, где 2 и 3 символы – номер ряда, 4 и 5 – номер позиции, 6 – этаж.

Выгрузка среды

Выгрузка созданного и заполненного объекта среды осуществляется с помощью функции StorageConnector.SetEnvironment(Environment environment).

```
//Создание объекта "среды"
environment = СоздатьОбъект("Cleverence.Warehouse.Environment");
environment.BarcodeLogin = barcodeCheck;
environment.ExitPassword = "123123";

//Создаем склад и его ячейки
warehouse = СоздатьОбъект("Cleverence.Warehouse.Warehouse");
warehouse.Id   = СокрЛП("1");
warehouse.Name = СокрЛП("Склад");
cell = СоздатьОбъект("Cleverence.Warehouse.Cell");
cell.Barcode = "1{d:2}{a:2}{b:1}";
cell.Name = "A-{d}-{a}-{b}";
warehouse.Cells.Add(cell);

//Создание "групп"
group = СоздатьОбъект("Cleverence.Warehouse.UserGroup");
group.Name = "Общая";
group.BatchMode = 0;
group.ServerSideInventory = 1;
environment.Groups.Add(group);

//Создание нужных типов документов
СоздатьТипыДокументов(group);

//Создание пользователей
спрПользователиЛС.ВыбратьЭлементы();
Пока спрПользователиЛС.ПолучитьЭлемент() = 1 Цикл
```

```
        user = СоздатьОбъект("Cleverence.Warehouse.User");
        user.Name = СокрЛП(спрПользователиЛС.Пользователь.Наименование);
        user.Password = СокрЛП(спрПользователиЛС.Пользователь.Пароль);
        user.Barcode = СокрЛП(спрПользователиЛС.Пользователь.Пароль);
        user.GroupName = "Общая";
        user.Description = "";
        user.Warehouses.Add(warehouse);
        //Добавление пользователей в среду
        Попытка
              environment.Users.Add(user);
        Исключение
              Сообщить("Ошибка!!! Дублируется штрихкод у " + user.Name + "!");
        КонецПопытки;
КонецЦикла;

//Отсылка "среды" в "Mobile SMARTS"
connector.SetEnvironment(environment);
```

Приведенный пример иллюстрирует простейшее заполнение объекта среды, с созданием одной, общей группы и выбором пользователей Mobile SMARTS из специально заведенного справочника. В примере опущена процедура создания типов документов (функция СоздатьТипыДокументов(group)), так как этой теме отведена отдельная глава.

§ 4. Номенклатура и штрихкоды товаров

Для хранения информации о товарах и их количественных характеристиках в системе используются справочник номенклатуры.

Каждая позиция номенклатуры (Cleverence.Warehouse.Product) в системе Mobile SMARTS содержит информацию о наименовании, артикуле, базовом штрихкоде, а также типах упаковки товара.

Для каждого товара в системе может быть задано несколько типов упаковки товара (Cleverence.Warehouse.Packing), например пачка, блок и коробка.

Упаковка имеет собственный штрихкод, который может отличаться от штрихкода самого товара. Этот штрихкод также идентифицирует товар, однако работа системы с ними несколько отличается. При сканировании в клиентском приложении базового штрихкода товара, пользователю будет предложено выбрать тип упаковки (если их несколько), либо вычисления будут производиться на основе основного типа упаковки, указанного в свойстве Product.BasePackingId. При сканировании штрихкода упаковки, выбор именно этой упаковки будет сделан системой автоматически; останется только ввести количество. Также следует учитывать что как базовый штрихкод, так и штрихкоды упаковок могут быть пустыми.

Приоритетным для системы является базовый штрихкод. Поэтому, если штрихкод у упаковки тот же, что и базовый штрихкод товара, при сканировании будет считаться, что выбран базовый штрихкод, а упаковку будет предложено выбрать из списка.

«1С:Предприятие 7»:

```
...
// Создание продукта и заполнение его полей значениями из справочника
// учетной системы
      product      = СоздатьОбъект("Cleverence.Warehouse.Product");
```

```
      product.Id    = СокрЛП(спрНоменклатура.Код);
      product.Name  = СокрЛП(спрНоменклатура.Наименование);
      product.Barcode = СокрЛП(спрНоменклатура.ШтрихКод);
      product.Marking = СокрЛП(спрНоменклатура.Артикул);
...
// Создание упаковки для продукта и заполнение ее свойств
      packing       = СоздатьОбъект("Cleverence.Warehouse.Packing");
      packing.Id    = СокрЛП(спрШтрихКод.Код);
      packing.UnitsQuantity = спрЕдиницы.Коэффициент;
      packing.Barcode = СокрЛП(спрШтрихКод.Наименование);
...

// Добавление упаковки к продукту
      product.Packings.Add(packing);
...
// Задание базовой упаковки товара
      product.BasePackingId = packing.Id;
```

Для корректной работы системы, ***каждая позиция номенклатуры должна иметь хотя бы одну упаковку***. В случае если учетная система не предусматривает наличие упаковок, либо их аналогов, следует для каждого товара добавлять фиктивную упаковку. Также необходимо ***в обязательном порядке*** проставить идентификатор базовой упаковки для товара.

Шаблоны штрихкода

Дополнительная возможность, предоставляемая упаковками, – возможность применять шаблоны штрихкода. Это позволяет вводить в систему нефиксированные штрихкоды, содержащие значимую информацию (например, срок годности товара или его количество). При сканировании будет производиться сравнение штрихкода на соответствие заданному шаблону и, в случае соответствия, происходить заполнение извлеченных из него данных.

Такой шаблон описывается в виде *###{шаблон}#…{шаблон}…###*, где *###* - некоторое количество фиксированных символов.

В качестве {шаблон} используются специальные выражения вида {имя:формат}, где имя задает имя параметра, а шаблон зависит от типа параметра. Если параметр – это строка или число, то в качестве шаблона можно указать количество символов штрихкода, отводимое под параметр.

В качестве имени можно использовать что угодно, в том числе и русские названия с пробелами. Все параметры и их значения попадут в сессию в качестве переменных. При занесении новой строки в документ те переменные сессии, имена которых совпадают с именами колонок в документе, попадут в поля этой новой строки. Т.е. если мы хотим сохранить в документе часть штрихкода, нужно 1) добавить в типе документа дополнительное поле строки, и 2) создать шаблон штрихкода, в котором упоминалось бы наименование этого поля.

Вот некоторые примеры использования шаблонов:

Шаблон	Описание

{Quantity:число}, число – количество символов в штрихкоде или два числа, разделенных точкой (целая и дробная части)	Позволяет точно определить количество сканируемого товара, избегая его ввода вручную. Переменная Quantity используется действием ввода количества товара, а также действиями занесения сток в документ. Пример шаблона: 234{Quantity:3}634987 Пример штрихкода: «234002634987» – будет выбрано «2» чего-то. Пример шаблона: 234{Quantity:3.2}4987 Пример штрихкода: «234001864987» – будет выбрано «1,86» чего-то.
{SSCC:число}, число – количество символов в штрихкоде.	Позволяет привязывать к упаковкам какой-либо уникальный номер SSCC. При сканировании такого штрихкода, код SSCC будет заноситься в колонку «SSCC» новой строки документа, что позволит в учетной системе отслеживать операции над конкретными единицами товара. Пример шаблона: 7{ SSCC:5}1143576 Пример штрихкода: «7231111143576». При сканировании такого штрихкода в свойство SSCC (SelectedProduct.SSCC) будет занесено «23111».
{Date:формат}, где формат – строковой формат даты	Позволяет заносить в штрихкод упаковки дату ее регистрации (например, дату приемки). Пример шаблона: 2183{Date:ddMMyyyy} Описание правил задания формата даты можно найти по ссылке http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpguide/html/cpconstandarddatetimeformatstrings.asp. Пример штрихкода для типа упаковки «коробка»: «2183220620005». При сканировании такого штрихкода в свойство выбранного товара (SelectedProduct.RegisteredDate) будет занесено 22 июня 2005 года.
{ExpDate:формат}, где формат – строковой формат даты	Позволяет ограничивать срок годности товара указанной датой. Пример шаблона: 43{ExpDate:yyyyMMdd}21354 Описание правил задания формата даты можно найти по ссылке http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpguide/html/cpconstandarddatetimeformatstrings.asp. Пример штрихкода: «432007110821354». В поле ExpiredDate выбранного товара будет занесена дата 8 ноября 2007 г. Это позволит, например, реализовать в процессе проверку срока годности выбираемого товара.

Все описанные выше шаблоны могут применяться как по одиночке, так и в совокупности, например:

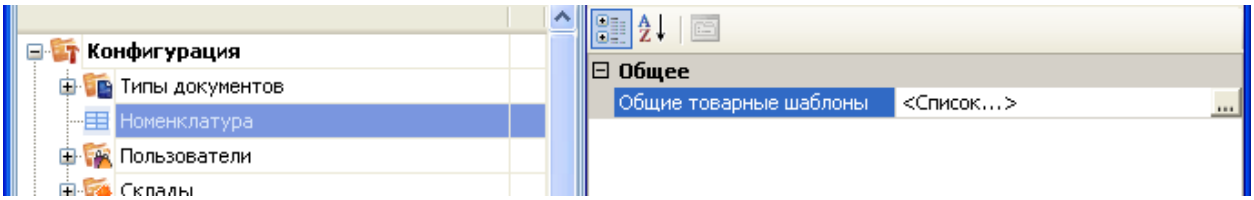
Шаблон	Пример
11{SKU:6}{ExpDate:ddMMyyyy}{Quantity:3}45	«1101022213022004011145» SKU – 010222 Срок годности – 13.02.2004 Количество – 11 единиц
544332{Quantity:2}{SKU:8}	«5443320105490321» SKU - 05490321 Количество – 1 единица

Шаблоны могут использоваться не только для распознавания уже существующих штрихкодов, введенных производителем, но и при создании собственных этикеток. При печати этикетки средствами системы, шаблон, соответствующий шаблону, генерируется автоматически из введенных пользователем данных.

Общие шаблоны штрихкода

Кроме шаблонов у упаковки можно задать общие шаблоны, действующие на весь справочник товара сразу.

Такие шаблоны можно задать прямо через панель управления, в узле конфигурации «Номенклатура»:



Правила задания общих шаблонов точно такие же, как и шаблонов в упаковках, кроме трех дополнительных полей в шаблоне:

Шаблон	Описание
{ProductId:число}, число – количество символов кода товара в штрихкоде ***обязательный элемент шаблона (можно заменить с помощью Barcode)***	Определяет идентификатор товара в штрихкоде, для поиска в справочнике номенклатуры. Является обязательным параметров в общем шаблоне. Пример: шаблон «{ProductId:5}{Quantity:3}{Date:ddMMyyyy}» При вводе штрихкода «1200502612042010» будет производиться поиск товара по коду «12005», и если он будет найден, то будет выбран товар с этим кодом, в количестве 26 базовых упаковок и проставлена RegistrationDate 12.04 2010 г.
{Barcode:число}, число – количество символов штрихкода товара в штрихкоде ***обязательный элемент шаблона (можно заменить с помощью ProductId)***	Определяет штрихкод товара, для поиска в справочнике номенклатуры. Является обязательным параметров в общем шаблоне. Пример: шаблон «{Barcode:13}{Quantity:3}» При вводе штрихкода «4607060501490005» будет производиться поиск товара по штрихкоду «4607060501490», и если он будет

	найден, то будет выбран товар с этим штрихкодом, в количестве 5 базовых упаковок.
{PackingId:число}, число – количество символов кода упаковки в штрихкоде ***необязательный параметр***	Позволяет дополнительно определить выемку кода упаковки из штрихкода. Не является обязательным параметром. Если он не задан, то найденный товар выбирается в базовом типе упаковки.

Использование общих шаблонов позволяет упростить задание однотипных шаблонированных штрихкодов, если они должны быть определены для всего товара.

Политика учета товара

Кроме простого ввода количества (непосредственно в том типе упаковки, который был отсканирован или выбран), система позволяет реализовать ввод по нескольким типам упаковки, например, одновременно вводить количество в коробках и штуках.

Для этих целей используется политика учета количества (Cleverence.Warehouse.QuantityPolicy).

Политика задает коллекцию идентификаторов упаковок товара, в которых будет производиться ввод количества. Кроме того, с помощью свойства QuantityPolicy.Multiline, вы можете задать, как будет производиться добавление записи в документ: в виде нескольких строк, по одной строке на каждый тип упаковки, заданной в политике; либо в виде одной строки, в которой количество будет пересчитано в базовый тип упаковки для товара.

Привязка политики к товару осуществляется с помощью свойства Product.QuantityPolicy.

В случае если планируется использовать локальный, загружаемый на терминал, справочник номенклатуры рекомендуется максимально минимизировать использование политик учета, с целью экономии памяти мобильного устройства. Для этого рекомендуется использовать одну политику учета для целой группы однотипных товаров.

Например, для всех товаров, количество которых должно вводиться в коробках и штуках, завести одну политику.

```
quantityPolicy = СоздатьОбъект("Cleverence.Warehouse.QuantityPolicy");
quantityPolicy.Multiline = 0;
quantityPolicy.PackingIds.Add("pid2");
quantityPolicy.PackingIds.Add("pid1");
```

Кроме такой политики, для всех товаров, с которыми она будет применяться, необходимо создать упаковки с идентификаторами pid1 и pid2, описывающие штуки и коробки соответственно.

```
packing      = СоздатьОбъект("Cleverence.Warehouse.Packing");
packing.Id   = "pid1";
packing.Name = "шт";
packing.SelfVolume = 0;
packing.SelfWeight = 0;
packing.UnitsQuantity = спрЕдиницы.Коэффициент;
product.Packings.Add(packing);

packing      = СоздатьОбъект("Cleverence.Warehouse.Packing");
packing.Id   = "pid2";
```

```
packing.Name = "кор";
packing.SelfVolume = 0;
packing.SelfWeight = 0;
packing.UnitsQuantity = спрЕдиницы.Коэффициент;
product.Packings.Add(packing);
```

Выгрузка номенклатуры

Для выгрузки заполненного справочника следует применять StorageConnector.SetProducts(ProductCollection products) - для нескольких товаров за один раз, либо, StorageConnector.SetProduct(Product product) - в случае выгрузки одного, нового или изменившегося, товара.

```
//создание коллекции продуктов
products = СоздатьОбъект("Cleverence.Warehouse.ProductCollection");

//создание продуктов
product = СоздатьОбъект("Cleverence.Warehouse.Product");
...
//добавление продукта в коллекцию для выгрузки
products.Add(product);

//выгрузка продуктов
connector.SetProducts(products);
```

Оба варианта функции дополняют существующий на сервере справочник номенклатуры. При этом, если на сервере уже есть товар с идентификатором, совпадающим с выгружаемым, то товар на сервере заменяется новой версией. Для полной очистки справочника или удаления одного товара на сервере необходимо использовать функции StorageConnector.RemoveProducts() и StorageConnector.RemoveProduct(string productId).

Альтернативная выгрузка номенклатуры для «1С:Предприятие»

Кроме основных методов выгрузки номенклатуры в объекте StorageConnector, для продуктов компании 1С разработаны также методы и обработки для выгрузки номенклатуры, соответствующие стандарту типовых конфигураций 1С для работы с ТСД.

Уже готовые обработки для различных конфигураций и платформ Вы можете найти в стандартной поставке Mobile SMARTS в папке «C:\Program Files\Cleverence Soft\Mobile SMARTS 2008\Demo\Конфигурация и обработки от драйвера 1С\Типовые обработки драйвера».

Использование по мере возможности этих обработок, или их модификаций под нетиповые конфигурации можно считать даже более предпочтительным вариантом, чем разработка своих выгрузок с нуля.

§ 5. Выгрузка, загрузка и удаление документов

Выгрузка документов на Сервер осуществляется с помощью функции StorageConnector.SetDocuments. Для загрузки, соответственно, применяется функция StorageConnector.GetDocuments.

Выгрузка документов

Каждая строка декларативной и текущей частей документа представляет собой объект типа Cleverence.Warehouse.DocumentItem и состоит из следующих фиксированных позиций.

ProductId	Идентификаторы продукта и упаковки. Указывают для строки номенклатуру, и упаковку, в которой измеряется товар.
PackingId	
DeclaredQuantity	Количественные характеристики товара. Исчисляются в указанной для строки упаковке. DeclaredQuantity отвечает за хранение заявляемого количества товара. Если документ создается на терминале, заявленное количество товара будет равно нулю. CurrentQuantity содержит текущее, набранное пользователем количество товара.
CurrentQuantity	
FirstCellId	Идентификаторы позиции хранения, к которой привязана строка документа. В этом качестве могут выступать ячейка или паллета, в зависимости от конкретного складского процесса.
SecondCellId	
RegisteredDate	Временные характеристики для товара в строке. Могут применяться, например, для сбора данных о дате занесения товара на склад его сроке годности. Заполнение дат может происходить либо из штрихкода по шаблону, либо ручным выбором.
ExpiredDate	
SSCC	Уникальный номер единицы хранения. Служит для идентификации конкретных штучных экземпляров товара. Заполняется из штрихкода товара по шаблону при выполнении операции.

Рассмотрим пример заполнения документа перед выгрузкой. В примере показано заполнение документа Mobile SMARTS на основе документа 1С (парДокумент).

```
//создание объекта документа и заполнение его шапочной части
document = СоздатьОбъект("Cleverence.Warehouse.Document");
document.Id = СформироватьИдентификатор(парДокумент, парТипДокумента);
document.Name = Строка(парДокумент);
document.Appointment = имяПользов;
document.CreateDate = парДокумент.ДатаДок;
document.DocumentTypeName = парТипДокумента;
document.InProcess = 0;
document.Finished  = 0;
document.Modified  = 0;
document.WarehouseId = 1;
document.Description = "приемка товара";
document.Priority = парДокумент.Приоритет;
document.Barcode = глШтрихКод("ЗаявкаНаПоставку", парДокумент.НомерДок);
document.DistributeByBarcode = 0;
```

При заполнении шапки документа, многие значения свойств переносятся из соответствующих значений документа учетной системы. Для формирования уникального идентификатора Document.Id написана отдельная функция СформироватьИдентификатор. Следует тщательно подойти к

формированию идентификаторов, так как почти всегда необходимо иметь и обратную связь: находить по идентификатору документа Mobile SMARTS документ-прототип в учетной системе. Это почти всегда требуется при обратной загрузке обработанного документа с систему.

```
//заполнение табличной части
парДокумент.ВыбратьСтроки();
Пока парДокумент.ПолучитьСтроку() <> 0 Цикл
      лКоличество = парДокумент.Количество * парДокумент.Единица.Коэффициент;
      documentItem = СоздатьОбъект("Cleverence.Warehouse.DocumentItem");
      documentItem.ProductId = парДокумент.Товар.Код;
      documentItem.PackingId = "pid1";
      documentItem.CurrentQuantity = 0;
      documentItem.DeclaredQuantity = лКоличество;
      document.DeclaredItems.Add(documentItem);
КонецЦикла;
```

После инициализации шапки документа, необходимо заполнить его декларативную часть. Так как в данном случае рассматривается документ приемки, то нам достаточно для каждой строки учетной системы создать эквивалентную строку в документе Mobile SMARTS. Проходя в цикле по строкам документа-прототипа, мы создаем новые строки Cleverence.Warehouse.DocumentItem, и заполняем их соответствующими значениями.

```
//создание коллекции документов для выгрузки
outDocuments = СоздатьОбъект("Cleverence.Warehouse.DocumentCollection");

//процедура создания документа
...

//добавление документа в коллекцию выгружаемых документов
outDocuments.Add(document);

//выгрузка документов
connector.SetDocuments(outDocuments);
```

Загрузка документов

```
//получение завершенных документов типа приемка с сервера
inDocuments = connector.GetDocument("accept", 1);

//получения количества выбранных документов
колВо = inDocuments.Count;
```

При загрузке документов не требуется заранее создавать объект коллекции. Он создается компонентой доступа автоматически. Кроме того, функция загрузки позволяет получить только выполненные документы, которые уже готовы к переносу назад, в учетную систему. Для этой цели применяется второй параметр функции загрузки (первый – имя типа документов). Если его значение ложное (в учетных системах, не поддерживающих булевый тип данных, обычно используется false = 0, true = 1), то Сервер вернет все выгруженные документы указанного типа.

В процедурах выгрузки и загрузки большую роль играют свойства Document.InProcess и Document.Finished. Первое свойство показывает, захвачен ли уже документ на мобильный терминал, или еще нет. Следует избегать удаления или повторной выгрузки уже захваченных документов, так как это может нарушить корректную работу системы. Второе свойство становится положительным после обработки документа на терминале и возврате его назад, на Сервер. Именно это свойство служит сигналом к тому, что документ можно переносить в учетную систему.

Иногда необходимо удалить документы, расположенные на Сервере. Для этой цели используются функция StorageConnector.RemoveDocuments.

```
//получаем все документы типа accept
serverDocuments = connector.GetDocuments("accept", 0);

//удаляем их
connector.RemoveDocuments(serverDocuments);
```

К удалению документов (особенно уже обработанных или захваченных на мобильный терминал) следует относиться крайне осторожно, во избежание утери ценных данных.

§ 6. Штрихкоды контейнеров и паллеты

Система поддерживает использование паллет в качестве позиции хранения товара.

Паллеты характеризуется своими штрихкодами, которые могут быть распечатаны как с помощью системы печати Mobile SMARTS, так и сторонними программами.

Для экономии ресурсов мобильного терминала, система, вместо выгрузки тысяч паллет, позволяет применять шаблоны штрихкодов, аналогичные шаблонам, применяемым в товарах. В качестве ключевых слов в шаблоне можно применять любую буквенную последовательность, например, “Main”, “First”, “Second”.

К примеру, добавление паллеты с Barcode = “23{Main:}” определяет паллеты со штрихкодами, начинающимися на «23»; а с Barcode = “5{Main:3}5” – задает паллеты со штрихкодами, начинающимися и заканчивающимися на «5», с центральной частью из 3х символов.

Сохранение справочника паллет на Сервере производится с помощью функции StorageConnector.SetPallets(PalletsBook palletsBook).

```
Процедура ВыгрузитьПаллеты()
	palletsBook = СоздатьОбъект("Cleverence.Warehouse.PalletsBook");
//создание шаблонированной паллеты
	pallet = СоздатьОбъект("Cleverence.Warehouse.Pallet");
	pallet.Barcode = "98{Main:}";
	palletsBook.Pallets.Add(pallet);

//создание обычной паллеты
	pallet = СоздатьОбъект("Cleverence.Warehouse.Pallet");
	pallet.Barcode = "34526";
	palletsBook.Pallets.Add(pallet);

	connector.SetPallets(palletsBook);
КонецПроцедуры
```

Приведенная функция выгружает объект, описывающий все паллеты, штрихкод которых начинается «98», и паллету с фиксированным штрихкодом «34526».

§ 7. Признаки

Признаки (Cleverence.Warehouse.Classificator) – это средство для привязки дополнительных свойств для ячеек, паллет и товаров.

Применение признаков имеет смысл в случаях, когда при обработке документа требуется указать дополнительную информацию об одном из указанных выше объектов, например, выбрать тип принятой паллеты: обычная, бракованная и т.д., или указать статус ячейки: пустая, с товаром, заполнена.

Работа признаков подразумевает выбор атрибута объекта из фиксированного набора значений.

Для внесения в систему признаков существует специальный справочник Cleverence.Warehouse.ClassificatorsBook. Он содержит коллекцию самих признаков ClassificatorsBook.Classificators, а также коллекцию их типов ClassificatorsBook.Types.

Тип признака Cleverence.Warehouse.ClassificatorType предназначен для группировки нескольких признаков в единую группу выбора. Кроме того, с помощью свойства ClassificatorType.Exclusive, он позволяет указать, могут ли несколько признаков одного типа быть привязаны к одному объекту.

Признак Cleverence.Warehouse.Classificator, кроме уникального идентификатора, содержит свое имя, штрихкод и идентификатор своего типа.

Приведенный ниже пример позволит нам программно организовать на мобильном клиенте выбор цвета паллеты из двух вариантов: красный и зеленый.

```
//создание справочника признаков
classifBook = СоздатьОбъект("Cleverence.Warehouse.ClassificatorsBook");

//создание и заполнение свойств типа признака
classType = СоздатьОбъект("Cleverence.Warehouse.ClassificatorType");
classType.Id="ЦветПаллеты";

//к одной паллете может быть назначен только один цвет
classType.Exclusive = 1;

//занесение типа в справочник
classifBook.Types.Add(classType);

//создание признака «красный»
classificator = СоздатьОбъект("Cleverence.Warehouse.Classificator");
classificator.TypeId = "ЦветПаллеты";
classificator.Barcode = "";
classificator.Id = "1";
classificator.Name = "красная";
classifBook.Classificators.Add(classificator);

//создание признака «зеленый»
classificator = СоздатьОбъект("Cleverence.Warehouse.Classificator");
```

```
classificator.TypeId = "ЦветПаллеты";
classificator.Barcode = "";
classificator.Id = "1";
classificator.Name = "зеленая";
classifBook.Classificators.Add(classificator);

//выгрузка справочника признаков
connector.SetClassificators(classifBook);
```

Для выгрузки заполненного справочника на Сервер используется команда StorageConnector.SetClassificators.

§ 8. Принтеры и печать этикеток через сервер Mobile SMARTS

Mobile SMARTS позволяет использовать произвольное количество принтеров и поддерживает печать этикеток, как из учетной системы, так и из мобильного клиента.

Информация о принтерах и схема их применения содержится в справочнике Cleverence.Warehouse.PrintersBook.

Каждый принтер (Cleverence.Warehouse.Printer), кроме уникального идентификатора, описывает имя и строку соединения (Printer.ConnectionString). Система может работать с любыми принтерами, поддерживающими печать в Windows (посредством windows driver). В качестве ConnectionString может использоваться либо имя принтера, например «EasyCoder PD4 (203 dpi)», либо строка соединения с принтером вида «\\server\printer» (для удаленного принтера).

Для выгрузки справочника принтеров применяется функция StorageConnector.SetPrinters.

```
//создание объекта принтера и заполнение его свойств
printer = СоздатьОбъект("Cleverence.Warehouse.Printer");
printer.Id = "commonPrinter";
printer.PrinterName = "Общий принтер";
printer.ConnectionString = "EasyCoder PD4 (203 dpi)";

//добавление принтера в справочник
printersBook.Printers.Add(printer);

//выгрузка справочника
connector.SetPrinters(printersBook);
```

Печать в системе производится через Сервер, поэтому доступ к принтерам должен быть настроен на машине, где было установлено серверное приложение.

Шаблоны этикеток

Для задания внешнего вида выводимых данных, в системе применяются шаблоны этикеток.

Все шаблоны этикеток имеют уникальное имя и хранятся на сервере. Их получение и сохранение осуществляются функциями StorageConnector.GetLabelTemplate и StorageConnector.SetLabelTemplate соответственно.

Для создания шаблонов используется специализированный редактор, вызов которого может быть реализован непосредственно из учетной системы (функция StorageConnector.EditTemplate). Этот же редактор встроен в панель управления.

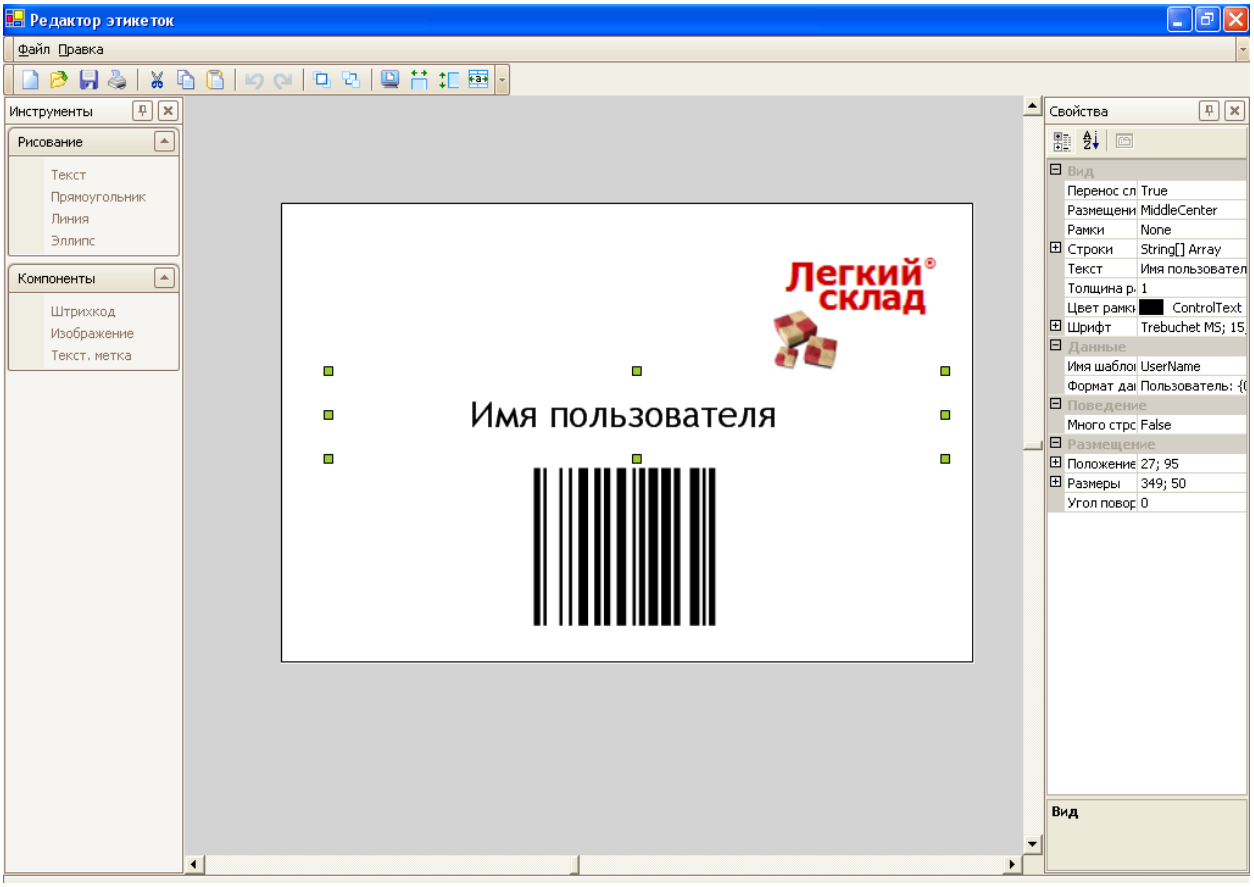
```
//загрузка шаблона этикетки
шаблонЭтикетки = connector.GetLabelTemplate("mainLabel");

Если ПустоеЗначение(шаблонЭтикетки.Name) = 1 Тогда
шаблонЭтикетки.Name = "mainLabel";
КонецЕсли;

//редактирование шаблона
модифШаблон = connector.EditTemplate(шаблонЭтикетки);

//сохранение шаблона
Если модифШаблон.TemplateCode <> шаблонЭтикетки.TemplateCode Тогда
      connector.SetLabelTemplate(модифШаблон);
КонецЕсли;
```

Редактор позволяет размещать на поверхности этикетки текстовые, штрихкодовые и графические элементы, а также задавать привязку этих элементов к данным при помощи свойства «Отображаемые данные». Например, можно разместить на шаблоне этикетки элемент «Текст» и в отображаемых данных указать «название товара», – это будет означать, что у шаблона этикетки появился аргумент «название товара», значение которого следует передавать при печати конкретной этикетки. Т.е. при печати текст в элементе будет заменен на то, что будет передано в качестве значения аргумента «название товара».



Печать из учетной системы

Для печати непосредственно из учетной системы применяется функция StorageConnector.PrintLabel. В качестве параметров эта функция использует уникальный идентификатор принтера Printer.Id и объект этикетки Cleverence.Warehouse.Label.

Для вывода в шаблон этикетки пользовательской информации, необходимо с помощью функции Label.Add, определить значение переменных, заданных при создании этикетки в редакторе.

```
//создание этикетки для печати
label = СоздатьОбъект("Cleverence.Warehouse.Label");

//инициализация переменных
label.Add("UserName", user.Name);
label.Add("PrintDate", РабочаяДата());
label.TemplateName = "userLabel";

//печать этикетки
connector.PrintLabel("commonPrinter",label);
```

Печать через сервер из мобильного клиента

Для печати этикеток с ТСД используется специальное действие «Печать этикетки». При этом не указывается, на какой принтер будет происходить печать, т.к. выбор принтера – отдельный процесс, выполняемый сервером терминалов Mobile SMARTS. Для выбора принтера на котором будет происходить печать, используется коллекция привязок принтеров (ProductsBook.PrinterMappings),

редактируемая в панели управления. Каждая такая привязка задает принтер для сочетания склада, типа документа и пользователя (можно указать только часть ограничений).

При печати сервер ищет наиболее полное совпадение из имеющихся и выбирает соответствующий ему принтер. Если ни одна привязка не подходит, выдается ошибка печати. В случае если Вы не описали привязками абсолютно все возможные сочетания параметров, то система будет искать наиболее подходящий. Например, если каждый ваш склад использует по одному принтеру, то вполне достаточным будет описывать соответствие только идентификатора склада идентификатору принтера, не задавая конкретное имя пользователя и тип документа.

Для печати на один единственный принтер достаточно добавить под него одну привязку без каких-либо ограничений. Соответственно, добавив к существующим одну привязку только с параметром PrinterId, без указания склада и т.п., вы как бы заведете принтер по умолчанию, печать на который будет производиться в тех случаях, когда более подходящий принтер найден не был.

Свойство «Параметры» в действии печати этикетки задают перечень данных, которые попадут в шаблон этикетки. Для этого списка нет стандартных обязательных значений – имена параметрам определяются нарисованным шаблоном этикетки. У элементов шаблона этикетки в дизайнере есть свойство «Отображать данные» - это именно то имя, которое следует использовать как имя параметра в действии печати этикетки. Имена произвольные.

§ 9. Беспроводная печать с помощью Mobile SMARTS

С 15 декабря 2009 г. в Mobile SMARTS доступна печать на беспроводные принтеры этикеток. В этой статье я хочу подробно рассказать о том, как ей пользоваться. Статья отражает функционал Mobile SMARTS на момент 25 декабря 2009 г., время от времени она будет обновляться, а указанная дата меняться.

Примечание: Беспроводная печать возможна только из версии клиента Mobile SMARTS под .NET Compact Framework 3.5. При установке клиента на терминал сбора данных в профиле терминала видно, какая версия Framework используется. .NET Compact Framework 3.5 может быть установлен только на операционные системы Windows CE 5.0, Windows Pocket 5.0 и старше.

Обычная печать на обычные принтеры в Mobile SMARTS проходит через посредника в виде специального сервера печати Mobile SMARTS. Т.е. обычно терминал сбора данных сам этикетку не формирует и никуда её не отсылает, он всего лишь передает своему серверу терминалов название этикетки и список значений, которые нужно подставить в этикетку при печати.

Сервер терминалов, в свою очередь, передает все эти данные серверу печати вместе с исходным кодом шаблона этикетки. И уже сервер печати подставляет в шаблон данные, а затем превращает этикетку в картинку и печатает эту картинку на указанный принтер. Т.е. это сервер печати рисует все тексты и штрихкоды. Для этикеточных принтеров картинку можно заменить на текстовую этикетку на языке принтера, и тогда уже сам принтер будет рисовать тексты и штрихкоды в соответствии с переданными командами.

Это что касается обычной печати. Для беспроводной печати клиент Mobile SMARTS умеет самостоятельно обрабатывать этикетки, написанные на языке команд печати конкретного используемого принтера. Это означает, что каждую этикетку для беспроводной печати придется создавать с использованием языка команд принтера. По крайней мере, пока это так.

Для поддержки беспроводной печати в Mobile SMARTS внесены следующие изменения:

1. Добавлено новое действие «Выбор принтера», которое позволяет искать и выбирать Bluetooth-принтеры;

2. В действие «Печать этикетки» добавлено свойство «Принтер». В нем можно указать путь к беспроводному принтеру или путь к переменной, в которой лежит такой путь или объект принтера , выбранный действием «Выбор принтера». Если свойство не заполнено, то печать выполняется в обычном режиме через сервер.

Конкретные шаги по каждому из приведенных этапов будут рассмотрены ниже в статье.

Создание шаблона этикетки для печати

У каждого производителя принтера есть свой набор любимых языков печати, которыми управляются их принтеры. В данной статье мы печатаем на принтер Zebra RW420, поэтому любимым языком будет CPCL. Язык CPCL поддерживается всеми RW, QL и MZ моделями Zebra. Скачать документацию по языку, а также по самому принтеру можно после регистрации на сайте Zebra.com

Не всегда обязательно изучать язык команд принтера для того, чтобы создать хорошую этикетку. По счастью, с каждым этикеточным принтером идет диск, на котором присутствует та или иная программа для визуального создания этикеток. Часть работы можно проделать в ней, однако для параметризации этикетки нужными переменными всё-таки придется вручную редактировать текст.

Например, для создания этикеток на CPCL компанией Zebra Technologies предоставляется программа Label Vista.

Используя Label Vista можно нарисовать следующую этикетку:

```
! 0 200 200 240 1
LABEL
CONTRAST 0
TONE 0
SPEED 5
PAGE-WIDTH 390
T 7 0 11 62 цена
T 4 3 99 49 100.00
BT 7 0 5
B EAN13 2 2 60 45 147
1234567890123
BT OFF
T 0 3 8 9 Название товара
FORM
PRINT
```

При этом Label Vista предупредит, что этикетка содержит непечатаемые символы, а результат печати будет такой:

При этом Label View предупредит, что этикетка содержит непечатаемые символы, а результат печати будет такой:

Т.е. русский текст не отобразился.

Печать русских текстов и шрифтов на принтер Zebra

Русские шрифты для конкретного купленного принтера не всегда существуют. А существующие могут не подходить по размеру. И новые взять негде. Например, в Label Vista существует возможность конвертирования обычных Windows-шрифтов (TrueType) в шрифты принтера (Scalable Fonts). Однако

конвертируются только первые 128 символов из полного набора, и русские буквы туда никогда не попадают.

Для вывода русских текстов на CPCL в Mobile SMARTS добавлена возможность вставки в этикету кода, который преобразует текст в картинку. Для этого необходимо открыть этикетку в простом текстовом редакторе и написать следующее:

```
{global::Zebra.CPCL.Text([<название шрифта>, <высота шрифта в миллиметрах>,]
[<x>,  <y>,] <текст>, <доступная ширина>)}
```

Координаты <x>, <y> задаются в единицах, которые в данный момент выбраны текущими (см. руководство по CPCL).

Метод global::Zebra.CPCL.Text возвращает команду UG и соответствующую картинку. Из шрифтов доступны «Tahoma» и «Courier New».

Например:

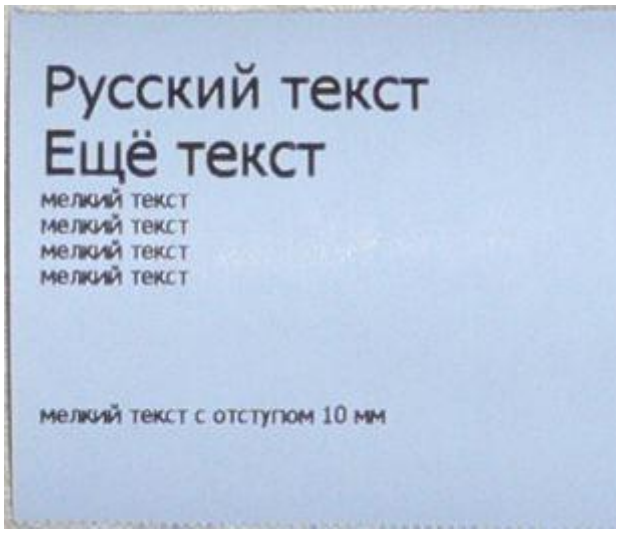
```
{global::Zebra.CPCL.Text("Tahoma", 3, 0, 26, "Русский текст", 360)}
{global::Zebra.CPCL.Text("Tahoma", 3, "Русский текст", 360)}
```

Также можно указывать:

```
! 0 200 200 290 1
PAGE-WIDTH 390
; SetFont(<название шрифта>, <высота в шрифта миллиметрах>).
{global::Zebra.CPCL.SetFont("Tahoma", 4)}

{global::Zebra.CPCL.Text("Русский текст", 360)}
{global::Zebra.CPCL.Text("Ещё текст", 360)}
{global::Zebra.CPCL.Text("Tahoma", 1.7, "мелкий текст", 360)}
{global::Zebra.CPCL.Text("Tahoma", 1.7, "мелкий текст", 360)}
{global::Zebra.CPCL.Text("Tahoma", 1.7, "мелкий текст", 360)}
{global::Zebra.CPCL.Text("Tahoma", 1.7, "мелкий текст", 360)}

; SetLP(<отступ до> [, отступ после]) в миллиметрах.
{global::Zebra.CPCL.SetLP(10, 0)}
{global::Zebra.CPCL.Text("Tahoma", 1.7, "мелкий текст с отступом 10 мм",
360)}
POSTFEED 20
PRINT
```



Т.е. координаты текста можно пропускать. Параметры, для которых указанно «в миллиметрах», работают правильно только при разрешении 200dpi, т.е. только если этикетка начинается на «! X 200 200 ..»

Важное замечание: код global::Zebra не является расширением языка печати и совершенно ничего не знает о том, что мы там пишем на CPCL. Т.е. вызовы SetLP, выравнивания текста Left|Right/CenterAlign и прочие вещи никак не связаны с SETLP, LEFT/RIGHT/CENTER и прочим в языке CPCL, т.е. одно о другом ничего не знает и друг на друга они не влияют.

Вставка в этикетку переменных значений

К сожалению, этикетки, созданные визуально, приходится много переделывать вручную. Как известно, в Mobile SMARTS шаблоны задаются путями к данным в фигурных скобках, например «{ScannedBarcode}» или «{Item.Product.Barcode}». В Label Vista можно задавать такие тексты, однако для штрихкода в ней можно указать только те символы, которые этим штрихкодом реально поддерживаются. Например, для EAN13 можно задать только цифры, мало того - только 12 или 13 цифр.

Несмотря на то, что принтер Zebra RW420 поддерживает параметризацию этикеток с помощью так называемых файлов формата, в которых вместо данных нужно указывать «\\», строка «\\» тоже не поддерживается Label Vista в качестве данных штрихкода EAN13. Тем более нельзя указать что-нибудь вроде «{ScannedBarcode}». В настоящий момент полноценно реализовать автоматическое выравнивание, вставку данных и прочие вещи только с привлечением языка управления принтером.

Ручной правкой кода этикетки можно добиться следующего результата:

```
! 0 200 200 530 1
PAGE-WIDTH 390
{global::Zebra.CPCL.Text("Tahoma", 4, 0, 0, "Мой магазин", 360)}
{global::Zebra.CPCL.Text("Tahoma", 2, 0, 40, "Адрес моего магазина", 360)}
TEXT 7 0 0 62 (495) 555-444-333
{global::Zebra.CPCL.Text("Courier New", 4, 0, 100, SelectedProduct.Product.Name, 360)}
{global::Zebra.CPCL.Text("Tahoma", 3, 0, 260, "цена", 60)}
RIGHT 390
TEXT 4 4 70 260 {SelectedProduct.Packing.price:0.00}
BARCODE-TEXT 7 0 5
CENTER 0
BARCODE EAN13 2 2 60 0 430 {SelectedProduct.Packing.Barcode}
POSTFEED 20
PRINT
```

Печать списков и длинных чеков

Для перебора списков, например, для печати длинного чека, используется режим Line Print и куски кода CPCL в качестве формата вывода значения списка:

```
! U1 SETLP 5 2 46
! U1 PAGE-WIDTH 390
AURORA {Text1}
! U1 SETLP 7 0 24
{Address}
(401) 555-4CUT
! U1 SETLP 7 0 54
! U1 BARCODE-TEXT 7 0 5
{Document.CurrentItems:
! U1 B EAN13 1 2 30 10 0 {Item.Packing.Barcode}
 }
! U1 POSTFEED 40
```

В данном случае для вывода списка всех строк Document.CurrentItems в этикетке написано

```
{Document.CurrentItems:
<формат элемента>
 }
```

, а в качестве формата элемента указано «! U1 B EAN13 1 2 30 10 0 {Item.Packing.Barcode}». Шаблон «! U1 B EAN13 1 2 30 10 0 {Item.Packing.Barcode}» будет обработан для каждой из строк в Document.CurrentItems. В переменной «Item» будут по очереди попадать строки документа, шаблоны будут обработаны, результатом каждого будет строка, и все они по очереди попадут в этикетку.

Копирование шаблона этикетки на терминал сбора данных

Созданный шаблон этикетки кладется в папку «LabelTemplates» либо на сервере Mobile SMARTS (если терминалы сбора данных работают с сервером), либо напрямую в папку установки клиента Mobile SMARTS на терминале сбора данных (если терминалы работают без сервера). Именем шаблона будет являться имя файла, а расширение у файла должно быть «.lbl». Т.е. шаблон «Ценник» должен называться «Ценник.lbl».

Указание пути к мобильному принтеру

У каждого Bluetooth-принтера есть свой уникальный адрес, а у каждого Wi-Fi-принтера свои IP и публичный порт, которые можно заполучить через меню при помощи кнопок на лицевой панели. Еще большее количество настроек можно увидеть на информационной распечатке принтера. Практически каждый этикеточный принтер на планете печатает информацию о себе с номерами прошивки и прочим, если его выключить, зажать кнопку «Feed», а затем, удерживая её, включить принтер и дождаться начала печати (2-3 секунды).

Печать на Bluetooth-принтер

В разных ситуациях могут понадобиться различные сценарии выбора принтера. В одном случае удобно давать пользователю возможность выбора принтера из нескольких, например, устроить поиск по

находящимся вокруг Bluetooth-устройствам. В другом желательно явно указать какие пользователи на какие принтеры будут печатать.

Mobile SMARTS позволяет реализовать все такие сценарии. Действие «Выбор принтера» позволяет искать Bluetooth-устройства, а действие «Печать этикетки» позволяет задать, откуда будет взята информация о том, куда печатать.

Самый простой способ тестовой печати – использовать действие выбора принтера, указать в нем, что выбранный принтер следует сложить в переменную «{SelectedPrinter}», а затем в действии печати указать, что принтер следует брать из «{SelectedPrinter}». Чтобы не искать принтер по 100 раз при последовательной печати, можно вставить перед выбором принтера проверку, лежит ли уже что-нибудь в переменной («{SelectedPrinter} != null»), и если лежит, то переходить сразу к печати.

Другой вариант – указать в действии печати прямой путь к принтеру или путь к переменной, где лежит такой путь. Формат пути следующий:

«bluetooth://» или «bt://», затем необязательное «[имя]@» и адрес вида «ЦЦ:ЦЦ:..» или «ЦЦЦ..». Если имя не указано, то в качестве имени будет использоваться «unknown». Вот примеры того, как разными способами можно указать один и тот же Bluetooth-принтер:

```
bluetooth://XXRC07-40-5254@00:03:7A:17:E5:3A

bluetooth://00:03:7A:17:E5:3A

bluetooth://00:03:7A:17:E5:3A/

bluetooth://00037A17E53A/

bluetooth://03:7A:17:E5:3A

bt:// XXRC07-40-5254@00:03:7A:17:E5:3A

bt:// XXRC07-40-5254@00:00:00:03:7A:17:E5:3A

bt://MeinePrintere@03:7A:17:E5:3A

bt:// XXRC07-40-5254@00:03:7A:17:E5:3A/

bt://03:7A:17:E5:3A

bt://037A17E53A
```

Печать на Wi-Fi-принтер

Wi-Fi-принтеры, а также простые сетевые и публичные принтеры доступны по протоколу TCP/IP, т.е. печатать на них можно, зная IP и номер порта. Wi-Fi и другие сетевые принтеры сами хранят свой IP и имеют прямой доступ в сеть. Кроме того, любой самый обычный офисный принтер можно вывесить в сеть по IP и порту, если завести виртуальный TCP-порт на компьютере, к которому он подключен.

Формат пути следующий:

«tcp://», необязательное «[имя]@», а затем IP и номер порта «в виде «XXX.XXX.XXX.XXX:<номер порта>». Если имя не указано, то в качестве имени будет использоваться «unknown». Вот примеры того, как разными способами можно указать один и тот же сетевой принтер:

tcp://Zebra-Z4M@10.10.1.24:169

tcp://Zebra-Z4M@10.10.1.24:169/

tcp://10.10.1.24:169

Печать через шнур

Хотя эта статья и называется «беспроводная печать», пару слов стоит сказать о печати через кабель. Действие печати этикетки понимает пути вида «COM0:», «COM3:» или «LPT:», отправляя этикетку в серийный порт и через соответствующий кабель прямо на принтер.

Собираем всё вместе

В целом процесс печати происходит следующим образом:

1. Действие печати этикетки ищет шаблон этикетки по указанному имени. Если клиент Mobile SMARTS работает в онлайн режиме с сервером, то шаблон скачивается с сервера. Если шаблон скачался, он сохраняется в папке «LabelTemplates», а для данного шаблона запоминается, что до следующего обмена данными шаблон повторно не скачивается. Т.е. если шаблон этикетки на сервере обновился, а клиент по нему уже печатал, обновление шаблона на терминале произойдет после цикла обмена данными (автоматически или по нажатию кнопки «Обмен данными»). Если же шаблон скачать с сервера не удалось, то файл с соответствующим файлом ищется в папке «LabelTemplates». Для батч-режима работы или в отсутствии соединения с сервером шаблон сразу ищется в папке «LabelTemplates» на терминале. Если ничего из этого не вышло, на экран выдается сообщение об ошибке, объясняющее что конкретно не удалось;

2. Действие печати этикетки ищет данные для подключения к принтеру по указанной переменной или пути;

3. Действие печати этикетки обрабатывает текст шаблона и получает на выходе готовый к печати текст этикетки;

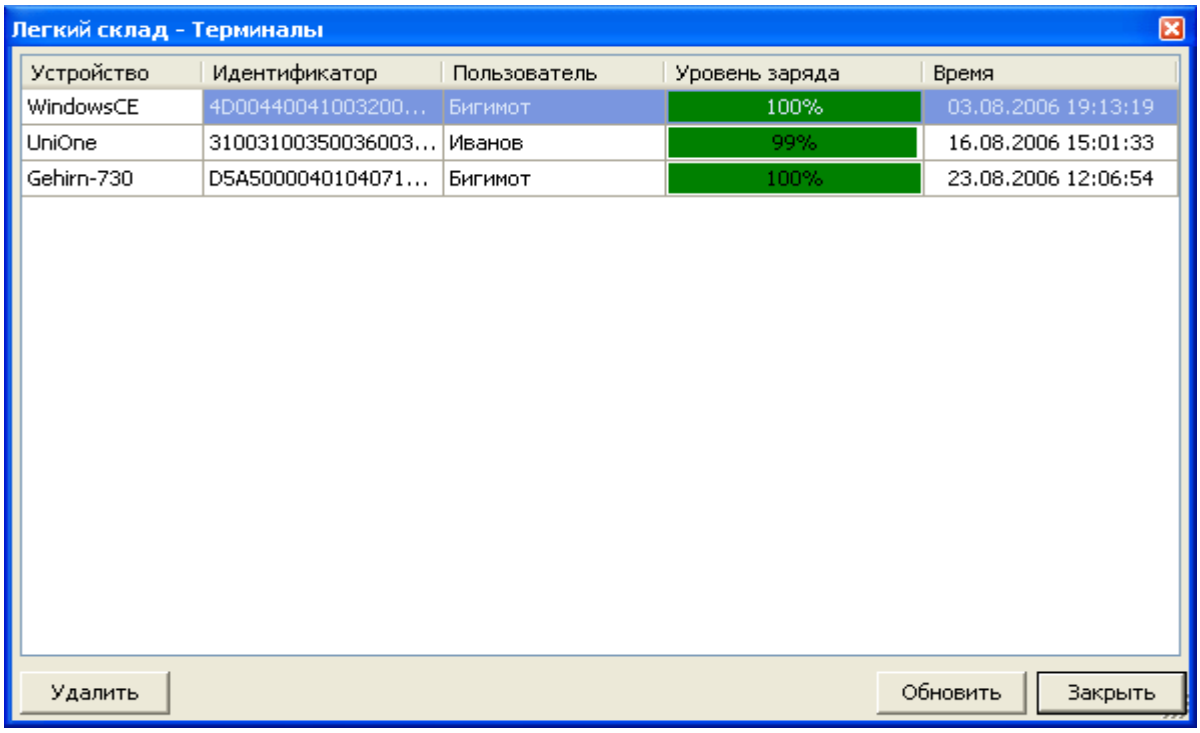
4. Действие печати этикетки устанавливает соединение с Bluetooth-принтером или сетевое соединение с Wi-Fi-принтером или открывает поток для случая печати через COM или LPT и отправляет этикетку на печать;

Если на каком-то из предыдущих этапов возникла ошибка, она записывается в лог ошибок на терминале, затем ошибка выдаётся на экран, после чего операция переходит к действию, указанному в свойстве «При ошибке печати».

Сервер терминалов, в свою очередь, передает все эти данные серверу печати вместе с исходным кодом шаблона этикетки. И уже сервер печати подставляет в шаблон данные, а затем превращает этикетку в картинку и печатает эту картинку на указанный принтер. Т.е. это сервер печати рисует все тексты и штрихкоды. Для этикеточных принтеров картинку можно заменить на текстовую этикетку на языке принтера, и тогда уже сам принтер будет рисовать тексты и штрихкоды в соответствии с переданными командами.

§ 10. Управление терминалами

Mobile SMARTS предоставляет возможность отслеживать состояние мобильных клиентов: текущий заряд батареи, имя работающего пользователя, последний сеанс связи и т.д.



Для вызова окна состояния устройств применяется функция StorageConnector.ShowDevicesInfo.

```
Процедура ПриОткрытии()
	connector = СоздатьОбъект("Cleverence.Warehouse.StorageConnector");
connector.InitializeServerConnection(СокрЛП(Константа.ПолучитьАтрибут("Строка Подключения")));
	connector.ShowDevicesInfo();
	Форма.Закрыть();
КонецПроцедуры
```

§ 11. Онлайн-вызов учетной системы с ТСД

В Mobile SMARTS существует специальный механизм обмена данными со внешними системами в режиме онлайн из программы мобильного терминала – так называемые *коннекторы*. При помощи коннекторов можно решить целый ряд весьма полезных задач, например:

- Получение реальных остатков по сканированной номенклатуре;
- Уточнение цен онлайн;
- Отправка набранных заказов с проверкой наличия и других условий;
- Подбор в ячейках, перемещение паллет и коробок с немедленной фиксацией изменений в учетной системе.

и т.п.

Для каждой учетной системы коннектор разрабатывается отдельно и особенности его работы зависят от реализации. Система Mobile SMARTS предъявляет к коннекторам только общие требования –

они явно сформулированы в программном интерфейсе, который должен реализовать любой из них (подробнее о разработке собственных коннекторов и о программном интерфейсе к ним см. «**Ошибка! Неверная ссылка закладки.**»).

Общие положения

Вызов внешних систем с терминала сбора данных (ТСД) происходит не напрямую, а через посредничество сервера Mobile SMARTS.

Данные для отправки во внешнюю систему формируются в клиенте Mobile SMARTS на ТСД и затем передаются серверу Mobile SMARTS с указанием, какой коннектор и для чего использовать.

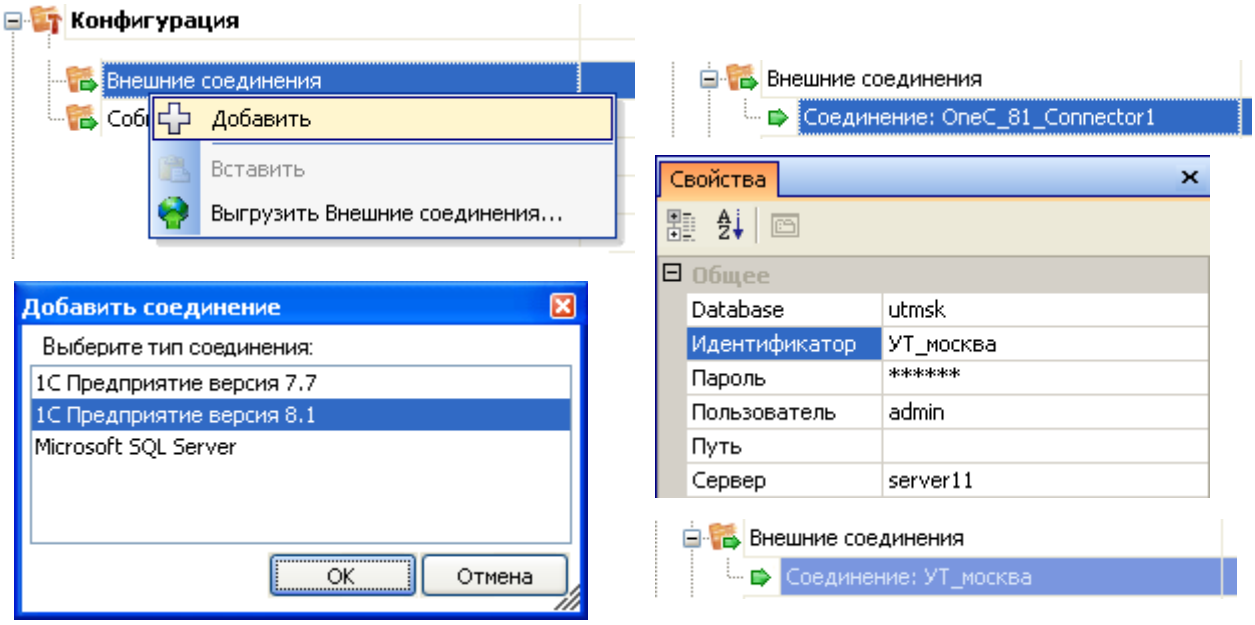
Коннектор указывается по имени, оно задается разработчиком или администратором при добавлении коннектора в общий список коннекторов в панели управления Mobile SMARTS. А «*для чего использовать*» указывается именами класса и метода во внешней системе. Что понимается под этими *классом* и *методом*, вообще говоря, определяется самим коннектором. Например, коннекторы к 1С позволяют вызывать экспортные функции и процедуры глобального контекста или контекста внешнего соединения, при этом класс не используется, а метод задает имя вызываемого метода или процедуры. Коннектор для Axapta использует и имя класса, и имя метода для вызова публичного статического метода указанного класса. Коннектор для Microsoft SQL Server не использует ни имени класса, ни имени метода, вместо этого он ожидает в передаваемых параметрах параметр query, который должен содержать SQL-запрос.

Конкретно обращение ко внешней системе с передачей и получением каких-то полезных данных происходит следующим образом:

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

Регистрация коннектора к внешней системе

Экземпляры коннекторов к внешним системам регистрируются в специальном узле дерева конфигурации в панели управления Mobile SMARTS:



Можно добавить несколько коннекторов с разными именами для подключения к нескольким базам данных одной и той же учетной системы. Свойство «Идентификатор» задает имя коннектора, которое требуется указать на ТСД при вызове внешней системы.

Файлы библиотек DLL, обеспечивающие работу стандартных коннекторов, по умолчанию не подключены к серверу и не загружаются им. При сохранении конфигурации на сервер может появиться ошибка вида «XmlSerializationException: Error serializing ServerEvents.xml. 'Connectors' cannot have …» – это означает, что добавленный коннектор зарегистрирован в панели управления, но соответствующие ему системные библиотеки (DLL) не найдены в папке bin на сервере Mobile SMARTS. Скорее всего они лежат в папке Connectors сервера, но не были скопированы в папку bin.

После добавления в список, копирования всех DLL и сохранения конфигурации на сервер Mobile SMARTS регистрация успешно завершена.

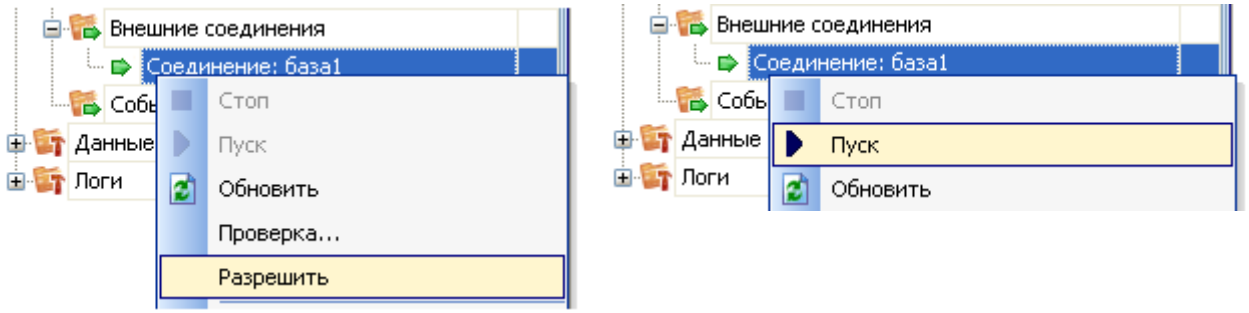
Запуск и остановка коннекторов

Все зарегистрированные коннекторы имеют два независимых друг от друга флажка «Разрешен/Запрещен» и «Остановлен/Запущен». Флаг «Разрешен/Запрещен» управляет возможностью обращения к коннектору с ТСД. Флаг «Остановлен/Запущен» определяет, загружены ли в память сервера компоненты подключения к внешней системе и установлено ли соединение.

Например, коннектор 1С 7.7 в режиме «Запущен» поднимает в память процесс «1cv7s.exe», который виден в менеджере задач операционной системы и обеспечивает работу OLE-компоненты доступа к 1С. Коннекторы для 1С 8 и Axapta работают внутри процесса сервера Mobile SMARTS «Cleverence.Warehouse.Server.exe» и выдают себя только увеличением занятой оперативной памяти.

Если коннектор разрешен, то он запускается при первом же обращении к нему (с ТСД или вручную). Поэтому если нужно загасить все подключения и гарантированно запретить запуск любых внешних подключений к системе, например для запуска 1С в монопольном режиме, следует не только остановить коннектор в панели управления, но и запретить его запуск (перевести в состояние «Запрещен»).

После регистрации коннектора следует проверить, выполняется ли подключение. Для этого следует разрешить доступ к коннектору, и затем запустить его:



В большинстве случаев попытка запуска завершится неудачей, т.к. у сервера Mobile SMARTS не будет хватать пользовательских прав на подключение ко внешней системе. По умолчанию сервер Mobile SMARTS запускается как служба Windows от имени Локальной системы (Local System). Под операционными системами Windows XP/2000/2003 этого достаточно для всех задач, кроме обращения к сетевым ресурсам (например, к базе данных SQL в сети или к сетевому принтеру). Под операционными системами Windows Vista и Windows 7 права Локальной системы ограничены до уровня обычного пользователя, ввиду чего многое будет запрещено.

В версиях Windows XP Home и Windows Vista Home работа с коннекторами на сервере Mobile SMARTS будет возможна только в том случае, если вызываемые внешние системы сами установлены на том же персональном компьютере, что и сервер Mobile SMARTS.

Для выхода из ситуации самым простым и самым неправильным будет дать серверу права администратора машины или домена – это следует максимально избегать. Самым правильным будет завести для сервера Mobile SMARTS отдельного доменного либо локального пользователя. Этому пользователю следует по очереди назначать требуемые права путем включения его в различные группы пользователей.

Для разрешения создавать OLE и COM объекты сетевых версий 1С или Axapta Business Connector созданного нового пользователя следует включить в группу «Пользователи DCOM» (Distributed COM Users). Для обращения к базам Microsoft SQL Server пользователя сервера Mobile SMARTS следует включить в группу «SQLServerUser$<MACHINE>» (или «SQLServerUser2005$<MACHINE>», «SQLServerUser2008$<MACHINE>»).

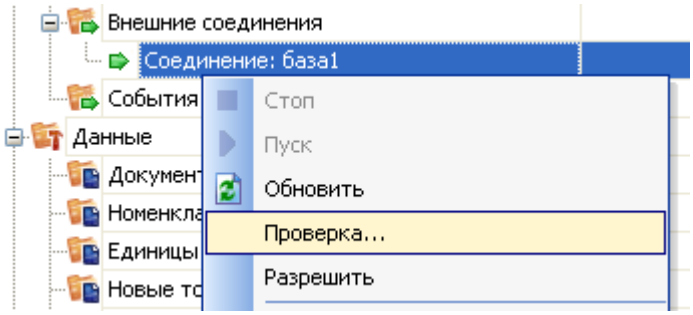
Другие ошибки при запуске коннекторов, такие как неверно заданные пути подключения, ошибки прав доступа к файлам и т.д., отображаются в диалоговых окнах с описанием на смеси английского и русского языков, при этом все возникшие проблемы с детальным описанием попадают в файл server_errors.log на сервере Mobile SMARTS.

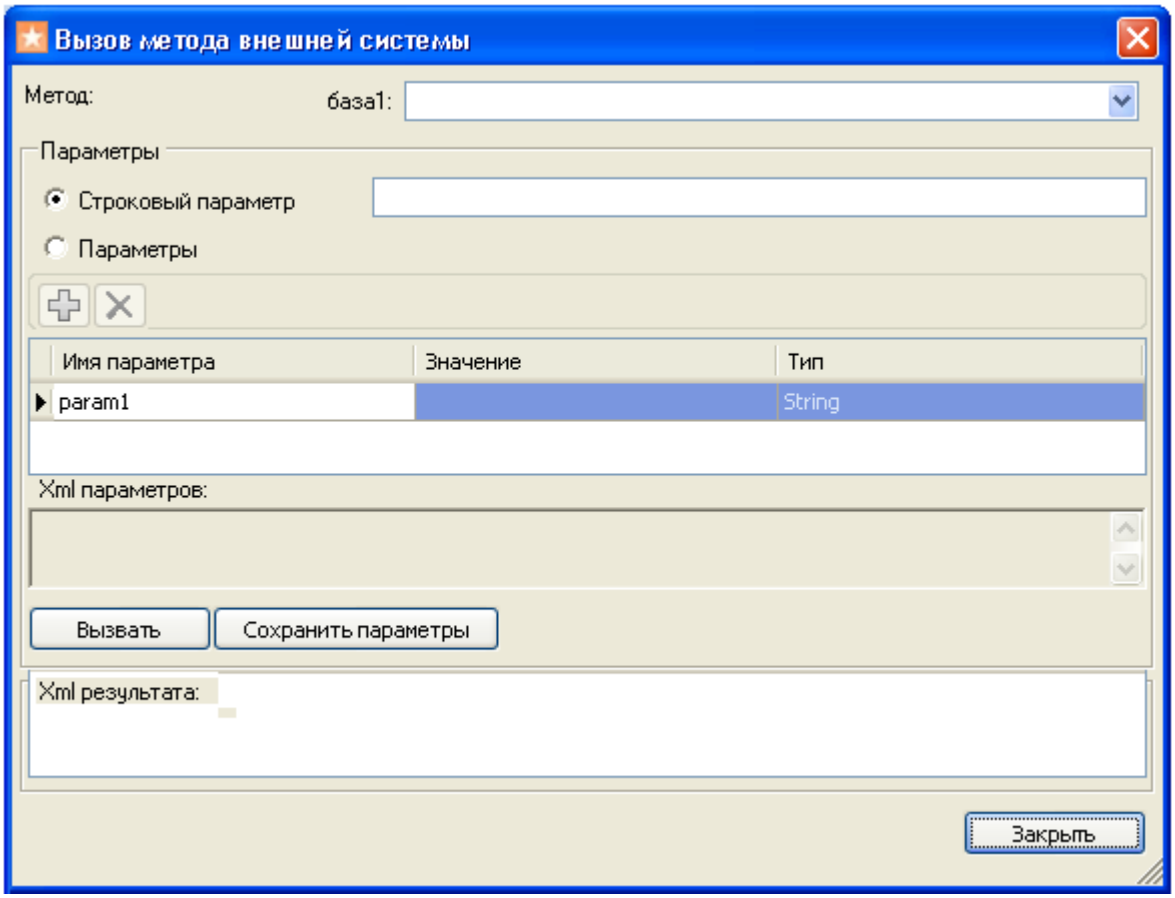
Примечание: Очень часто в решении проблемы помогает поиск в интернете по тексту, выдаваемому в качестве текста ошибки.

Поддержка коннекторов в самой внешней системе

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

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





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

Как видно, коннектору передаются *именованные параметры*. Метод внешней системы, который будет принимать эти параметры, может получить их в двух вариантах, в зависимости от реализации коннектора. Первый вариант – будет учитываться только порядок передачи параметров, а имена не имеют значения и отбрасываются. Второй вариант – метод принимает единственный строковой аргумент, в котором передается XML с сериализованным объектом Cleverence.Warehosue.InvokeArgs; это выглядит примерно как «<?xml version="1.0" encoding="utf-8" ?><InvokeArgs xmlns:clr="http://schemas...».

Программирование в 1С 7.7

Перед чтением этого раздела следует ознакомиться с содержанием предыдущего раздела «Поддержка коннекторов в самой внешней системе».

Предположим, что перед нами стоит задача поиска каких-либо данных по штрихкоду номенклатуры. Код на языке 1С, осуществляющий поиск данных, оформляется в виде экспортного метода в глобальном контексте. Если принимать аргументы обычным способом, то это будет выглядеть примерно так:

```
Процедура глПолучитьРозничныйОстатокТСД (Штрихкод) Экспорт
	Остаток = …
	…
	Возврат Остаток;
КонецПроцедуры // глПолучитьРозничныйОстатокТСД ()
```

В Mobile SMARTS существует некоторый аналог ТаблицыЗначений для передаваемых параметров, который называется InvokeArgs. Подробнее о свойствах и методах объектов, используемых при работе с Mobile SMARTS можно почитать в файле документации «Mobile SMARTS 2008 – Компонента доступа.chm».

Если принимать аргументы в виде InvokeArgs, то код будет выглядеть так:

```
Процедура глПолучитьРозничныйОстатокТСД (XML_Аргументов) Экспорт
	connector = СоздатьОбъект("Cleverence.Warehouse.StorageConnector");
	Аргументы = connector.InvokeArgsFromXml(XML_Аргументов);
	Штрихкод = Аргументы.ПолучитьСтроку("Штрихкод");
	Остаток = …
	…
	Возврат Остаток;
КонецПроцедуры // глПолучитьРозничныйОстатокТСД ()
```

Весь смысл использования более сложного способа передачи параметров состоит в том, чтобы уменьшить зависимость между кодом 1С и кодом Mobile SMARTS на мобильном терминале. При передаче одного только штрихкода пользы немного, но когда аргументов больше пяти – метод глПолучитьРозничныйОстатокТСД всё равно будет принимать один строковой параметр – и если при вызове метода на ТСД мы захотим передать лишний параметр или перепутаем порядок аргументов, то ничего страшного не случится. Главное не перепутать имена.

Возврат аргументов также возможен в сложном стиле. Это полезно при возвращении сразу нескольких значений. В Mobile SMARTS есть некоторый аналог ТаблицыЗначений для возвращаемых аргументов, называемый InvokeResult. Код возврата значений с использованием InvokeResult будет выглядеть примерно так:

```
	connector = СоздатьОбъект("Cleverence.Warehouse.StorageConnector");
	Результаты = СоздатьОбъект("Cleverence.Warehouse.InvokeResult");
	Результаты.Добавить("Остаток", Остаток);
	Результаты.Добавить("КодТовара", Номенклатура.Код);
```

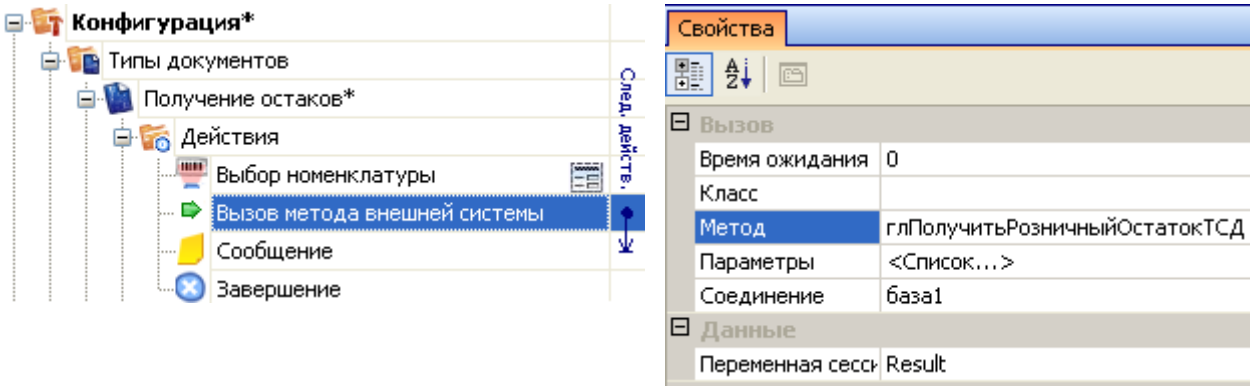
```
        Результаты.Добавить("НаименованиеТовара", Номенклатура.Наименование);
        …
        Возврат connector.СохранитьОбъектВ_XML(Результаты);
КонецПроцедуры // глПолучитьРозничныйОстатокТСД ()
```

Как видно из кода, метод возвращает строку, содержащую XML с результатами. Этот XML будет правильно проинтерпретирован сервером и клиентом Mobile SMARTS, в результате чего в сессии программы на терминале сбора данных окажутся переменные с названиями, переданными в метод «Добавить».

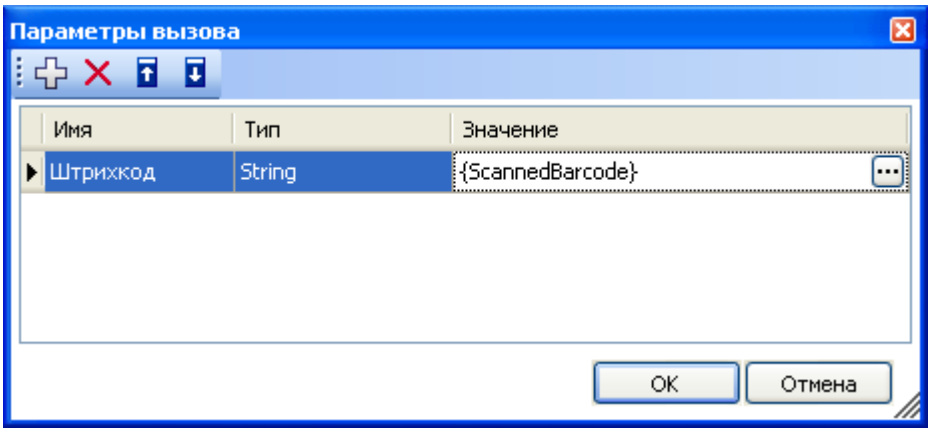
Для проверки работы кода 1С можно воспользоваться отладкой из панели управления Mobile SMARTS (см. предыдущий раздел этого же параграфа).

Чтобы вызвать новый метод 1С с терминала сбора данных, в программу обработки документа следует добавить действие «Вызов метода внешней системы». Для ознакомления с основами программирования в Mobile SMARTS следует смотреть раздел «Основы программирования в Mobile SMARTS».

Создадим новый виртуальный тип документа «Получение остатков» и протестируем в нём интересующий нас функционал:



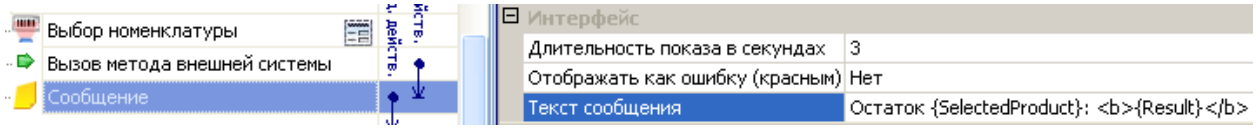
В свойстве «Параметры» действия «Вызов метода внешней системы» указываем именованные параметры для передачи, а именно «Штрихкод» как значение переменной «ScannedBarcode», в которую действие «Выбор номенклатуры» положило строку со считанным штрихкодом:



Примечание: Действие номенклатуры позволяет не только сканировать товары, но и выбирать их из списка при вводе спецштрихкода «0». Чтобы при выборе из списка передать в 1С не «0», а штрихкод выбранного товара, вместо «ScannedBarcode» следует указать «{SelectedProduct.Packing.Barcode}».

В свойстве «Переменная сессии для результата» действия «Вызов метода внешней системы» указано «Result». Это означает, что результат вызова метода 1С попадет в переменную и именем «Result».

Для отображения полученных остатков на экране терминала, укажем в действии «Сообщение» следующий шаблон отображения информации:



Если результаты вызова передаются в виде InvokeResult, то шаблон отображения может выглядеть следующим образом: «Остаток {Result.НаименованиеТовара}: {Result.Остаток}».

Программирование в 1С 8

Перед чтением этого раздела следует ознакомиться с содержанием предыдущего раздела «Поддержка коннекторов в самой внешней системе».

Предположим, что перед нами стоит задача поиска каких-либо данных по штрихкоду номенклатуры. Код на языке 1С, осуществляющий поиск данных, оформляется в виде экспортного метода в глобальном контексте. Если принимать аргументы обычным способом, то это будет выглядеть примерно так:

```
Процедура глПолучитьРозничныйОстатокТСД (Штрихкод) Экспорт
	Остаток = …
	…
	Возврат Остаток;
КонецПроцедуры // глПолучитьРозничныйОстатокТСД ()
```

В Mobile SMARTS существует некоторый аналог ТаблицыЗначений для передаваемых параметров, который называется InvokeArgs. Подробнее о свойствах и методах объектов, используемых при работе с Mobile SMARTS можно почитать в файле документации «Mobile SMARTS 2008 – Компонента доступа.chm».

Если принимать аргументы в виде InvokeArgs, то код будет выглядеть так:

```
Процедура глПолучитьРозничныйОстатокТСД (XML_Аргументов) Экспорт
	connector = новый COMОбъект("Cleverence.Warehouse.StorageConnector");
	Аргументы = connector.InvokeArgsFromXml(XML_Аргументов);
	Штрихкод = Аргументы.ПолучитьСтроку("Штрихкод");
	Остаток = …
	…
	Возврат Остаток;
КонецПроцедуры // глПолучитьРозничныйОстатокТСД ()
```

Весь смысл использования более сложного способа передачи параметров состоит в том, чтобы уменьшить зависимость между кодом 1С и кодом Mobile SMARTS на мобильном терминале. При передаче одного только штрихкода пользы немного, но когда аргументов больше пяти – метод глПолучитьРозничныйОстатокТСД всё равно будет принимать один строковой параметр – и если при вызове метода на ТСД мы захотим передать лишний параметр или перепутаем порядок аргументов, то ничего страшного не случится. Главное не перепутать имена.

Возврат аргументов также возможен в сложном стиле. Это полезно при возвращении сразу нескольких значений. В Mobile SMARTS есть некоторый аналог ТаблицыЗначений для возвращаемых аргументов, называемый InvokeResult. Код возврата значений с использованием InvokeResult будет выглядеть примерно так:

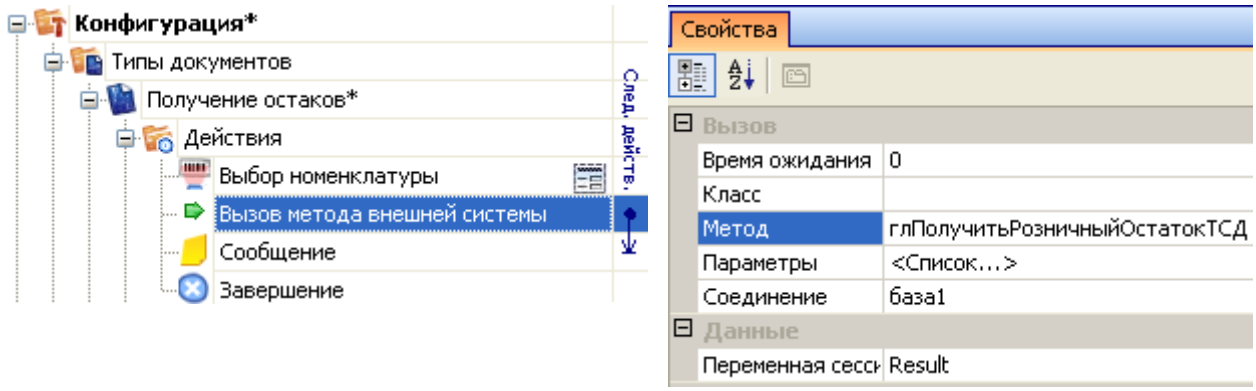
```
    connector = новый COMОбъект("Cleverence.Warehouse.StorageConnector");
    Результаты = новый COMОбъект("Cleverence.Warehouse.InvokeResult");
    Результаты.Добавить("Остаток", Остаток);
    Результаты.Добавить("КодТовара", Номенклатура.Код);
    Результаты.Добавить("НаименованиеТовара", Номенклатура.Наименование);
    …
    Возврат connector.СохранитьОбъектВ_XML(Результаты);
КонецПроцедуры // глПолучитьРозничныйОстатокТСД ()
```

Как видно из кода, метод возвращает строку, содержащую XML с результатами. Этот XML будет правильно проинтерпретирован сервером и клиентом Mobile SMARTS, в результате чего в сессии программы на терминале сбора данных окажутся переменные с названиями, переданными в метод «Добавить».

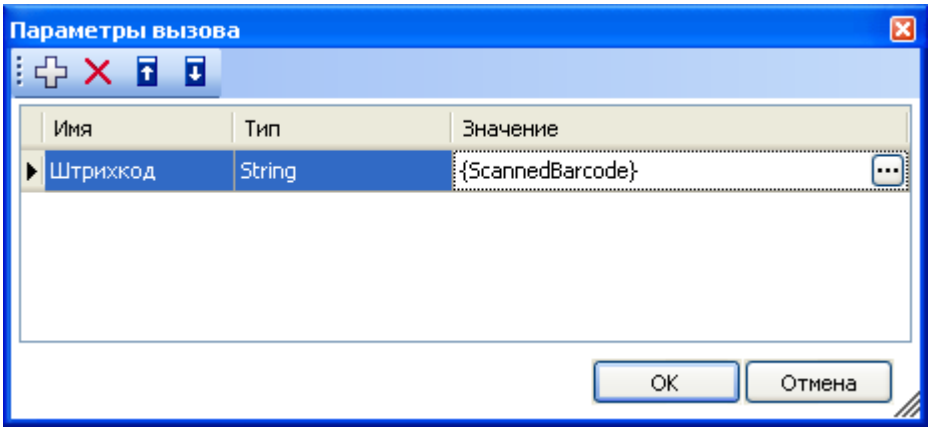
Для проверки работы кода 1С можно воспользоваться отладкой из панели управления Mobile SMARTS (см. раздел «Поддержка коннекторов в самой внешней системе» этого же параграфа).

Чтобы вызвать новый метод 1С с терминала сбора данных, в программу обработки документа следует добавить действие «Вызов метода внешней системы». Для ознакомления с основами программирования в Mobile SMARTS следует смотреть раздел «Основы программирования в Mobile SMARTS».

Создадим новый виртуальный тип документа «Получение остатков» и протестируем в нём интересующий нас функционал:



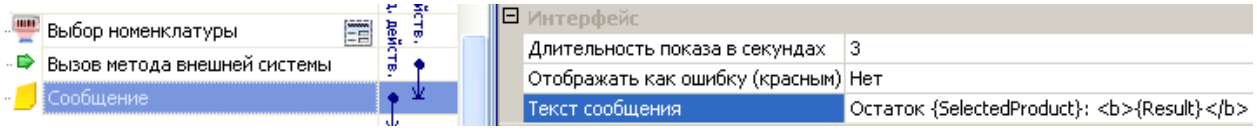
В свойстве «Параметры» действия «Вызов метода внешней системы» указываем именованные параметры для передачи, а именно «Штрихкод» как значение переменной «ScannedBarcode», в которую действие «Выбор номенклатуры» положило строку со считанным штрихкодом:



Примечание: Действие номенклатуры позволяет не только сканировать товары, но и выбирать их из списка при вводе спецштрихкода «0». Чтобы при выборе из списка передать в 1С не «0», а штрихкод выбранного товара, вместо «ScannedBarcode» следует указать «{SelectedProduct.Packing.Barcode}».

В свойстве «Переменная сессии для результата» действия «Вызов метода внешней системы» указано «Result». Это означает, что результат вызова метода 1С попадет в переменную и именем «Result».

Для отображения полученных остатков на экране терминала, укажем в действии «Сообщение» следующий шаблон отображения информации:



Если результаты вызова передаются в виде InvokeResult, то шаблон отображения может выглядеть следующим образом: «Остаток {Result.НаименованиеТовара}: {Result.Остаток}».

§ 12. Разработка коннекторов к внешним системам

Характер изложения информации в данном параграфе предполагает, что вы знакомы с платформой разработки Microsoft .NET, библиотекой .NET Framework и способны отлаживать свои программы.

Если вы недостаточно хорошо владеете требуемыми понятиями и навыками, рекомендуем обратиться к документации платформе .NET.

Введение

Mobile SMARTS 2008 предоставляет несколько средств интеграции с внешними системами:

1. COM-компонента доступа к серверу «Mobile SMARTS» для внешней системы

   Позволяет выгружать на сервер «Mobile SMARTS» справочники пользователей, их групп, справочник номенклатур и единиц измерения, выгружать документы-задания и загружать обратно документы исполненных заданий, а также

формировать на сервере «Mobile SMARTS» необходимые дополнительные справочники произвольного формата.

2. Реализации интерфейса IConnector

Позволяют серверу «Mobile SMARTS» вызывать произвольные методы внешней системы или служебной компоненты, тем самым позволяя серверу «Mobile SMARTS» сигнализировать внешней системе или компоненте о различных событиях, если это явно прописано в конфигурации, а также обеспечивать вызовы сервисных методов с толстого клиента на терминале и возврат соответствующих значений. Интерфейс IConnector можно использовать для написания любых расширений.

Реализация любого из указанных интерфейсов должна быть выполнена под Microsoft .NET Framework 1.1 или 2.0 на любом из доступных языков .NET, скомпилирована в виде отдельной DLL и помещена в папку bin сервера «Mobile SMARTS». Она будет подхвачена сервером сразу после его перезапуска. Если во время загрузки возникли проблемы, они будут отражены в server_errors.log в папке установки сервера.

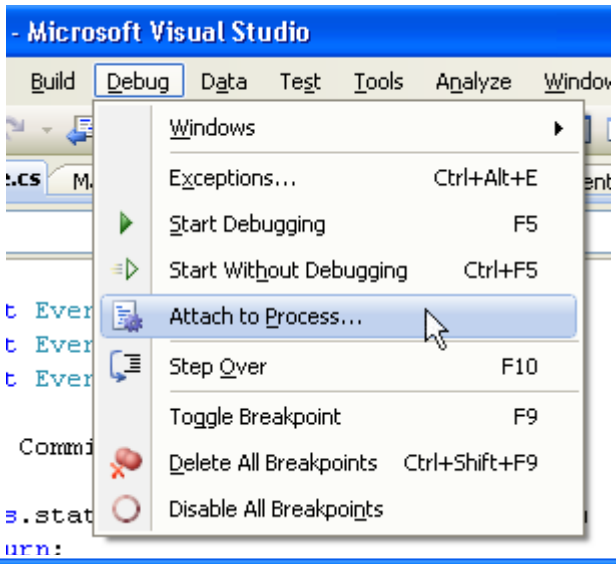
Код коннекторов не только находится в одном приложении с сервером, но и имеет прямой доступ ко всем данным сервера, в частности к среде окружения Environment, задающей текущую конфигурацию, хранилищу документов DocumentStorage и всем справочникам, таким как ProductsBook, UnitsBook и т.д. (см. «Mobile SMARTS 2008 Компонента доступа»). В то время как в COM-компоненте все такие классы указаны, как имеющие экземпляры (чтобы создавать их в COM), на стороне сервера они представлены в виде статических классов с прямым доступом ко всем свойствам и коллекциям.

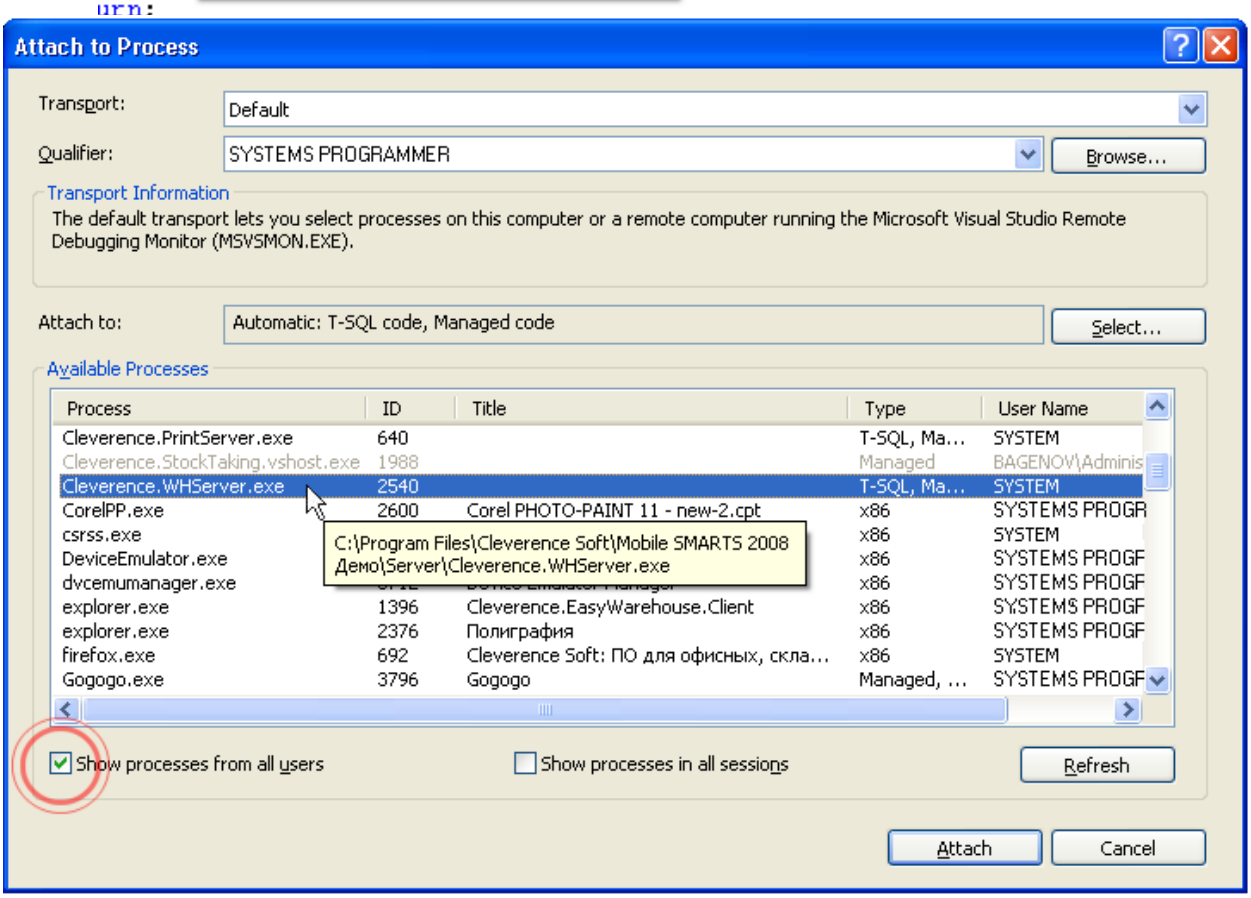
Создание коннекторов под Mobile SMARTS – это серверное это многопоточное программирование с высокими требованиями к скорости выполнения операций. Доступ к методам и коллекциям статических данных сервера не залочен и они не потокобезопасны. При обращении к данным сервера (в особенности на запись) код коннектора должен самостоятельно организовывать критические сессии, обрамляя код структурой:

```
lock(Cleverence.Warehouse.Lock.<Нужное свойство>)
{
        ...
}
```

где «Нужное свойство» – свойство класса Lock, из имени которого понятно объект sync root каких данных оно возвращает.

Среда Microsoft Visual Studio позволяет выполнять отладку разработанного коннектора во время его реальной работы при помощи функции «Attach to process»:





Для этого необходимо открыть свой проект Microsoft Visual Studio, в котором ведется разработка коннектора, скомпилировать его и скопировать полученные dll и pdb файлы в папку bin сервера (вручную или путем PostBuild Event), а затем перезапустить сервер и выполнить «Attach to process». Теперь можно устанавливать в исходном коде коннектора точки останова и проводить отладку.

Интерфейс IConnector

Интерфейс IConnector существует в двух вариантах – серверном и клиентском. Серверный вариант реализует собственно логику работы коннектора. Клиентский вариант позволяет пользователю задавать параметры подключения и любые другие свойства коннектора в панели управления «Mobile SMARTS». Оба варианта должны быть реализованы в виде двух отдельных dll.

Клиентская версия должна реализовать только два свойства: Id и Enabled. Свойство Id отличает конкретный экземпляр коннектора ото всех других коннекторов к той же системе (но другим базам).

Серверная версия должна реализовать свойства Id и Enabled, а также методы Initialize и InvokeMethod. Подключение коннектора к системе происходит не в момент его создания, а в момент первого к нему обращения через метод Initialize, о вызове которого позаботится сервер «Mobile SMARTS».

Оба класса-реализации должны находиться в одинаковых namespace и иметь совершенно одинаковые имена. Схема передачи параметров из панели управления на сервер работает следующим образом: в панели управления пользователь создает экземпляр коннектора, выбрав его по имени в диалоговом окне создания нового внешнего соединения. Пользователь редактирует публичные свойства этого объекта при помощи PropertyGrid. В момент сохранения информации на сервер клиентский экземпляр коннектора сериализуется в XML автоматической сериализацией инфраструктуры Mobile SMARTS, при этом в XML попадают все публичные редактируемые свойства по их именам. Затем на сервере инфраструктура Mobile SMARTS пытается десеарилизовать полученный XML в объект путем поиска класса по имени, указанном в XML, создания его экземпляра и проставления ему всех переданных публичных свойств по именам. Именно поэтому полные имена реализаций должны совпадать, а серверная реализация предоставлять на запись все те свойства с теми же именами, что были реализованы в клиентской версии (при этом серверная версия может содержать любое число собственных свойств и методов сверх этого).

Для реализации клиентской версии интерфейса IConnector следует создать новый проект в Microsoft Visual Studio, подключить в него в качестве Referenced Assembly следующие сборку Cleverence.MobileSMARTS.ComConnector.dll из папки панели управления и создать новый класс, реализующий интерфейс Cleverence.Connectivity.IConnector из этой сборки.

Для реализации серверной версии интерфейса IConnector следует создать новый проект в Microsoft Visual Studio, подключить в него в качестве Referenced Assembly следующие сборки из папки bin сервера:

- Cleverence.DataCollection.dll
- Cleverence.Connetivity.dll
- Cleverence.MobileSMARTS.dll

и создать новый класс, реализующий интерфейс Cleverence.Connectivity.IConnector из серверной сборки Cleverence.Connetivity.

Пример реализации клиентского варианта интерфейса

```
using System;
using System.Configuration;
using System.Runtime.InteropServices;
```

```
namespace MySystem
{
	[Guid("CD86EA85-B168-4823-B509-E916D685BB72")]
	[ProgId("MySystem.MySystemClientConnector")]
	[Cleverence.Warehouse.Design.DisplayTypeName("Шлюз к моей системе v.1")]
	public class MySystemConnector : IConnector
	{
		public MySystemConnector()
		{
		}

		#region IConnector Members

		private string id;
		[System.ComponentModel.Description("Идентификатор.")]
		public string Id
		{
			get { return this.id; }
			set { this.id = value; }
		}

		private bool enabled;
		[System.ComponentModel.Browsable(false)]
		public bool Enabled
		{
			get { return this.enabled; }
			set { this.enabled = value; }
		}

		#endregion

		#region Дополнительные свойства

		private string user;
		[System.ComponentModel.Description("Имя пользователя.")]
		public string User
		{
			get { return this.user; }
			set { this.user = value; }
		}

		private string password;
		[System.ComponentModel.Description("Пароль.")]
		[System.ComponentModel.TypeConverter("Cleverence.Warehouse.Design.
	PasswordConverter, Cleverence.Warehouse.Com.Design")]
		public string Password
		{
			get { return this.password; }
			set { this.password = value; }
		}

		#endregion
	}
}
```

Пример реализации серверного варианта интерфейса

```
using System;
using System.Configuration;
using System.Runtime.InteropServices;
```

```
namespace MySystem
{
	public class MySystemConnector : IConnector, IDisposable
	{
		public MySystemConnector()
		{
		}


		private MYSYS.Interop.COMConnector connector;


		private string user;
		public string User
		{
			get { return this.user; }
			set { this.user = value; }
		}

		private string password;
		public string Password
		{
			get { return this.password; }
			set { this.password = value; }
		}

		#region IConnector Members

		private string id;
		public string Id
		{
			get { return this.id; }
			set { this.id = value; }
		}

		public bool Initialized
		{
			get { return this.connector != null; }
		}

		private bool enabled = true;
		public bool Enabled
		{
			get { return this.enabled; }
			set
			{
				if(value == false && this.connector != null)
					this.Dispose();

				this.enabled = value;
			}
		}

		public void Initialize()
		{
			this.Dispose();
			this.connector = new MYSYS.Interop.COMConnector();
			this.connector.Logon(this.user, this.password);
		}
```

```
            public object InvokeMethod(string methodName, object[] args)
            {
                  if(this.enabled == false)
                        throw new InvalidOperationException(
                              this.GetType().Name + " is not enabled.");

                  if(this.connector == null)
                        throw new InvalidOperationException(
                              this.GetType().Name + " is not initialized.");

                  return this.connector.Call(methodName, args, null, null);
            }


            #endregion


            #region IDisposable Members

            public void Dispose()
            {
                  if(this.connector == null)
                        return;

                  Marshal.ReleaseComObject(this.connector);
                  this.connector = null;
            }

            #endregion

            ~ MyConnector()
            {
                  this.Dispose();
            }
      }
}
```

Указатель

Ш

Э

Я

Контакты

 По вопросам поддержки обращайтесь по указанным реквизитам компании:

Cleverence Soft,
email: support@cleverence.ru