1С:ПРЕДПРИЯТИЕ
система программ

ИКС ТЕХНОЛОГИИ

Колпинский Н.Н.

1С:ПРЕДПРИЯТИЕ

Практика программирования на платформе V7

1С:ПРЕДПРИЯТИЕ.

Практика программирования на платформе V7

ИКС Технологии Москва, 2002 г.

Документация: Колпинский Н.Н.

Руководитель учебного центра «ИКС Технологии»: Романова С. В.

Компьютерная верстка : Макарчук М.А.

Компания «ИКС Технологии», 2002 г.

728-9191 (многоканальный)

training@xtek.ru www.xtek.ru

Содержание

I.Введение

Данные материалы рассчитаны на пользователей, знакомых с интерфейсом «1С:Предприятия» и имеющих навыки программирования. Основное назначение методических материалов - помочь начинающим разработчикам в конфигурировании системы «1С:Предприятие». Они также могут использоваться как дополнение к книгам по администрированию и конфигурированию системы «1С:Предприятие». Материалы построены на основе 2-х летнего опыта ведения соответствующего курса и содержат ответы на наиболее актуальные для начинающих разработчиков вопросы по конфигурированию. Они используются в качестве методического пособия при прохождении курса по настройке системы «1С:Предприятие», а также могут использоваться для самостоятельного обучения конфигурированию на платформе V7.

Материал изложенный в данном пособии охватывает все виды объектов системы 1С: Предприятие и позволяет самостоятельно составить техническое задание на разработку конфигурации на платформе V7.

Рост объемов обрабатываемой информации выдвигает на передний план проблему эффективности средств организации данных и доступа к ним. Для этого создаются различные системы баз данных. Основными критериями при построении таких систем являются быстрота доступа к информации и возможность группировки различных данных в виде удобном для пользователя. Система обслуживания данных состоит из специальных форм хранения данных, которые позволяют объединить данные с общими свойствами в отдельные структуры называемые *базами данных,* и программы, позволяющей сохранить данные в базах данных, считать любой элемент данных из них, а также создать удобный интерфейс для интерактивной работы с базами данных.

При проектировании системы нужно учитывать, что для того чтобы получить из нее какие-либо данные, необходимо сначала накопить соответствующие записи в системе.

Пример предложенный для рассмотрения раскрывает основные свойства и методы объектов, а также приёмы организации учета хозяйственной деятельности предприятия. При создании реальных баз данны последовательность разработки отличается от предложенной в примере: вначале определяются параметры необходимые для анализа деятельности предприятия и формы их представления - соответственно объекты учета и отчета; после этого создаются документы, которые определяют, как накапливается информация в объектах учета, справочники и другие базовые объекты.

1.1. Создание баз данных

Базу данных можно представить в виде таблицы. Набор данных в базе состоит из *записей,* а каждая запись данных — из отдельных *полей.* Структура всех записей базы одинакова: все они содержат одну и ту же последовательность полей, но наполнение полей у каждой записи свое. Таким образом, в строке таблицы содержится *запись* элемента данных, свойства которого расположены в отдельных *полях* строки и сгруппированы по колонкам таблицы. Создание базы данных производится в несколько этапов.

1.При определение структуры базы данных устанавливается, из каких полей будет состоять отдельная запись базы данных и задается тип каждого поля.

2.Создание интерактивных форм для просмотра, ввода и редактирования данных. При создании форм можно блокировать доступ к конфиденциальной информации в базе данных, а также ограничить «рабочую зону » оператора, сконцентрировав его внимание на необходимых ему полях.

1.2.Реляционные базы данных

Рассмотрим пример создания базы данных адресов. Запись адреса состоит из следующих составляющих:

Почтовый индекс,

Страна,

Город,

Административный округ,

Улица,

Дом,

Подъезд,

Квартира,

Абонент.

Можно конечно для каждого абонента записывать полную строку адреса, но гораздо удобнее создать отдельные списки стран, городов, округов и улиц для

того, чтобы не набирать каждый раз одни и те же значения. Для этого необходимо структурировать запись адреса, разбив её на отдельные поля.

Почтовый индекс	Страна	Город	Адм. Округ	Улица	Дом	Подъезд	Квартира	Абонент
1	2	3	4	5	6	7	8	9
256986	Россия	Москва	Северо-Восточный	Полярная	14	5	44	----------
986475	Россия	Москва	Северо-восточный	Полярная	15	4	88	----------
127000	Россия	Москва	Северо-восточный	Сухонская	12	1	22	----------

Таблица 2	Таблица 3	Таблица 4	Таблица 5
Страны	Города	Округа	Улицы
Россия	Москва	Северо-восточный	Сухонская
Украина	Минск	Северный	Полярная
Белоруссия	Архангельск		Шокальского

Чтобы можно было в первую таблицу подставлять значения из других таблиц нужно «привязать» таблицы 2-5 к соответствующим ячейкам таблицы 1. Для этого нужно задать тип полей таблицы 1: 1 и 6-9 — число, 10 — строка, 2 — тип значения Страны, 3 — тип значения Города, 4 — тип значения Округа, 5 — тип значения Улицы.

Таким образом, реляционные базы данных (БД) представляют собой систему связанных таблиц. Для связи между таблицами в них создаются ключевые поля. Ключевое поле является как бы адресом данной записи в таблице, с помощью которого устанавливается отношение (реляция) между различными базами данных. Теперь если мы хотим записать (или получить) в поле одной таблицы элемент данных из другой таблицы, мы должны определить тип поля первой таблицы как «вторая таблица», а в значение поля (первой таблицы) записать значение ключевого поля записи второй таблицы.

В нашем примере надо учесть, что таблицы городов будет подчинена таблице стран, таблица округов - таблице городов, а таблица улиц - таблице округов.

Ключ	Почтовый индекс	Страна	Город	Адм.Округ	Улица	Дом	Подъезд	Квартира	Абонент
A1	296528	C1	Г1	O1	У2	14	5	44	-----
A2	258335	C1	Г1	O1	У2	15	4	88	-----
A3	437557	C1	Г1	O1	У1	12	1	22	-----

Таблица 2

Ключ	Страны
C1	Россия
C2	Украина
C3	Белоруссия

Таблица 5

Ключ	Подчинение	Улицы
У1	O1	Сухонская
У2	O1	Полярная
У3	O1	Шокальского

Таблица 3

Ключ	Подчинение	Города
Г1	C1	Москва
Г2	C3	Минск
Г3	C1	Архангельск

Таблица 4

Ключ	Подчинение	Округа
O1	Г1	Сев.Вост.
O2	Г1	Северный
O3	Г2	

1.2.1.Индексация БД

Для облегчения поиска элемента данных по БД создаются таблицы индексов, в которых содержатся первичные ключи — значения ключевых полей БД, и описание очередности записей. Таблица БД остается неупорядоченной. Таблица индексов, как правило, гораздо меньше таблицы БД, поэтому поиск и сортировка записей таблицы индексов происходит гораздо быстрее, чем поиск и сортировка по таблице БД.

Наряду с первичным индексом таблица индексов может содержать индексы по другим полям таблицы БД, что позволяет упорядочить БД по значению индексированного поля. Записи индексных таблиц можно сортировать.

Таблица индексов 1	Таблица индексов 2	Таблица индексов 3	Таблица индексов 4	Таблица индексов 5
A1	C1	Г1	O1	У1
A2	C2	Г2	O2	У2
A3	C3	Г3		У3

1.3.Основные принципы обработки информации

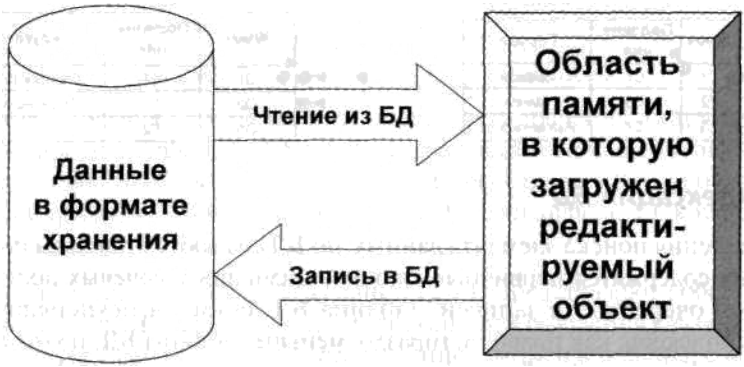
С точки зрения управления информацией, различают два типа памяти, к которым производится обращение из системы управления. Первый тип — это постоянная память, формат хранения информации — файлы на жестком диске или другом носителе. Эта память, как правило, энергонезависимая, позволяющая хранить большие объемы информации и относительно

медленная. Второй тип — это оперативная память, в которой формируются

интерактивные образы и выполняются алгоритмы обработки информации. Оперативная память — энергозависима, то есть при выключении питания её состояние не сохраняется, это быстрая и постоянно обновляемая из формата хранения память.

Таким образом, процесс обработки информации состоит из следующих основных этапов:

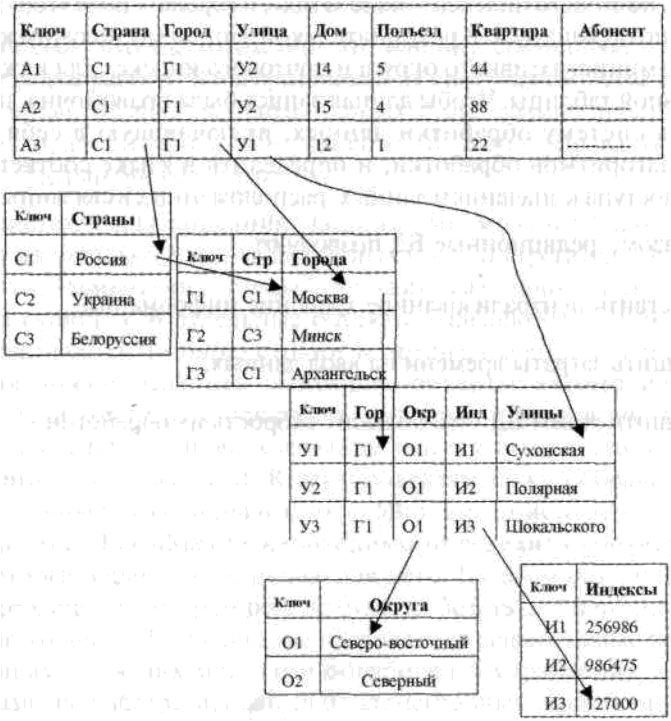
- Чтение информации из формата хранения в оперативную память,
- Интерактивное отображение и изменение информации,
- Запись информации в формат хранения.



1.3.1.Создание подчиненных структур таблиц и обращение к полям таблиц

Можно создать такую совокупность связанных таблиц, которая будет отражать реальное подчинение объектов информации. В разных странах в принципе могут быть города с одинаковыми названиями, также как и улицы в разных городах, но в округе не может быть улиц с одинаковыми названиями и

почтовый индекс также однозначно определяется при выборе улицы. Таблицы для хранения информации будут иметь следующий вид:



Ключ	Страна	Город	Улица	Дом	Подъезд	Квартира	Абонент
А1	С1	Г1	У2	14	5	44	----------
А2	С1	Г1	У2	15	4	88	----------
А3	С1	Г1	У1	12	1	22	----------

Ключ	Страны
С1	Россия
С2	Украина
С3	Белоруссия

Ключ	Стр	Города
Г1	С1	Москва
Г2	С3	Минск
Г3	С1	Архангельск

Ключ	Гор	Окр	Инд	Улицы
У1	Г1	О1	И1	Сухонская
У2	Г1	О1	И2	Полярная
У3	Г1	О1	И3	Шокальского

Ключ	Округа
О1	Северо-восточный
О2	Северный

Ключ	Индексы
И1	256986
И2	986475
И3	127000

Такая структура позволяет при выборе страны получить список городов этой страны, при выборе города — список улиц, при выборе улицы однозначно определяется административный округ и почтовый индекс абонента. Интерактивная форма для представления данной системы будет содержать следующие значения:

Ключ	Страна	Город	Улица	Адм. Округ	Почтовый индекс	Дом	Подъезд	Квартира	Абонент
А1	С1	Г1	У2	Улица.Окр	Улица.Инд	14	5	44	----------
А2	С1	Г1	У2	Улица.Окр	Улица.Инд	15	4	88	----------
А3	С1	Г1	У1	Улица.Окр	Улица.Инд	12	1	22	----------

Обратим внимание на поля «Адм. Округ» и «Почтовый индекс». В них обращение к конкретному значению выполняется посредством разделенных точкой названий полей соответствующих таблиц. Значение улицы определено в

каждой записи данных - в строке таблицы, поэтому к нему можно

обращаться. С другой стороны значение улицы, тоже имеет свою структуру, в которую входят, помимо названия, значения административного округа — поле «Окр», почтового индекса — поле «Инд», и города — поле «Гор», к которым так же можно обращаться. В результате такой записи мы получим конкретные значения административного округа и почтового индекса для каждой строки интерактивной таблицы. Чтобы данная запись была правомочна, необходимо определить систему обработки данных, включающую в себя язык для описания алгоритмов обработки, и определить в языке соответствующую операцию доступа к значениям данных, расположенных в связанных таблицах.

Таким образом, реляционные БД позволяют

- осуществить централизованное хранение информации,

- уменьшить затраты времени на ввод данных,

- уменьшить объем БД, что повысит скорость их обработки.

II.Основные понятия системы

Ключевым понятием системы 1С: Предприятие является ***объект.***

Представьте себе, что мы хотим описать машину, состоящую из множества узлов, в процессе работы. В алгоритмических системах свойства машины мы можем представить в виде элементов данных, описываемых простыми типами данных: число, дата и строка, изменяя которые можно управлять машиной в целом или отдельными её узлами.

В нашей системе, для описания машины, мы можем создать структуру, состоящую из отдельных взаимосвязанных узлов и агрегатов машины, а так же их свойств. Элементами системы будут как сама машина, так и её составные части, и их свойства. В отдельные объекты выделяются группы элементов данных с одинаковыми параметрами и предназначением. При описании автомобиля можно выделить следующие объекты: автомобиль в целом, система подачи топлива, карбюратор, жиклеры, другие агрегаты системы подачи топлива с детализацией, ходовая часть и её узлы с детализацией, другие части машины с детализацией. К этим объектам можно обращаться, как к элементам данных, а не только к их свойствам, описываемым простыми типами данных. Для объекта необходимо определить соответствующие методы, которые характерные только для него. В самом деле, очевидно, что методы управления карбюратором не будут работать для передней подвески или рулевой колонки. Такая система позволяет создавать новые типы данных с характерными для них методами обработки и управления, что, в свою очередь, позволяет абстрагироваться от элементарных свойств описываемого предмета, и создавать более глубокие и разветвленные связи в реляционной структуре данных.

Таким образом, ***объект*** - это инкапсуляция данных и алгоритмов их обработки (от английского encapsulation - пакетирование). Другими словами - это формальное описание совокупности понятий, характеризующих элементы данных с одинаковыми свойствами (синий и красный — это различные значения одного и того же свойства - «цвет») и предназначением, в котором объединяются как свойства этих данных, так и методы обработки, характерные для типа данных. В контексте баз данных объект — совокупность данных с

одинаковыми свойствами и предназначением, имеющих общие структуры хранения и интерактивного представления, и методами их обработки.

Объектная архитектура - это архитектура, в которой процессы, структуры данных (файлы), операции ввода/вывода информации и любые другие составные части архитектуры представляются как объекты.

В системе "1С:Предприятие" нельзя создать любой объект с заданными свойствами. Эта система изначально содержит в себе типовые наборы свойств и методов объектов, и можно создавать в системе объекты, используя эти наборы, называемые ***видами метаданных*** их можно представить в виде некоторых шаблонов объектов, создаваемых в "1С:Предприятии".

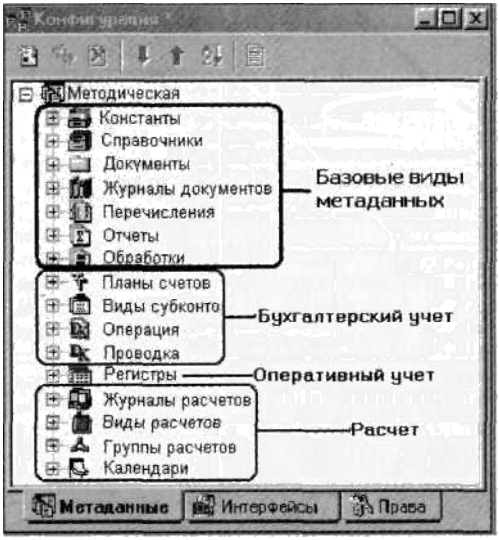
Это те "кирпичики" и "шестеренки" из которых создаются объекты системы. Таким образом мы будем создавать не любые объекты, а объекты метаданных.

Благодаря такой структуре существенно уменьшается время разработки БД. Экономится время на описание объектов: в "1С:Предприятии" связанный объект с двумя - тремя десятками реквизитов (связей) можно "накидать" за

5-7 минут. Основное время разработки при этом уделяется описанию

алгоритмов управления данными и их обработки средствами системы. Используя реляционную структуру полученной базы, можно создавать всевозможные выборки для генерации отчетов.

Другими словами, *метаданные* - это "информация о данных", представляющая виды данных, характерные для системы "1С:Предприятие".



Объекты метаданных определяются ***видами метаданных,*** которые мы видим в корне дерева метаданных: это константы, перечисления, отчеты, обработки, справочники, документы и пр.. Свойства вида метаданных определены в самой системе " 1С:Предприятие" и распространяются на любой объект метаданных данного вида. *Для объектов метаданных вида Перечисление, Справочник, Документ, Регистр, ЖурналРасчетов, Календарь и Счет идентификатор вида объекта метаданных возвращается методом Вид().*

Объект метаданных - это объект определенного в конфигурации вида метаданных.

Таким образом:

•Объекты метаданных, создаваемые при конфигурировании, подразделяются по ***видам.***

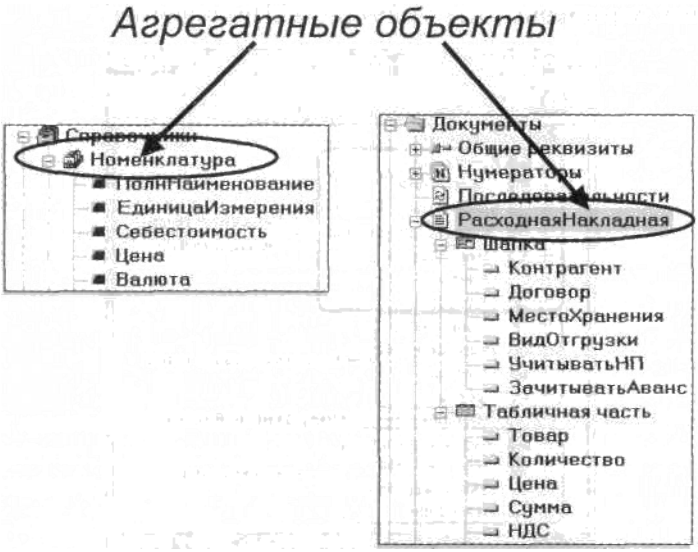
•Свойства вида метаданных определены в самой системе «1С:Предприятие» и распространяются на любой объект метаданных данного вида.

•Мы можем создать объект только определенного на платформе вида метаданных.

•Объектом в системе "1С:Предприятие" является как сама конфигурация, так и

любой объект метаданных, являющийся элементом конфигурации.

Объект метаданных, имеющий в своем составе подчиненные объекты, называется ***агрегатным объектом,*** например объекты типа Справочники или Документы. Доступ к подчиненным объектам осуществляется через атрибуты агрегатного объекта.



Примерами объектов метаданных являются конкретные объекты определенного вида метаданных, создаваемые пользователем в процессе конфигурирования:
Справочник.Номенклатура, Документ.РасходнаяНакладная, а также атрибуты и реквизиты агрегатных объектов метаданных: например цена в справочнике Номенклатура или в документе расхода товаров.

Таким образом, при создании нового объекта метаданных основные его свойства: ключевые поля, методы и др., задаются системой в соответствии со свойствами выбранного вида метаданных. Система "1С:Предприятие", однако, позволяет присоединять объекты, созданные в других средах разработки, с помощью директивы *ЗагрузитьВнешнююКомпоненту("*<Имя файла>*").*

2.1. Типы данных определенные в системе «1С:Предприятие»

В системе "1С:Предприятие" определены следующие типы данных: ***Базовые типы данных:*** число, строка и дата, - определены во встроенном языке;

Служебные типы метаданных, определенные во встроенном языке, но не

представлены в дереве метаданных: это объекты Периодический, Таблица,

Запрос, СписокЗначений, ТаблицаЗначений, Бухгалтерские Итоги, Файловая система (ФС), Xbase, Форма и Метаданные;

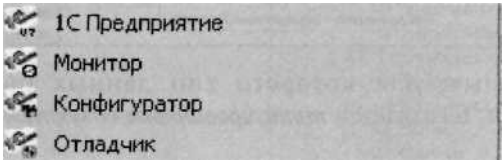
Некоторые объекты метаданных могут образовывать типы значений для других объектов метаданных. Они называются ***типообразующими объектами метаданных.***

Объект метаданных, для которого тип данных определяется при конфигурировании, называется ***типизированным объектом метаданных.***

Таким образом, связи между типизированными и типообразующими объектами отображают реляционную структуру БД.

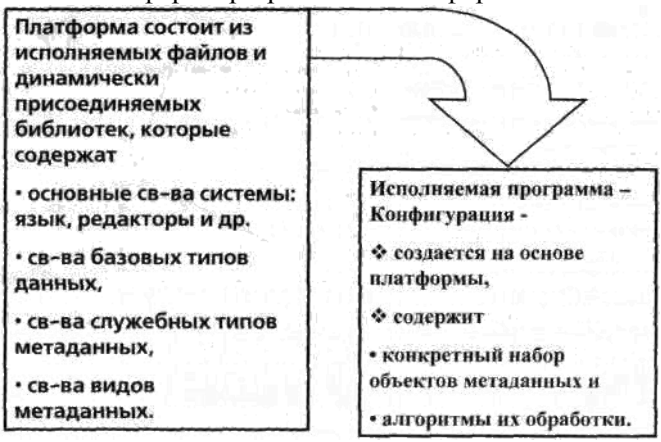
III.Структура системы 1С:Предприятие

Система «1С:Предприятие» предназначена для решения задач по автоматизации учёта товарных и материальных средств, взаиморасчётов с контрагентами, расчёта заработной платы, расчёта амортизационных средств, бухгалтерского учёта по любым разделам и других необходимых предприятию разрезов учёта. Она представляет собой систему управления различными объектамив предметной области и позволяющих записывать, корректировать и получать информацию в заданном виде и формате. Существует четыре режима запуска системы: собственно Предприятие, где вводятся конкретные значения элементов данных, т.е. производится заполнение базы данных, Конфигуратор, описывающий структуру системы и алгоритмы обработки данных, Отладчик - для отладки алгоритмов в пошаговом режиме и хронометрирования времени выполнения операторов программы, и Монитор - для контроля над деятельностью пользователей в системе.



Система «1С:Предприятие» является совокупностью двух тесно связанных частей: технологической платформы и конфигурации.

Технологическая платформа разрабатывается фирмой «1С». Конфигурация, в



отличие от технологической платформы, может произвольно изменяться конечным пользователем. В *конфигураторе* системы настраиваются конкретный набор объектов и особенности учёта. К ним относятся основные свойства плана счетов, виды аналитического учёта, состав и структура используемых справочников, документов, отчётов, система хранения

оперативных итогов и т.д. На уровне системы «1С:Предприятие» определены сами понятия и стандартные операции по их обработке. Средства конфигурирования позволяют описать конкретные структуры информации (объекты) и алгоритмы (процедуры и функции), описывающие специфику их обработки, для отражения различных особенностей учёта.

Конфигуратор позволяет записать в конфигурацию информацию об авторе конфигурации и его логотип, и, если необходимо, защитить эту информацию паролем от несанкционированного изменения. Просмотр информации об авторе конфигурации выполняется при помощи пункта «О программе» из меню «Помощь» главного меню системы 1С:Предприятие - как в режиме запуска «1С:Предприятие», так и в режиме Конфигуратора. Кроме этого, Конфигуратор позволяет также заменить центральную часть заставки системы в режиме запуска «1С:Предприятие».

3.1. Компонентная структура 1С:Предприятия

Система «1С:Предприятие» имеет компонентную структуру. Основные объекты: Текст, Таблица, Запрос, Отчёты, Обработки, Константы, Справочники, Документы, Журналы документов и Перечисления, и базовые типы данных поставляются с любой компонентой. В отдельные компоненты выделены объекты учета.

Компонента «Бухгалтерский учёт»

Компонента «Бухгалтерский учёт» включает типы метаданных *Бухгалтерские счета, Виды субконто, Операции, Проводки и Бухгалтерские итоги,* что позволяет отразить (описать и анализировать) хозяйственную деятельность предприятия с точки зрения бухгалтерского учёта.

Компонента «Оперативный учёт»

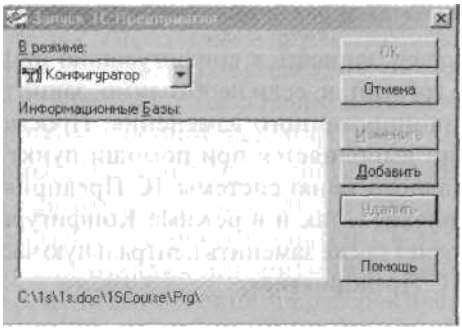
Компонента «Оперативный учёт» включает типы метаданных *регистры и* понятие *точка актуальности итогов,* позволяющие регистрировать и получать информацию о движениях и остатках товарных, материальных, денежных и других средств предприятия ***в реальном времени.***

Компонента «Расчёт»

Компонента «Расчёт» включает типы метаданных *календари, виды и группы расчётов, и журналы расчётов* и предназначена для автоматизации сложных периодических расчётов, в том числе - с пересчётом результатов «задним числом» и ведением архива расчётов за прошедшие периоды.

3.2. Регистрация новой конфигурации

Запустим 1С:Предприятие через меню Пуск или, используя соответствующий ярлык на «Рабочем столе».



Выберем режим «Конфигуратор» и нажмем кнопку «Добавить» для регистрации новой базы данных.

В окне «Регистрация Информационной Базы» введем название базы, выберем или создадим каталог для её размещения и нажмем кнопку «ОК». Затем нажмем кнопку «ОК» в окне запуска «1С:Предприятия». Если у вас загружена SQL-версия программы (файл 1cv7s.exe), вам будет предложено выбрать формат базы данных (БД) SQL или DBF. Если вы выбрали формат DBF, то вам будет предложено выбрать нужную кодовую страницу.

Задаваемая сортировка алфавитно-цифровых символов должна совпадать с установками операционной системы, в среде которой будет запускаться система «1С:Предприятие». В случае несовпадения установок запуск

невозможен. Следует помнить, что система «1С:Предприятие» не

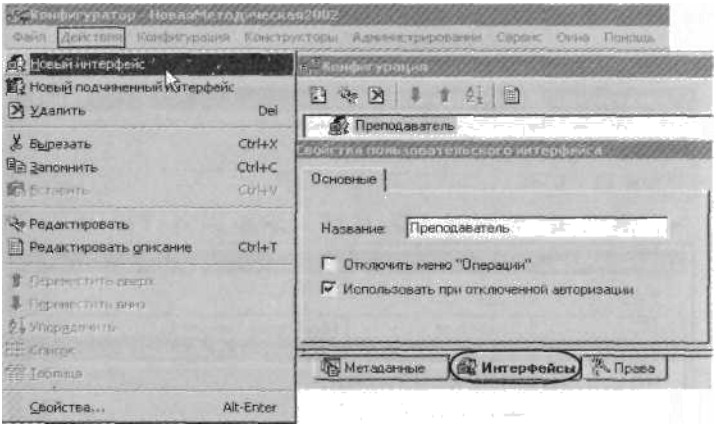
приспособлена для работы одновременно с несколькими кодовыми страницами. Поэтому в процессе работы с одной информационной базой недопустимо переключение национальных настроек операционной системы, приводящее к смене кодовых страниц. **Кодовую страницу можно изменить в Конфигураторе - меню «Администрирование».**

Создание интерфейсов и пользователей.

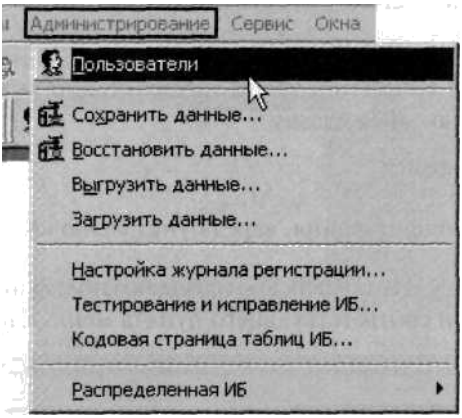
Интерфейсы нужны в системе, чтобы интерактивно активизировать объект или выполнить какую- либо задачу.

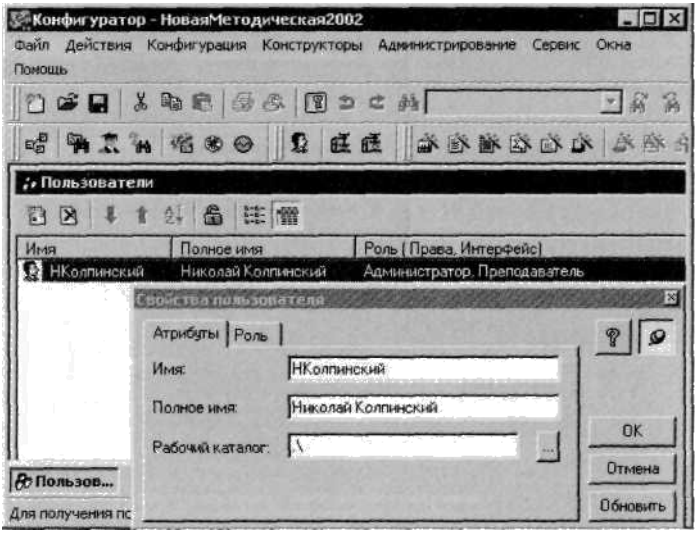
Для создания интерфейса

- откроем окно конфигурация, через пункт меню «Конфигурация»,
- выберем закладку «Интерфейсы» и создадим новый интерфейс с помощью клавиши Ins или соответствующего пункта меню «Действия».



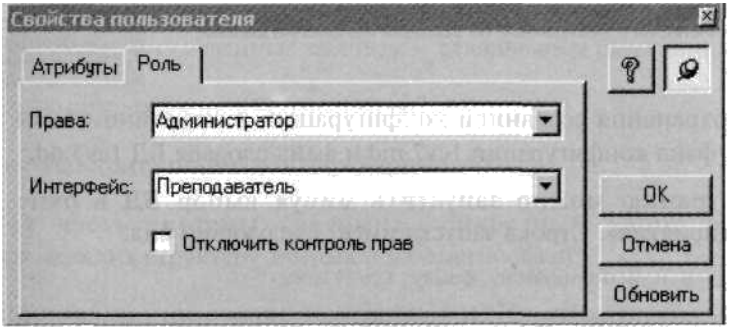
Для авторизации доступа в систему создается список пользователей. Создадим пользователя с помощью пункта меню «Администрирование» или соответствующей пиктограммы в панели «Администрирование», отредактируем его свойства.



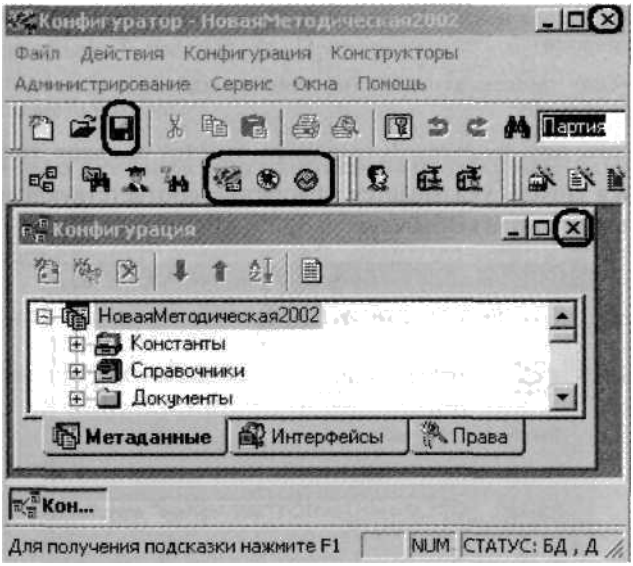


Каждому пользователю определяется пароль, пользовательский каталог для записи временных файлов, набор доступных интерфейсов и набор прав на интерактивное и программное использование объектов системы. По

умолчанию в системе имеется набор прав "Администратор", который, тем не менее, не предоставляет все права доступные в системе.

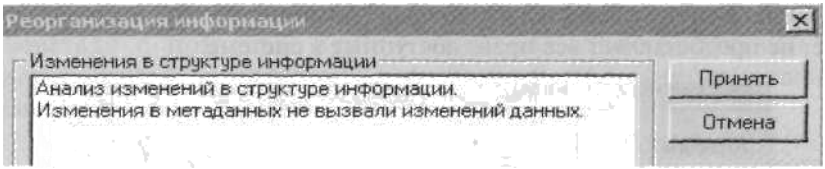


После этого сохраним изменения в конфигурации.



Сделать это можно различными способами, как показано на рисунке. При запуске режимов "Предприятия", "Отладчика" или "Монитора" из конфигуратора сначала сохраняются изменения в конфигурации, после чего запускается выбранный режим.

При сохранении изменений в конфигурации производится реструктуризация существующей базы данных, что может повлечь потерю некоторых данных в результате некорректного изменения соответствующих объектов системы. Поэтому обязательно внимательно просмотрите список измененных объектов, прежде чем принять изменения в конфигурации.



После сохранения созданной конфигурации, в выбранном вами каталоге появятся файл конфигурации 1cv7.md и файл словаря БД 1cv7.dd.

Конфигурацию можно запустить минуя выбор БД в окне запуска «1С:Предприятия». Строка запуска имеет следующий вид:

```
"<Путь к исполняемому файлу 1cv77.exe>"

<Режим:enterprise | config | debug | monitor> /

D"<Путь к БД>"

/N"<Имя пользователя системы>" (список пользователей создается в конфигураторе)

/U"<каталог пользователя>" (в строке запуска его можно переопределить)

/P"<пароль пользователя>". (задается строкой в явном виде)
```

Обычно строка запуска указывается в свойствах соответствующего ярлыка на закладке «Ярлык» в поле «Объект».

Вспомним, что мы узнали из пройденного материала.

Вопросы для самоконтроля

Для чего создают связанные таблицы — **реляционные** базы данных?

Для чего создаются индексные файлы?

Из чего состоит система «1С: Предприятие»?

Какая часть системы содержит конкретный набор объектов, определяющих структуру данных и их взаимосвязи?

Из каких элементов состоит платформа «1С: Предприятие»?

Чем отличаются компоненты платформы?

Что такое объект в системе «1С:Предприятие»?

Что такое метаданные?

Какие виды объектов можно создавать на платформе «1С:Предприятие»?

Какие типы объектов изначально определены в системе «1С:Предприятие»?

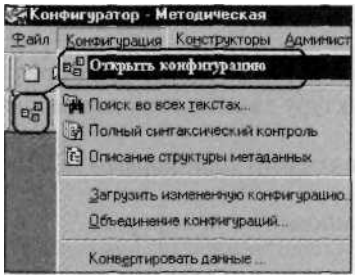
Какие части системы содержатся в платформе «1С: Предприятие»?

Какой самый критичный момент при сохранении изменений в системе?

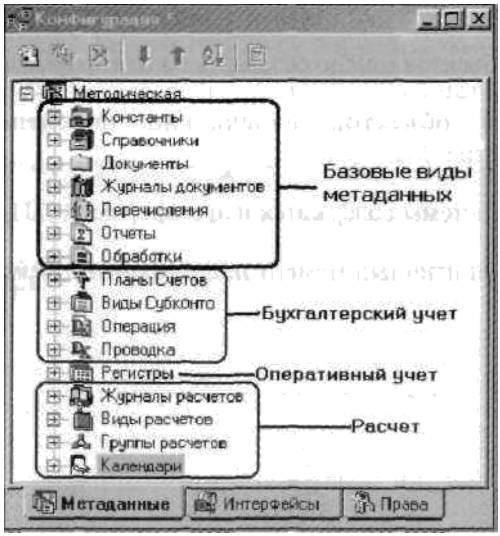
IV.Элементы конфигурирования на платформе 7.7

4.1. Работа с объектами метаданных

Работа с объектами метаданных выполняется в окне «Конфигурация». Для открытия этого окна выберите пункт «Конфигурация» из меню «Операции» главного меню Конфигуратора.



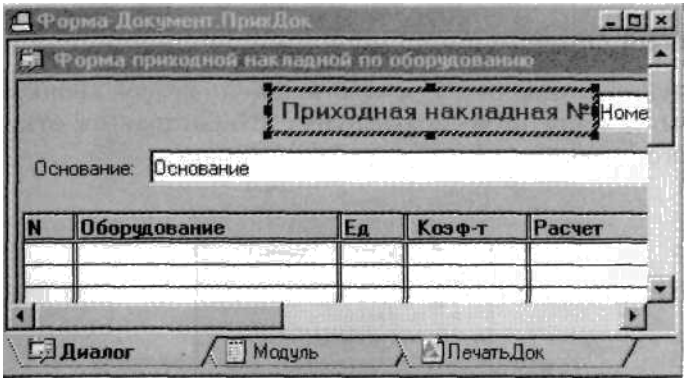
При его открытии активной становиться закладка «Метаданные».



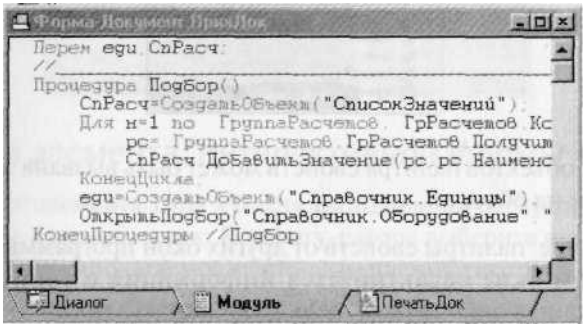
4.2. Редактор форм

Одним из свойств некоторых объектов метаданных является форма. Для редактирования её свойств в системе «1С: Предприятие» существует три вида редакторов:

- редактор диалогов (представлен в окне редактора форм закладкой с именем «Диалог»);



- редактор текстов (закладка называется «Модуль», так как он используется для редактирования программных модулей);



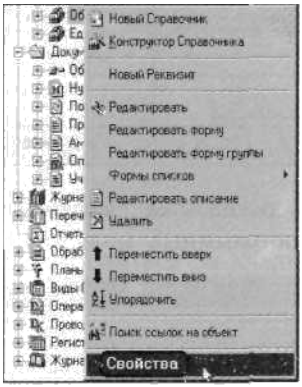
- табличный редактор (закладка по умолчанию называется «Таблица» и используется для редактирования шаблона печатных форм объектов). В форме может быть несколько табличных редакторов с различными названиями, так же табличный редактор может отсутствовать в форме.

- Для нового объекта метаданных, имеющего форму, последняя всегда

содержит три закладки. При открытии форма автоматически открывается на закладке «Диалог» - редакторе диалогов.

4.3.Палитра свойств

Палитру свойств можно открыть, если щелкнуть правой кнопкой мыши на выбранном объекте и выбрать пункт «Свойства» из открывшегося контекстного меню.

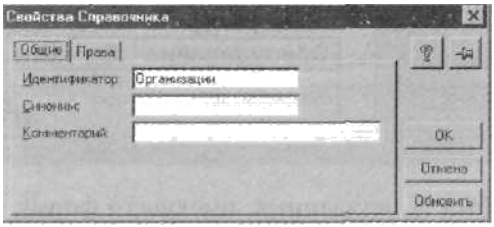


Для некоторых объектов палитра свойств может быть вызвана также двойным щелчком мыши на объекте.

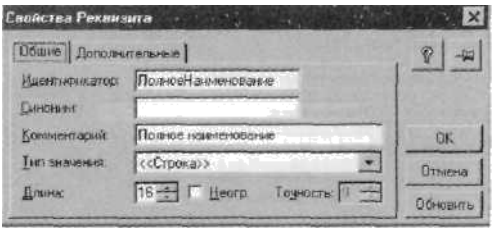
Основное отличие палитры свойств от других окон программы заключается в том, что в этом окне редактируется информация о свойствах объекта, выбранного в каком-либо другом окне. Если нажать кнопку («Прикрепить»), палитра свойств постоянно будет присутствовать на экране, и, в зависимости от того, какой объект выбран, состав информации в ней будет изменяться. Палитра свойств всегда присутствует на экране в единственном «экземпляре».

В конфигураторе существует несколько палитр свойств:

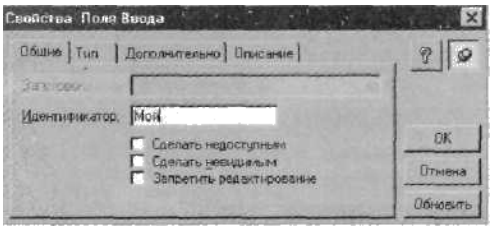
1) для объекта метаданных - позволяет ввести идентификатор объекта, «всплывающий» комментарий, а также права доступа по категориям прав, и правила миграции данных в распределенных базах.



2) для реквизитов позволяет ввести также тип объекта, форматирование для базовых типов и дополнительные свойства в зависимости типа объекта.

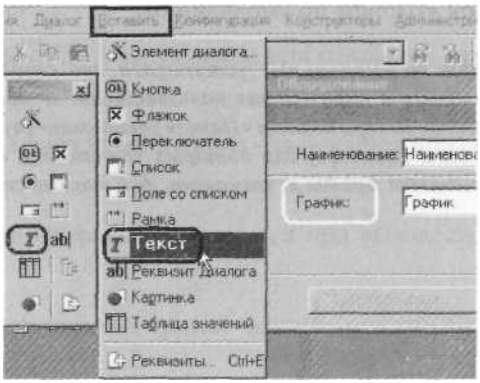


3) для полей ввода элементов формы, позволяющих вводить не только реквизиты объекта, но и справочные поля.



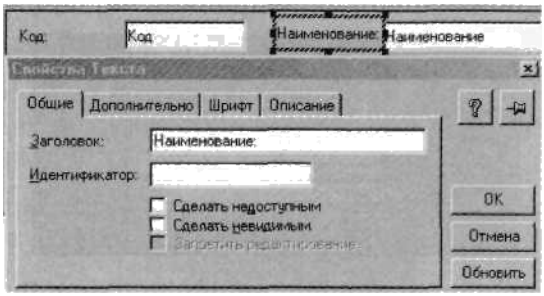
4.4.Свойства элемента интерактивной формы «Текст»

Вставка в диалог элемента типа «Текст» выполняется следующим образом: в меню «Вставить» главного меню Конфигу-ратора выберите пункт «Текст» и в диалоге очертите предполагаемое место размещения текста.



Для редактирования свойств текста используется палитра свойств, для ее вызова следует дважды щелкнуть элемент мышью. На закладке «Общие»

палитры свойств в поле «Заголовок» можно ввести текст, который должен выводиться в интерактивной форме объекта.



Изначально в этом поле находится слово «Текст», которое подставляет редактор диалогов.

Закладка «Шрифт» позволяет задавать начертание, размер, цвет и другие свойства шрифта. Чтобы изменить шрифт установленный по умолчанию следует снять флажок «Шрифт по умолчанию».

Если флажок «Шрифт по умолчанию» установлен, то свойства шрифта будут определяться установками в параметрах пользователя. В этом случае изменения системного шрифта, сделанные при помощи «Панели управления» (пункт «Экран») окажут влияние на все элементы диалогов, для которых установлен флажок «Шрифт по умолчанию» - у них изменится шрифт и, возможно, что диалоги станут нечитаемыми.

V. Формат исходных текстов программных модулей. Элементы программирования на встроенном языке 1С

5.1.Виды программных модулей

Для написания процедур и функций в программе предусмотрено наличие различных видов модулей, которые запускаются в строго определенные моменты работы с системой.

Модуль Формы списка справочника - запускается при вызове формы списка справочника.

Модуль Формы группы справочника - запускается при открытии формы группы справочника.

Модуль Формы элемента справочника - запускается при открытии формы элемента справочника.

Модуль **Формы документа** - запускается при открытии формы документа.

Модуль документа - запускается при проведении документа, удалении проведенного документа, при снятии проведения, при выполнении архивации записей журнала расчетов.

Модуль Формы журнала документов - запускается при вызове формы журнала документов.

Модуль Формы счёта - запускается при открытии формы бухгалтерского счёта.

Модуль Формы списка плана счетов - запускается при вызове формы списка плана счетов.

Модуль Формы Операции запускается при вызове формы бухгалтерской операции.

Модуль **Формы журнала операций** - запускается при вызове формы журнала операций.

Модуль Формы журнала проводок- запускается при вызове формы журнала проводок.

Модуль Формы журнала расчетов - запускается при вызове формы журнала

расчетов.

Модуль Формы вида расчета - запускается при расчете записей журнала расчетов.

Модуль Формы отчета или обработки - запускается при открытии диалоговой формы отчета или обработки.

5.2. Элементы встроенного языка

Процедуры и функции

Действия, заданные в диалоге, оформляются в модуле как *процедуры*. Процедурой является некоторый отдельный алгоритм, имеющий имя - *имя процедуры*. Процедура оформляется строкой начала процедуры и строкой конца процедуры.

```
Процедура МояПроцедура()
  <Тело процедуры>
КонецПроцедуры
```

Между этими строками располагается тело процедуры - алгоритм, описывающий действия, которые процедура будет выполнять. В процедуре можно определить список передаваемых параметров, значения которым передаются при вызове процедуры и используются в теле процедуры. По умолчанию параметру процедуры всегда передается ссылка на значение. Для передачи самого значения используется ключевое слово Знач.

- Передача параметров по ссылке

```
Процедура Моя( а )
                    а = 12;
КонецПроцедуры
Процедура А1(а)
                    а=10 ;
                    б = а ;
                    Моя(б);
КонецПроцедуры
```

В конце процедуры А1 переменная «б» равна 12, потому что и переменная «б», и параметр процедуры Моя «а» идентифицируют один и тот же адрес.

■ Передача параметров по значению

```
Процедура Моя(а)
        а=12;
КонецПроцедуры
Процедура А1(а)
        а=10 ;
        б = а ;
        Моя  ( Знач б);
КонецПроцедуры
```

В конце процедуры А1 переменная «б» равна 10 , потому что при вызове процедуры Моя() передавалась не ссылка, а значение.

Функция отличается тем, что возвращает значение, используя, оператор «Возврат».

```
Функция МояФункция()
        ..................
        Возврат Авс;
КонецФункции
```

Поэтому функции можно ставить в правой части операции присваивания и в поле «формула» элемент а диалоговой формы типа «Текст».

В языке также определено предварительное описание процедур и функций с помощью ключевого слова *Далее*

```
Процедура Моя(а) Далее
Процедура al(a)
        Моя(б) ;
КонецПроцедуры
Процедура Моя(а)
        а=12;
КонецПроцедуры
```

Переменная — это строка идентификатора, определяющая область оперативной памяти, в которую записываются значения объекта.

Переменные могут определяться явно, с помощью ключевого слова Перем перед именем переменной, или неявно при присваивании им значения.

Область действия переменной определяется контекстом, в котором она создана. Это может быть локальный контекст процедуры или функции,

```
Функция С1()
        Перем а, б, в;
        . . . . . .
        Г=а;  Возврат г;
КонецФункции
```

контекст модуля объекта или глобальный контекст.

```
Перем а,б,в;
Функция С1()
    . . . . . .
КонецФункции
```

Выражение — это текст на встроенном языке.

Оператор это логически завершенная последовательность выражений.

Операторы в тексте разделяются точкой с запятой. Отдельно стоящая точка с запятой воспринимается системой, как пустой оператор.

Для объединения частей выражения — операндов, используются операции. Операнды должны иметь тип, соответствующий операции (см. Таблицу). Порядок выполнения операций в выражении соответствует общепринятому, с учетом скобок и приоритетов операций.

Операция определенные во встроенном языке

Имя оператора	Обозн.	Приоритет	Операнды
Сложение	"+"	5	Число+Число, Дата+ Число(дней)
Вычитание	"-"	5	Число-Число,Дата-Число,Дата-Дата (Число дней между датами)
Умножение	*	6	Число*Число
Деление	/	6	Число/Число
Остаток	%	6	Число%Число (операнды от деления округляются до целого значения)
Конкатенация	+ (-)	5	Строка1+Строка2 («-» пробелы в конце Строки1 перемещаются в конец результата)
Больше	>	4	Число>Число, Строка>Строка, Дата > Дата
Больше или равно	>=	4	Число>=Число, Строка>=Строка, Дата>=Дата
Меньше	<	4	Число<Число, Строка<Строка, Дата<Дата
Меньше или равно	<=	4	Число<= Число, Строка<=Строка, Дата <= Дата
Равно	=	4	Число= Число, Строка=Строка, Дата= Дата, Значение объекта = Значение объекта
Не равно	<>	4	Число<> Число, Строка< >Строка (длина строки), Дата< >Дата, Значение объекта< >Значение объекта
Конъюнкция	И(AND)	2	(Значение условия) И (Значение условия)
Дизъюнкция	ИЛИ (OR)	1	(Значение условия) ИЛИ (Значение условия)
Отрицание	НЕ(NOT)	3	Не (Значение условия)

5.3. Правила адресации объектов

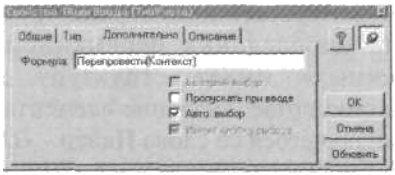
Переменные типа объект метаданных могут использоваться только с именами или ссылками на объекты метаданных или с их методами, между которыми ставится операция точка, имена видов объектов метаданных могут использоваться только с именами объектов метаданных: <Объект>.[<Атрибут>|<Метод()>]

```
Спр=Создать0бъект(«Справочник.Товары»);

Спр.НайтиПоКоду(1);

Спр.Цена.Получить('01.04.00');

Кнст=Константа.ОсновнойСклад;
```

Встроенный язык "1С:Предприятия" можно охарактеризовать как «язык с поздним связыванием», в котором связь между объектами формируется в момент выполнения кода программы, поэтому ошибки в программе, вызванные неправильным доступом к объектам проявляются только при запуске неправильного алгоритма из модуля соответствующего объекта.

5.4. Ввод формулы в поле ввода диалога формы

Для ввода формулы следует активи-зировать закладку «Дополнительные» палитры свойств, щелкнув мышкой в ее название.



Сама формула записывается в поле «Формула» с использованием синтаксиса языка "1С:Предприятие". Простейшим примером формулы является имя процедуры или функции, описанной в модуле: *При Выборе Группы()*. Для написания формулы можно также использовать *операцию присваивания*. Операция присваивания используется для занесения результатов расчета некоторого выражения в переменную или в элемент формы. Операция присваивания обозначается знаком «=». Справа от знака указывается выражение, а слева - куда заносить ре-зультат вычисления выражения.

```
Цена=Товар.Цена;
```

В поле «Формула» элемента формы типа поле ввода можно указать не одну, а несколько формул, разделяя их знаком «;». Эти формулы будут вычисляться последовательно слева направо.

В зависимости от типа поля вычисление выражения происходит следующим образом:

Для текстовых полей выражение вычисляется каждый раз, когда в форме происходит любое изменение. Поэтому поле должно содержать либо имя элемента данных (реквизита), либо вызов функции. В любом случае результат выражения преобразуется к тестовому представлению и отображается в поле элемента формы.

Для полей ввода формула вычисляется при потере фокуса на элементе интерактивной формы: то есть в двух случаях

1. пользователь изменил поле и нажал <Enter>;
2. пользователь «покинул» поле, используя <Tab>, <Shift+Tab> (а также акселераторы) или мышь.

5.5.Метод СоздатьОбъект и основные методы позиционирования объектов. Операторы передачи управления

Обратим внимание на метод СоздатьОбъект,

```
Об= СоздатьОбъект("<ИмяОбъектаМетаданных>");
```

который передает переменной «Об» ссылку на объект типа «ИмяОбъектаМетаданных» в оперативной памяти. Имя объекта метаданных должно в точности соответствовать идентификатору вида объекта **с учетом регистра букв**. После определения переменной «Об», если объект «ИмяОбъектаМетаданных» имеет структуру хранения, её надо спозиционировать на конкретное значение элемента данных, например, с помощью метода, начинающегося со слова Найти - *«Об.Найти***»,* который возвращает значение 1, если значение найдено, и 0 в противном случае. В результате исполнения этого метода происходит копирование значения элемента данных с дисковой памяти в оперативную.

Другие способы позиционирования объектов основаны на методах *Выбрать ***(),* который определяет множество значений поля выборки (записей в БД), и *Получить ***(),* возвращающем значение 1, если найдено следующее значение в поле выборки. Методом

```
Об.Выбрать***();
```

определим множество перебираемых значений, которое затем последовательно перебираем в цикле

```
Пока Об.Получить***()=1 Цикл

        <Тело цикла: операторы выполняющиеся
        периодически, пока верно условие цикла >

КонецЦикла;
```

Если значения атрибутов объекта были изменены, и им были присвоены некоторые значения.

```
Об.Атрибут=Знач_е;
```

Для сохранения новых значений атрибутов объекта его необходимо записать с помощью метода «Записать».

```
Об.Записать();
```

В языке описана ещё одна конструкция для организации цикла.

```
Для
ПараметрЦикла=НачальноеЗначениеПараметраЦикла
По КонечноеЗначениеПараметраЦикла Цикл
<Тело цикла: операторы выполняющиеся периодически>
КонецЦикла;
```

Передача управления по условию

```
Если <Условие> Тогда
            <Операторы, исполняемые если Условие верно
ИначеЕсли <Условие1>  Тогда
            < Операторы, исполняемые если Условие
            неверно, а Условие1 верно>
Иначе
            < Операторы, исполняемые если ни одно из
            условий неверно
КонецЕсли;
```

Конструкции «ИначеЕсли» и «Иначе» необязательны. Конструкций «ИначеЕсли» может быть несколько.

Безусловная передача управления на исполняемый оператор программного блока

Перейти <Метка>;

Условный оператор

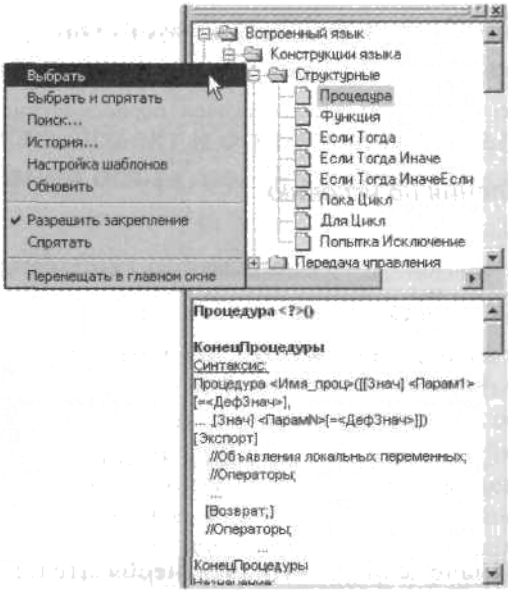
```
Переменная = ?(<Условие >,<Выражение, значение которого будет
присвоено переменной если условие выполняется>.< Выражение,
значение которого будет присвоено переменной если условие не_
выполняется >);
```

5.6. Отладка текстов программ

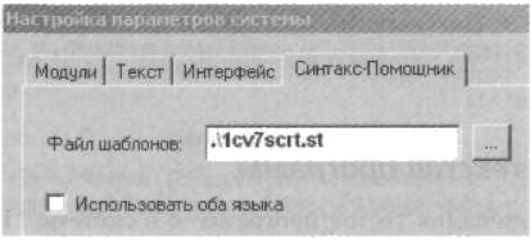
Для удобства написания тестов программ с в системе "1С:Предприятие"

существует «Синтакс-Помощник», в котором описаны все методы и атрибуты видов метаданных, а также основные конструкции языка. В нем можно использовать как русскоязычные, так англоязычные варианты имен. Тексты из Синтакс-Помощника можно подсмотреть в файлах /lCv77/BIN/lcv7Lang.als и lcv7tOpr.als.

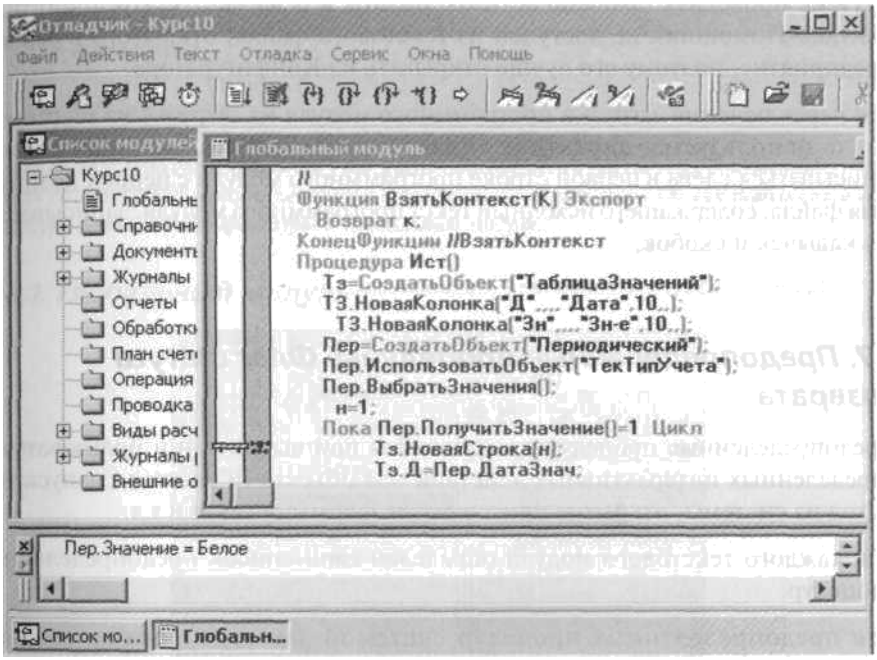
Предусмотрены методы вставки конструкций языка из Синтакс-Помощника в текстовые модули. Панель свойств Синтакс-Помощника, как обычно, вызывается щелчком правой кнопки мыши.



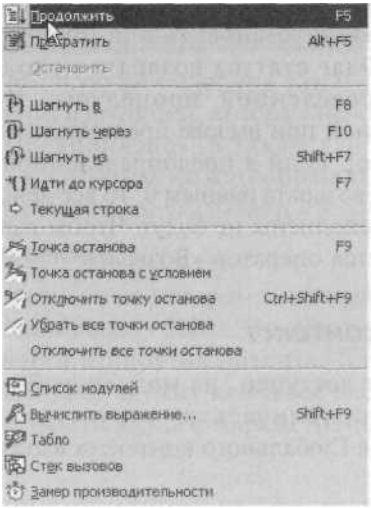
Синтакс-Помощник позволяет также настраивать собственные шаблоны пользователя, которые записываются в файл с расширением *.st. В типовых конфигурациях это файл 1cv7scrt.st. Этот файл можно присоединить к конфигурации через меню «Сервис» - «Параметры», закладка «Синтакс-Помощник».



Отладка текстов программ производится с помощью «Отладчика», в котором доступны: стек вызова исполняемых процедур и функций, пошаговое исполнение алгоритма вычислений, возможность контроля вычислений, а также замеры производительности.



Список действий меню «Отладка» приводится на следующем рисунке.



К сожалению, «Отладчик» не позволяет редактировать алгоритмы, поэтому отладка алгоритмов часто производится во внешнем текстовом файле (при запущенном Конфигураторе), в который переносится весь текст модуля объекта. Этот прием позволяет отредактировать отдельный текстовый блок, и тут же проверить правильность его работы в режиме "1С:Предприятия". Синтакс-Помошник не доступен из текстового файла, открытого в режиме Предприятие, поэтому его лучше открывать из Конфигуратора.

Для переключения загрузки программного модуля на загрузку из текстового файла используется директива «#ЗагрузитьИзФайла», которая должна записываться слева в первой строке программного модуля с первой позиции. Имя файла, содержащего исходный текст программного модуля, записывается без кавычек и скобок.

```
Пример: #ЗагрузитьИзФайла textMod.txt
```

5.7. Предопределенные процедуры. Флаг статуса возврата

Предопределенные процедуры вызываются при выполнении пользователем определенных интерактивных действий над объектами системы: запуск или выход из системы, открытие или закрытие формы объекта и т.д.

Для каждого текстового модуля определен свой список предопределенных процедур.

Для предопределенных процедур системой зарезервированы имена и определены списки параметров.

В предопределенных процедурах определен **флаг статуса возврата,** текущее значение которого можно установить или прочитать с помощью метода *СтатусВозврата()*. Флаг статуса возврата используется системой при завершении предопределенной процедуры. Начальное значение, устанавливаемое системой при вызове предопределенной процедуры, равно 1 (выполнить действие). Если в предопределенной процедуре установить значение флага статуса возврата равным 0, то все действия, предусмотренные данной процедурой, выполнены не будут. Чтобы выйти из процедуры до её завершения, используется оператор «Возврат».

5.8. Глобальный контекст

Глобальный контекст доступен из модуля любого объекта и позволяет оптимизировать алгоритмическую структуру и уменьшить объём конфигурации. В состав Глобального контекста входят:

Глобальный модуль

Общие таблицы

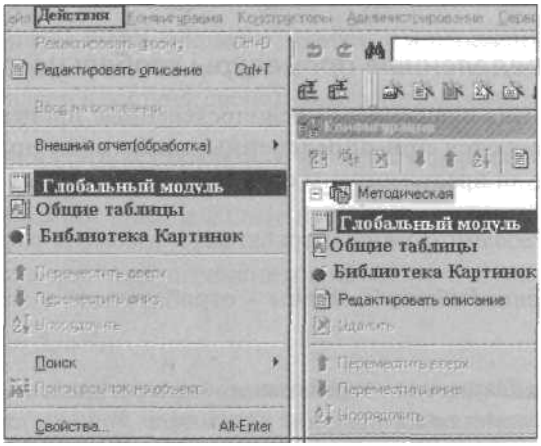
Библиотека картинок

Константы Перечисления

Группы расчетов

Предположим, что счет-фактура у нас распечатывается из различных документов, тогда можно создать один шаблон печатной формы счета-фактуры в Общих таблицах и одну процедуру печати счет-фактуры в Глобальном модуле.

5.8.1. Глобальный модуль

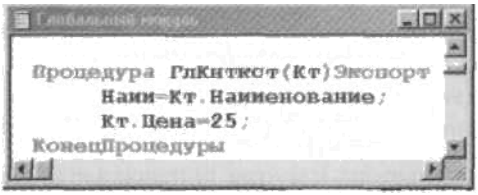


Глобальный модуль имеет такую же структуру, как и любой другой модуль. В нём помещаются процедуры и функции, которые необходимо выполнять при запуске системы «1С:Предприятие», а также глобальные переменные, процедуры и функции, к которым можно обращаться из модуля любого объекта. Для этого при их описании указывается ключевое слово *Экспорт.*

Передача контекста объекта

Контекст - это элемент данных, на котором в данный момент интерактивно или программно спозиционирован объект. По значению контекста доступна вся совокупность значений атрибутов объекта. Для передачи контекста объекта в Глобальный контекст служит ключевое слово *Контекст,* которое используется в качестве фактического параметра при вызове процедур и

функций Глобального модуля, в описании которых есть ключевое слово ***Экспорт.***



Различают контекст объекта и контекст интерактивной формы объекта. Контекст формы можно передавать через параметр КонтФормы в формы других объектов, открываемые методами ОткрытьФорму() и ОткрытьПодбор(), а также через параметр предопределенной процедуры ОбработкаПодбора(). Этот параметр позволяет управлять реквизитами первой формы, из которой была открыта вторая форма, в контексте второй формы.

5.8.2. Предопределенные процедуры глобального модуля

Для каждого модуля существуют предопределенные процедуры (действия), которые выполняются при определенных действиях пользователя. Для глобального модуля предусмотрены следующие процедуры:

ПриНачалеРаботыСистемы - отрабатывает в момент загрузки программы

ПриЗавершенииРаботыСистемы - отрабатывает в момент закрытия программы

ПриУдалении Документа - отрабатывает в момент удаления документов или постановки пометки на удаление документа

ПриУдалении Элемента - отрабатывает в момент удаления или постановки пометки на удаление элемента справочника

ПриЗаписи Истории - отрабатывает в момент записи истории значения периодического элемента

ПриУдалении Истории - отрабатывает в момент удаления из списка истории значения периодического элемента

ПриЗаписи Константы - отрабатывает в момент записи значения константы

ПриОтмене ПроведенияДокумента - отрабатывает в момент отмены проведения документа

ПриИзмененииВремениДокумента - отрабатывает в момент изменения времени существующего документа

ПриУстановкеОтбора - отрабатывает в момент установления отбора

ПриСменеРасчетного Периода - отрабатывает в момент смены расчетного периода журналов расчетов.

Вопросы для самоконтроля Что

такое контекст?

Чем отличаются процедуры и функции?

Как определяются переменные во встроенном языке «1С:Предприятия»?

Как производится вызов процедур, функций и переменных модуля?

Как осуществляется доступ к свойствам агрегатных объектов метаданных?

Чем отличается контекст объекта от контекста формы объекта?

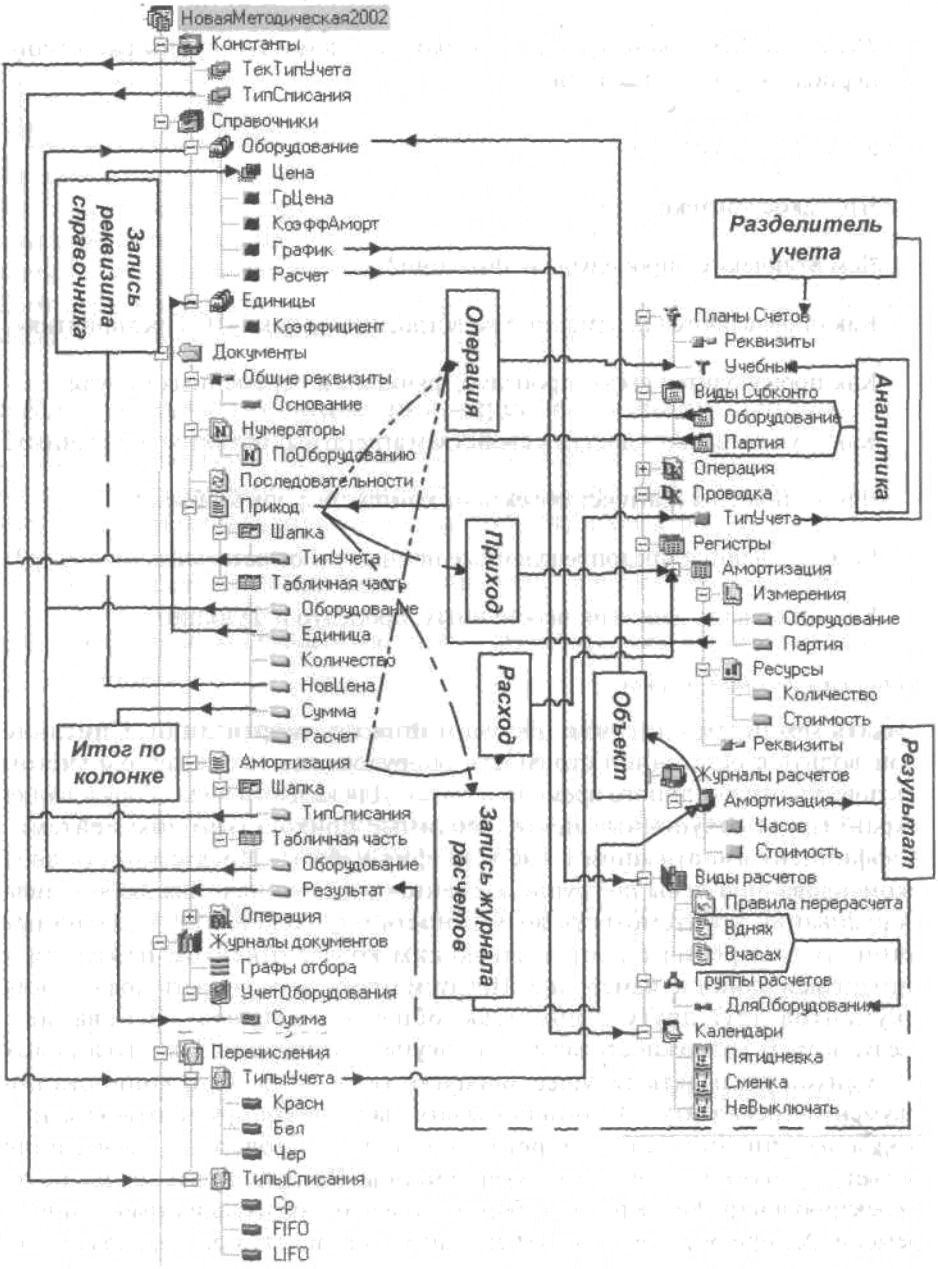
Чем отличаются предопределенные процедуры от остальных процедур?

Какова область действия переменных, процедур и функций?

Постановка учебной задачи

Создать механизм списания оборудования по амортизации. Списание производить с остаточной стоимости оборудования ежемесячно с учетом фактически отработанного времени в часах. Для каждого вида оборудования сохранять все закупочные цены, вводимые приходными документами, коэффициент амортизации в час и график работы. Предусмотреть ввод рекомендованной цены по группе элементов справочника. Для любого вида оборудования предусмотреть возможность оприходования в различных единицах измерения с соответствующим коэффициентом пересчета к наименьшей единице измерения. Предусмотреть возможность подчинения Документов друг другу с помощью общего реквизита «Основание». Предусмотреть возможность задавать в документах тип учета. При вводе новых Документов указывать текущее значение типа учета. При копировании Документов, реквизиту «Основание» задать значение копируемого документа. Документ списания сделать регламентным, с проверкой на наличие соответствующих документов в текущем месяце. Предусмотреть возможность корректировки графиков работы оборудования по фактически отработанному времени. Алгоритм расчета суммы списания определять при оприходовании

новой партии оборудования. Создать движения по регистру остатков и по бухгалтерским счетам с аналитикой по оборудованию и по приходным партиям.


НоваяМетодическая2002
Константы
ТекТипУчета
ТипСписания
Справочники
Оборудование
Цена
ГрЦена
КоэффАморт
График
Расчет
Единицы
Коэффициент
Документы
Общие реквизиты
Основание
Нумераторы
ПоОборудованию
Последовательности
Приход
Шапка
ТипУчета
Табличная часть
Оборудование
Единица
Количество
НовЦена
Сумма
Расчет
Амортизация
Шапка
ТипСписания
Табличная часть
Оборудование
Результат
Операция
Журналы документов
Графы отбора
УчетОборудования
Сумма
Перечисления
ТипыУчета
Красн
Бел
Чер
ТипыСписания
Ср
FIFO
LIFO
Запись реквизита справочника
Итог по колонке
Операция
Приход
Расход
Запись журнала расчетов
Объект
Разделитель учета
Планы Счетов
Реквизиты
Учебный
Виды Субконто
Оборудование
Партия
Операция
Проводка
ТипУчета
Регистры
Амортизация
Измерения
Оборудование
Партия
Ресурсы
Количество
Стоимость
Реквизиты
Журналы расчетов
Амортизация
Часов
Стоимость
Виды расчетов
Правила перерасчета
Вднях
Вчасах
Группы расчетов
ДляОборудования
Календари
Пятидневка
Сменка
НеВыключать
Аналитика
Результат

VI. Пример создания простой реляционной структуры

Каждый объект содержит набор данных со своими методами доступа и обработки.

6.1. Перечисления

Перечисления являются внутренними константами, присущими данной конфигурации. Они записываются не в отдельную таблицу, а непосредственно в файл конфигурации 1cv7.md и могут редактироваться только в конфигураторе. Это не агрегатные объекты, потому что содержат статичный (неизменяемый) набор значений. Они предназначены для создания некоторого постоянного списка возможных значений для выбора. Перечисления удобны в тех случаях, когда возникает необходимость в ограничении количества возможных вариантов значений константы, реквизита справочника или документа и т. д.

Пример выбора значения перечисления по его номеру:

```
Спис=Создать0бъект(«СписокЗначений»);
Всего = Перечисление.ТипыУчета.КоличествоЗначений();
Для Ном = 1 По Всего Цикл
  Спис.ДобавитьЗначение.(Перечисление.ТипыУчета.ЗначениеПоНомеру (Ном)
        , Перечисление.ТипыУчета.Идентификатор());
КонецЦикла;
```

Служебный объект метаданных типа «СписокЗначений» позволяет работать с одномерными массивами , в том числе создавать список с пометками и с интерактивным представлением объекта, поэтому при записи значения в список надо задать строковое представление значения для возможности интерактивной работы со списком значений.

Метод **ДобавитьЗначение**(<Значение>,<Строка>)
Добавляет значение в список значений.
<Значение> - значение, которое добавляется в список;
<Строка> - символьное представление значения (необязателен, по умолчанию - стандартное символьное представление объекта).

6.2.Константы

Константы - предназначены для хранения информации, относящейся ко всему предприятию, необходимой для многократного использования в печатных Формах, расчетах и т. д. Информация может быть постоянной или

периодически меняющейся. Использование констант облегчает изменение величин, которые используются в разных местах конфигурации.

Константы могут сохранять историю изменения значения по датам. Для этого необходимо на закладке «Дополнительные» панели свойств константы установить флажок «Периодический».

Для работы с периодическими значениями используются методы Получить() и Установить().

Получить(<Дата>)возвращает значение периодической константы на заданную дату. Параметры:<Дата> - необязательный параметр. Выражение типа дата или значение типа документ или позиция документа. Этот параметр задает момент времени, на который требуется получить значение периодической константы. Значение по умолчанию: ТА - если используется компонента "Оперативный учет", Рабочая дата - если компонента "Оперативный учет" не используется.

Установить(<Дата>,<Значение>)- установить значение периодической константы на дату. Параметры:<Дата> - дата, на которую требуется установить значение периодической константы; <Значение> - новое значение константы.

Примеры записи значения периодической константы:

```
Константа.ТипУчета.Установить(Дата_зн, зн_е);
```

Используется так же **служебный объект Периодический,** предназначенный для работы с периодическими объектами, и имеющий атрибуты Значение и ДатаЗнач для чтения и редактирования истории изменения их значению.

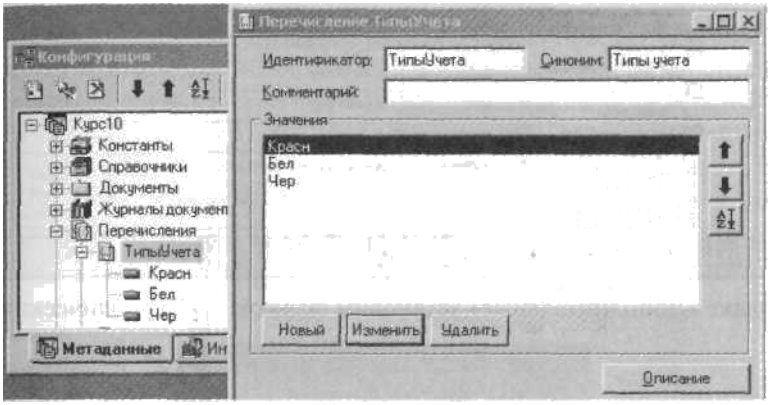
Метод объекта периодический **ИспользоватьОбъект**(<ИмяРеквизита>,<Объект>) сопоставляет объект типа 'Периодический' периодической константе или периодическому реквизиту справочника. Возвращает: 1 - если вызов метода закончился успешно, 0 - иначе.
<ИмяРеквизита> - строка с названием периодического реквизита справочника или периодической константы;
<Объект> - значение элемента справочника, для которого задается применение периодического реквизита (для констант не нужен).
Если наименование реквизита пустая строка и передан объект типа справочник, то выборка будет осуществляться по всем реквизитам справочника.

```
ТипУч = Константа.ТипУчета;

ТипУч = СоздатьОбъект(«Периодический») ;

ТипУч.ИспользоватьОбъект(«ТипУчета»);

ТипУч.Значение= зн_е;

ТипУч.ДатаЗнач = Дата_зн;

ТипУч.Записать();
```

Упражнение 1. Создайте перечисление «ТипыУчета», задайте несколько значений типов учета, не забудьте определить пользовательские представления значений, необходимые для интерактивного выбора значения типа перечисление.

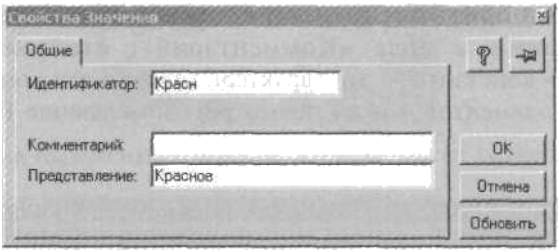
Встаньте на ветвь метаданных «Перечисления», по правой кнопке мыши выберите пункт контекстного меню «Новое перечисление». В появившемся

окне с помощью кнопки «Новый» введите возможные значения созданного вами перечисления. В поле «Идентификатор» укажите имя, по которому вы будете обращаться к данному значению программно из текстовых модулей.



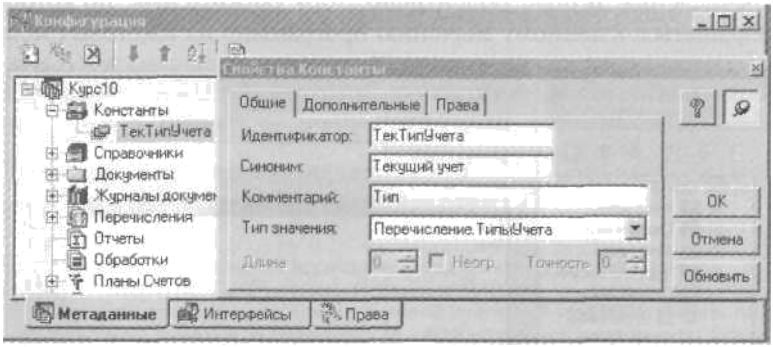
«Идентификатор» - это строка, начинающаяся с буквы или знака подчеркивания и не содержащая пробелов и служебных символов.

В поле «Представление» укажите пользовательское представление данного значения.

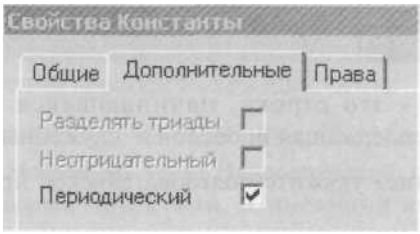


Упражнение 2. Создайте периодическую константу «ТекТипУчета» типа Перечисление.ТипыУчета. Сохраните изменения, запустите режим Предприятие и, используя меню Операции, введите значения константы на разные даты.

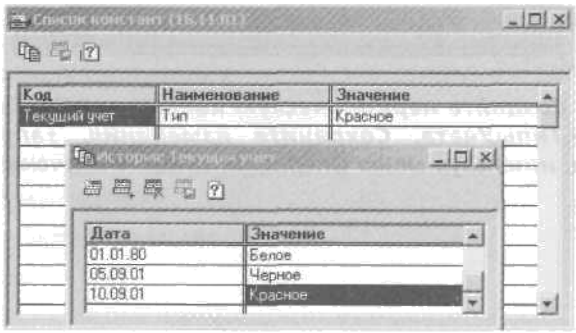
В Конфигураторе:



На закладке «Дополнительные» установим флажок — «Периодический».



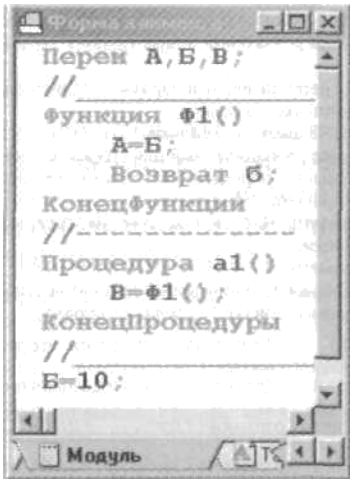
В Предприятии обратите внимание, что строка, записанная при конфигурировании в поле «Комментарий», отображается в поле «Наименование» константы — это характерно только для констант. Историю периодических элементов можно посмотреть по клавише F5 или щелкнув иконку 



6.3. Создание программного модуля

6.3.1. Структура программного модуля

Прогаммный модуль всегда привязан к какому-либо объекту системы, чаще к интерактивной форме. Трансляция модуля в исполняемый код производится за один проход, поэтому важна последовательность описания переменных и процедур модуля. Его можно поделить на три части: Блок описания переменных модуля, Блок описания процедур и функций модуля и Раздел основной программы модуля.



Блок определения переменных модуля. Отделяется от остального текста точкой с запятой.

Блок описаний процедур и функций модуля. Если в теле одной процедуры вызывается другая процедура, то последняя должна быть описана выше по тексту (например: с помощью ключевого слова Далее после описания процедуры).

Раздел основной программы -содержит отдельные операторы, выполняющиеся До открытия интерактивной формы объекта

Упражнение 3(Необязательное) Создайте процедуры в глобальном модуле, которые позволят нам редактировать любые периодические константы с помощью служебных объектов «ТаблицаЗначений», «Периодический» и "Метаданные». В теле процедуры опишем переменные для объектов ТаблицаЗначений и Периодический.

Служебный объект ТаблицаЗначений - позволяет создавать, редактировать и

выбирать значения из двумерных массивов.

Служебный объект Метаданные - позволяет получить доступ к любому объекту метаданных даже в неизвестной конфигурации. Он является атрибутом Глобального контекста и для него зарезервировано ключевое слово «Метаданные», поэтому его не надо создавать с помощью метода СоздатьОбъект.

1 этап.

Создадим в глобальном модуле процедуру, допустим, История, в которой определим объект типа таблица значений и опишем для него две колонки для записи даты и значения. Для колонок определим идентификаторы и ширину.

Метод **НоваяКолонка** (<Идентификатор>,<Тип>,<Длина>,<Точность>,<Заголовок>,<Ширина>,<Формат>, <Положение>) присоединяет к таблице значений новую колонку справа и возвращает номер новой колонки.
Параметры: <Идентификатор> - необязательный параметр. Идентификатор колонки. Если не указан, то обращение к колонке возможно только по номеру.
<Тип> - необязательный параметр. Задает тип колонки. **<Длина>** - необязательный параметр, определенный для строки или числа. **<Точность>** - необязательный параметр, определенный только для числа. **<Заголовок>** - необязательный параметр.
<Ширина> - необязательный параметр. Числовое выражение, содержащее ширину колонки (в символах) для представления колонки. **<формат>** - необязательный параметр.
<Положение> - необязательный параметр. Определяет вариант выравнивания при визуальном отображении значений данной колонки. Число: 1 - слева; 2 - справа.

```
Процедура История()

    // Просмотр истории периодических объектов

    ТаблицаЗначений=Создать0бъект("ТаблицаЗначений");

    ТаблицаЗначений.НоваяКолонка("Дт",,,,"Дата",11);

    ТаблицаЗначений.НоваяКолонка("Зн",,,,"Значение",15);

КонецПроцедуры
```

Создадим объект Периодический, определим, какие периодические константы есть в конфигурации, спозиционируем объект Периодический в цикле на текущую константу, если она периодическая.

Метод **Выбран()** проверяет спозиционирован ли объект типа "Метаданные" на конкретном объекте метаданных или нет. Возвращает число: 1 - объект соответствует объекту метаданных (спозиционирован), 0 - если не соответствует.

```
Процедура История()
  // Просмотр истории периодических объектов
  ТаблицаЗначений=СоздатьОбъект("ТаблицаЗначений");
  ТаблицаЗначений.НоваяКолонка("Дт",,,,"Дата",11);
  ТаблицаЗначений.НоваяКолонка("Зн",,,,"Значение" , 15) ;
  //Найдем периодические Объекты
  Периодический=Создать0бъект("Периодический");
  // По константам
```

```
i = 1;

Пока Метаданные.Константа(i).Выбран()=1 Цикл
```

```
        //Получим "Имя Константы"  в переменной Описатель
        Описатель=Метаданные.Константа(i).Идентификатор;
        Если Периодический.ИспользоватьОбъект(Описатель)=1 Тогда
             // Заполним таблицу значений периодическими значениями
        КонецЕсли;
        i=i+1;
    КонецЦикла;
    КонецПроцедуры
```

2 этап.

Создадим процедуру для заполнения таблицы значений периодическими значениями. Запишем получаемые в цикле значения и даты значений периодической константы в таблицу значений, используя атрибуты служебных объектов Периодический (Значение и ДатаЗнач) и ТаблицаЗначений (НомерСтроки и <ИдентификаторКолонки>).

Метод **ВыбратьЗначения**(<ДатаНачала>,<ДатаКонца>) открывает выборку периодических значений по датам и возвращает: 1 - если вызов метода закончился успешно, 0 - иначе. Если даты не заданы, то выбираются все значения.
Метод **ПолучитьЗначение**() позволяет получить из выборки следующий элемент и возвращает: 1 - если элемент выбран, 0 - иначе.
Метод периодического объекта **ТекущийРеквизит**() возвращает наименование константы или текущего реквизита справочника. Метод используется после получения очередного значения из выборки.
Метод **НоваяСтрока**(<НомерСтроки>) добавляет новую строку в таблицу значений.
Параметры:<НомерСтроки> - необязательный параметр. Числовое выражение, содержащее позицию, в которую следует вставить новую строку.

В этой процедуре определим два параметра: откуда читаем «П», куда пишем «Тз».

```
Процедура ЗаполнениеТаблицы(Тз,П)
П.ВыбратьЗначения();
Пока П.Получитьзначение()=1 Цикл
  Тз.НоваяСтрока();
  Тз.Дт=П.ДатаЗнач;
  Тз.Зн=П.Значение;
  Тз.Пар=П.ТекущийРеквизит();
КонецЦикла;
КонецПроцедуры //ЗаполнениеТаблицы
```

Организуем вызов этой процедуры из процедуры История, отсортируем полученную таблицу значений по дате и вызовем процедуру

РедактированиеИстории, в которой будет описан алгоритм просмотра и редактирования периодических элементов.

Метод таблицы значений **Сортировать**(<Колонки>,<ДокумПоДате>) позволяет сортировать таблицу значений по колонкам.
Параметры:
<Колонки> - строковое выражение, которое определяет колонки, порядок и направление сортировки. Формат передаваемой строки - это разделенные запятыми номера или идентификаторы колонок со знаком направления сортировки ("+" - сортировать по возрастанию; "-" - сортировать по убыванию; "*" - сортировать по внутреннему значению). Знак направления сортировки можно указывать до или после обозначения колонки через пробел или без пробела. По умолчанию направление сортировки принимается по возрастанию.
<ДокумПоДате> - необязательный параметр. Имеет смысл только в том случае, если значениями таблицы значений являются документы. В этом случае можно задавать сортировку документов по их хронологии. Число: 1 - сортировка по хронологии документов; 0 - нет. Значение по умолчанию - 0.

```
Процедура История()    // Просмотр истории периодических объектов
    ТаблицаЗначений=Создать0бъект("ТаблицаЗначений");
    ТаблицаЗначений.НоваяКолонка("Дт",,,,"Дата",11);
    ТаблицаЗначений.НоваяКолонка("Зн",,,,"Значение",15);
    ТаблицаЗначений.НоваяКолонка("Пар",,,,"Параметр",15);
    //Найдем периодические Объекты
    Периодический=Создать0бъект("Периодический");
    // По константам
    i = 1;
    Пока Метаданные.Константа(i).Выбран()=1 Цикл
        //Получим "Имя Константы" в переменной Описатель
        Описатель=Метаданные.Константа(i).Идентификатор;
        Если Периодический.ИспользоватьОбъект(Описатель)=1 Тогда
          // Заполним таблицу значений периодическими значениями
          ЗаполнениеТаблицы(ТаблицаЗначений,Периодический);
        КонецЕсли;
        i=i+1;
    КонецЦикла;
    ТаблицаЗначений.Сортировать("Дт");
    РедактированиеИстории(Периодический,ТаблицаЗначений);
КонецПроцедуры
```

3 этап.

В процедуре РедактированиеИстории просмотрим записанные значения в отдельном окне, с помощью метода таблицы значений ВыбратьСтроку(), и

создадим СписокЗначений для того, чтобы можно было выбрать действие над значением из таблицы значений.

Выбрать**Строку**(<Строка>,<Заголовок>,<Таймаут>). Метод открывает окно для интерактивного выбора строки в таблице значений и возвращает число: 1 - если выбор произведен (нажата кнопка ОК); 0 - если выбор не произведен (нажата кнопка "ОТМЕНА"); -1 (минус единица) - закончилось время <Таймаут> ожидания отклика пользователя.

Не забудем, что параметры методов, которым передается какое-либо значение при вызове метода, должны быть предварительно явно или неявно описаны; в данном случае это параметр <Строка>. Далее получим строку из таблицы значений и спозиционируем объект периодический на константу и на выбранное значение константы.

Метод таблицы значений **ПолучитьСтрокуПоНомеру**(<НомерСтроки>) позволяет получить строку таблицы значений по номеру. Указанная строка становится текущей. <НомерСтроки> - номер строки, на которую следует переместиться.

Метод объекта периодический **НайтиЗначение**(<Дата>,<Режим>) находит периодическое значение на заданную дату.Возвращает: 1 - если вызов метода закончился успешно, 0 -иначе. Параметры:

<Дата> - дата, на которую требуется найти периодический реквизит справочника или периодическую константу;

<Режим> - режим поиска на тот случай, когда на заданную дату не существуе| значения периодического реквизита: -1, возвращается значение на предыдущую дату;

0, возвращается код завершения неуспешной операции;

1, возвращается значение на последующую дату.

```
Процедура   РедактированиеИстории(Периодический,Таблица)

  Перем НомерСтр;

  Если  Таблица. ВыбратьСтроку (НомерСтр/История
                                периодических объектов")=1 Тогда

  Таблица. ПолучитьСтрокуПоНомеру (НомерСтр) ;

  Периодический. ИспользоватьОбъект (Строка (Таблица. Пар) ) ;

  Периодический.НайтиЗначение(Таблица. Дт, 0 ) ; КонецЕсли;

КонецПроцедуры
```

4 этап.

Определим, что мы будем делать с найденной записью, запишем эти действия в СписокЗначений, организуем выбор действия из списка и выполним выбранное действие для периодического значения, После изменения значения, чтобы увидеть это изменение, перезаполним ТаблицуЗначений., вызвав процедуру История().

Метод списка значений

Выбрать**Значение**(<Значение>,<Заголовок>,<Позиция>,<Таймаут>,<Способ выбора>) позволяет открыть окно для интерактивного выбора значения из списка.

Возвращает: -1 (минус единица) - закончилось время <Таймаут> ожидания отклика пользователя, 1 -если выбор произведен, 0 - иначе. Параметры: <Значение> - идентификатор переменной, куда помещается результат выбора;

<Заголовок> - строка заголовка диалогового окна.
<Позиция> - идентификатор переменной, куда помещается номер позиции выбранного значения в списке.
<Таймаут> - необязательный параметр. Числовое выражение, значение которого задает время ожидания системы (в секундах) на отклик пользователя.
<Способ выбора> - необязательный параметр. Число, значение которого задает способ выбора значения. О - в виде диалога; 1 - выбор производится в виде меню, которое подстраивается по месту текущего элемента диалога или ячейки таблицы; 2 - выбор маленьким списком (список похож на выбор значения перечисления), также привязанным к позиции элемента диалога
Значение по умолчанию - 0.
Метод может использоваться только для переменных, созданных функцией СоздатьОбъект.

```
Процедура РедактированиеИстории(Периодический,Таблица)

    Перем НомерСтр, Значение, Позиция;

    СписокДействий=Создать0бъект("СписокЗначений");

    СписокДействий.ДобавитьЗначение("У","Удалить");

    СписокДействий.ДобавитьЗначение("И","Изменить");

    Если Таблица.ВыбратьСтроку(НомерСтр,"История периодических
                                                объектов")=1 Тогда

        Таблица.ПолучитьСтрокуПоНомеру(НомерСтр);

        Периодический.ИспользоватьОбъект(Строка(Таблица.Пар));

        Периодический.НайтиЗначение(Таблица.Дт,0);

        Если СписокДействий.ВыбратьЗначение(Значение,"Выберите
        действие",Позиция,,1)=1 Тогда

            Если Значение="У" Тогда

                Периодический.Удалить();

            ИначеЕсли Значение="И" Тогда

                Стр=Таблица.Зн;
                ДатаЗначения=Периодический.ДатаЗнач;

                Если ВвестиПеречисление(Стр,"Введите
                значение ")<>1 Тогда

                    Возврат;

                КонецЕсли;

                Периодический.ДатаЗнач=ДатаЗначения;

                Периодический.Значение=Стр;

                Периодический.Записать();

            КонецЕсли;

        КонецЕсли;
```

```
            История();
        КонецЕсли;
КонецПроцедуры
```

Мы видим, что у нас процедура РедактированиеИстории вызывается из процедуры История, и, наоборот, процедура История вызывается из процедуры РедактированиеИстории. Поэтому одну из них надо описать перед описанием другой с помощью ключевого слова Далее. Окончательный текст блока будет следующим.

```
Процедура РедактированиеИстории(Периодический,Таблица) Далее
Процедура ЗаполнениеТаблицы(Тз,П)
    П.ВыбратьЗначения();
    Пока П.Получитьзначение()=1 Цикл
        Тз.НоваяСтрока();
        Тз.Дт=П.ДатаЗнач;
        Тз.Зн=П.Значение;
        Тз.Пар=П.ТекущийРеквизит();
    КонецЦикла;
КонецПроцедуры //ЗаполнениеТаблицы

Процедура История() // Просмотр истории периодических объектов
     ТаблицаЗначений=Создать0бъект("ТаблицаЗначений");
     ТаблицаЗначений.НоваяКолонка{"Дт",,,,"Дата",11);
     ТаблицаЗначений.НоваяКолонка("Зн",,,,"Значение",15);
     ТаблицаЗначений.НоваяКолонка("Пар",,,,"Параметр",15);
     //Найдем периодические Объекты
     Периодический=СоздатьОбъект("Периодический");
     // По константам
     i = 1;
     Пока Метаданные.Константа(i).Выбран()=1 Цикл
          //Получим "Имя Константы" в переменной Описатель
          Описатель=Метаданные.Константа(i).Идентификатор;
          Если Периодический.ИспользоватьОбъект(Описатель)=1 Тогда
             // Заполним таблицу значений периодическими
             значениями
```

```
                ЗаполнениеТаблицы(ТаблицаЗначений,Периодический);
            КонецЕсли;
            i=i+1;
        КонецЦикла;
        ТаблицаЗначений.Сортировать("Дт");
        РедактированиеИстории(Периодический,ТаблицаЗначений);
КонецПроцедуры

Процедура РедактированиеИстории(Периодический,Таблица)
    Перем НомерСтр,Значение,Позиция;
    СписокДействий=СоздатьОбъект("СписокЗначений");
    СписокДействий.ДобавитьЗначение("У","Удалить");
    СписокДействий.ДобавитьЗначение("И","Изменить");
    Если Таблица.ВыбратьСтроку(НомерСтр,"История периодических
    объектов")=1 Тогда
        Таблица.ПолучитьСтрокуПоНомеру(НомерСтр);
        Периодический.ИспользоватьОбъект(Строка(Таблица.Пар));
        Периодический.НайтиЗначение(Таблица.Дт,0);
        Если СписокДействий.ВыбратьЗначение(Значение,"Выберите
        действие".Позиция,,1)=1 Тогда
            Если Значение="У" Тогда
                Периодический.Удалить();
            ИначеЕсли Значение="И" Тогда
                Стр=Таблица.Зн;
                ДатаЗначения=Периодический.ДатаЗнач;
                Если ВвестиПеречисление(Стр,"Введите значение
                ")<>1 Тогда
                    Возврат;
                КонецЕсли;
                Периодический.ДатаЗнач=ДатаЗначения;
                Периодический.Значение=Стр;
                Периодический.Записать();
            КонецЕсли;
        КонецЕсли;
```

```
        История();
    КонецЕсли;
КонецПроцедуры
```

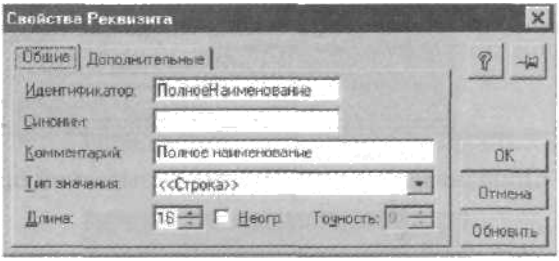
Упражнение 4. Вставьте в меню пользовательского интерфейса вызов процедуры История() глобального модуля.

Чтобы процедура История() когда-нибудь выполнилась её надо вызвать.

Для редактирования интерфейса в конфигураторе необходимо открыть окно «Конфигурация» и перейти к закладке «Интерфейсы», для этого достаточно просто щелкнуть мышкой по этой закладке. На экране появится список пользовательских интерфейсов. Каждый интерфейс - это совокупность меню и набора панелей инструментов. Для создания пункта меню, перейдём в режим редактирования меню интерфейса.

Для редактирования пользовательских интерфейсов в Конфигураторе существует 2 специализированных редактора: редактор меню и редактор панелей инструментов. Для вызова редактора меню необходимо:

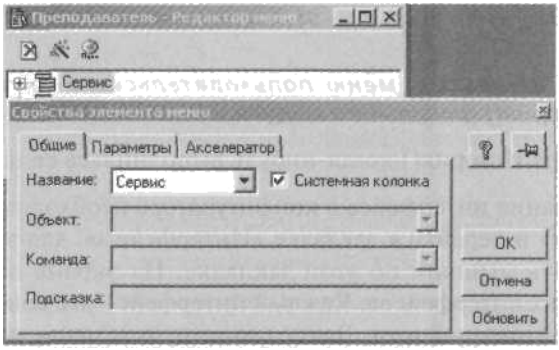
- щелкнуть правой кнопкой мыши на строке с наименованием интерфейса, который требуется отредактировать (у нас в конфигурации этот интерфейс - единственный);
- из контекстного меню выбрать строку «Редактировать меню...».



В результате этих действий будет открыто окно - «Редактор меню». В котором меню пользовательского интерфейса выводится не в том виде, в котором мы с ним работаем в «1С:Предприятии», то есть не в виде колонок, а в виде дерева, при этом ветви верхнего уровня соответствуют колонкам.

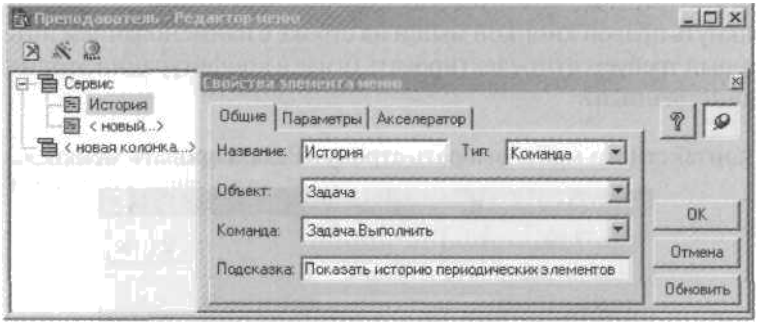
На платформе V77 можно не только создавать новые пункты меню, но и Редактировать системные пункты меню.

• Откроем свойства элемента меню <новая колонка...> через контекстное меню и отредактируем его.

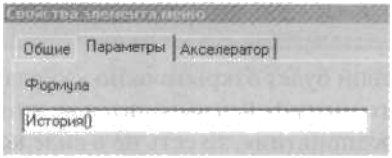


• Щелкнем крестик слева от пункта меню «Сервис» и откроем двойным щелчком мыши свойства пункта меню <новый...>.

• выберем команду «Задача .Выполнить»;

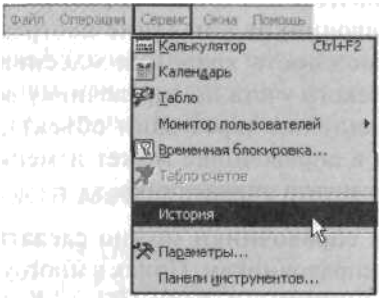


• На закладке «Дополнительно» укажем имя вызываемой процедуры.



Можно ещё отредактировать в палитре свойств поля «Название» и «Подсказка», автоматически заполненные программой, так, чтобы они более наглядно и естественно отражали назначение команды меню. Содержимое поля «Название» будет являться собственно названием команды меню, а содержимое поля «Подсказка» будет выводиться в нижней части экрана *(статус-строке)* при установке курсора на данный пункт меню. Теперь если нажать кнопку «ОК» - введенная нами команда попадет в нижнюю часть ветви «Сервис». Можно перетащить ее в другое место дерева меню.

Сохраним сделанные изменения и запустим режим Предприятие. Запустим процедуру, выбрав созданный пункт интерфейса.



Упражнение 3.1 (дополнительное). Самостоятельно, используя Синтакс-Помошник, отредактируйте алгоритм, описанный в упражнении 3 таким образом, чтобы с помощью него можно было добавить значение в историю периодического элемента.

Упражнение 3.2 (дополнительное). Самостоятельно отредактируйте алгоритм, описанный в упражнении 3.1 таким образом, чтобы с помощью него можно было изменить или значение или дату значения периодического элемента.

Вопросы для самоконтроля

Как создаются новые типы объектов на платформе «1С:Предприятие»?

Почему значения перечислений можно редактировать только в конфигураторе?

Чем отличаются периодические константы от простых констант?

VII. Справочники

Справочники - предназначены для хранения различных списков, перечней и т. д. Достоинство справочников - наличие неограниченного количества реквизитов (т. е. возможность хранения массива информации, а при организации аналитического учета по справочнику возможность получения широкого спектра аналитики через один объект), наличие нескольких уровней. Информация в справочнике может изменяться, а также хранить историю изменения реквизита справочника.

Для удобства выборки справочники можно сделать многоуровенными и подчиненными другим справочникам. Поиск в многоуровенном справочнике осуществляется либо по полному коду *ПолныйКод()* или наименованию *ПолноеНаименование(),* либо по атрибуту Родитель. В многоуровенном справочнике одному значению родительской группы соотносится несколько элементов или подгрупп справочника. Аналогично одному элементу справочника-владельца сопоставляется множество элементов подчиненного справочника. Таким образом эти связи образуют отношения один ко многим. Перед использованием подчиненного справочника сначала должен быть выбран элемент справочника «Владелец». После этого при включенном режиме «Иерархический список» в окне подчиненного справочника будут показаны элементы, подчиненные выбранному элементу справочника «Владелец».

Справочники имеют несколько видов форм для представления информации: Форма элемента, форма группы, форма списка. Форм списка может быть несколько. Каждая из этих форм имеет модуль, в который помещаются различные процедуры и функции.

7.1. Предопределенные процедуры модуля формы и модуля формы списка справочника

ВводНового - процедура, которая вызывается в момент ввода пользователем нового элемента справочника.

ПриЗаписи - процедура, которая вызывается перед записью элемента в справочник.

ПриВводеСтроки - процедура, которая вызывается в момент ввода новой строки в форме списка справочника.

ПриРедактированииНовойСтроки - процедура, которая вызывается в момент начала редактирования новой строки в форме Списка Справочника (после Отработки Процедуры *ПриВводеНовой Строки).*

ПриНачалеРедактированияСтроки - процедура, которая вызывается в момент начала редактирования существующей строки в форме списка справочника.

ПриПереносеЭлементаВДругуюГруппу - процедура, которая вызывается в момент переноса элемента справочника из одной группы в другую. Осуществляется посредством атрибута Родитель.

7.2. Атрибуты объекта метаданных типа «Справочник»

Атрибут или Реквизит - это некое хранилище информации определенного типа. Каждый введенный нами реквизит позволит вводить в объект данного вида некоторые значения.

Для объектов некоторых видов характерны атрибуты, например: код и наименование для справочников. Ввод атрибутов обеспечивается системой автоматически, наименования идентификаторов атрибутов не изменяются.

Атрибуты Код и Наименование характерны для любого справочника. По ним можно производить сортировку элементов справочника. Тип Наименования - строка. Тип Кода можно задать — или число, или строка.

Атрибут Родитель определен для элементов или групп справочника подчиненных какой-либо группе справочника и возвращает ссылку на неё.. Доступ к значению родительской группы выбранного элемента справочника осуществляет Метод *ИспользоватьРодителя(<Группа >)*, устанавливающий выборку элементов по группе справочника.

Метод *ПринадлежитГруппе(<Группа>)* проверяет, принадлежит ли указанной группе текущий элемент справочника (независимо на каком нижележащем уровне он находится). Метод возвращает: 1 - если элемент принадлежит указанной группе , 0 - если нет.

Атрибут Владелец определен для элементов подчиненного справочника и возвращает ссылку на сопряженный элемент справочника владельца. Доступ к владельцу выбранного элемента справочника в связанном справочнике осуществляет Метод *ИспользоватьВладельца(<Элемент>)*. <Элемент> - это значение элемента связанного справочника, которому подчинен данный

справочник. Метод устанавливает выборку по элементу связанного справочника и возвращает значение текущего владельца для справочника (на момент до исполнения метода). В контексте Модуля формы списка справочника метод относится ко всему текущему справочнику.

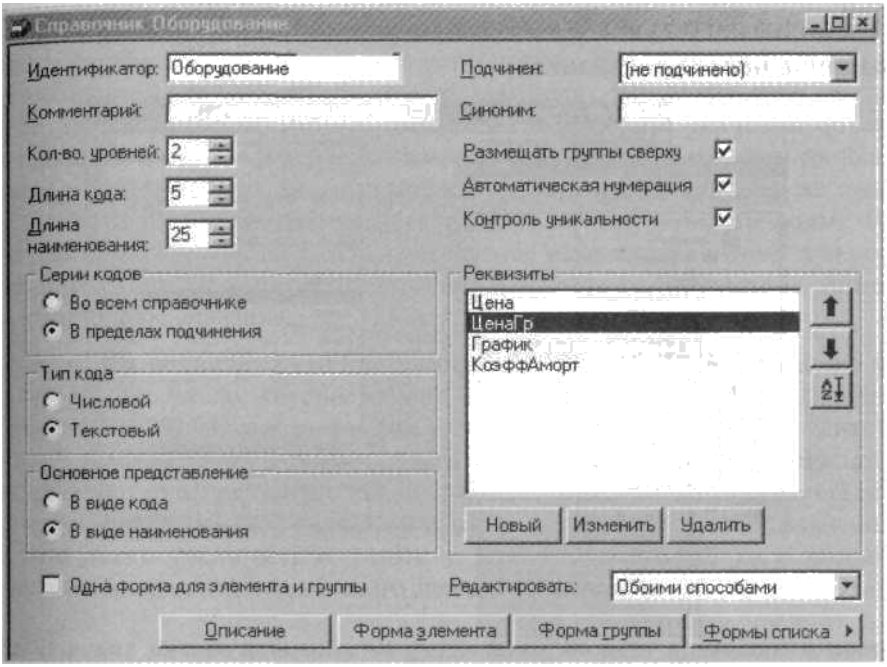
Реквизиты это дополнительные свойства справочника, создаваемые при конфигурировании. Реквизиты можно создавать и для элемента, и для группы справочника. Они предоставляют доступ к одному из свойств объекта.

Редактирование структуры справочника

Упражнение 5. Создадим два вида справочников: «Оборудование» и «Единицы», причем справочник «Единицы» сделаем подчиненным справочнику «Оборудование», Справочник «Оборудование» сделаем иерархическим, и определим в нем периодический реквизит элемента справочника, изменяемый документами «Цена», реквизит «График», тип реквизита — Календарь», предназначенный для вычисления интервалов времени по различным графикам работы, и реквизит «КоэффАморт», тип реквизита — Число, а также реквизит группы справочника «ГрЦена», тип реквизита — Число.

В дереве метаданных выделим ветвь «Справочники», далее в главном меню выберем пункт «Действие» и в появившемся подменю пункт «Новый элемент».

Появится окно редактирования структуры справочника (или конструктор справочника, если установлен флажок «Использовать конструкторы для создания новых объектов»), в которой редактируются свойства объекта.



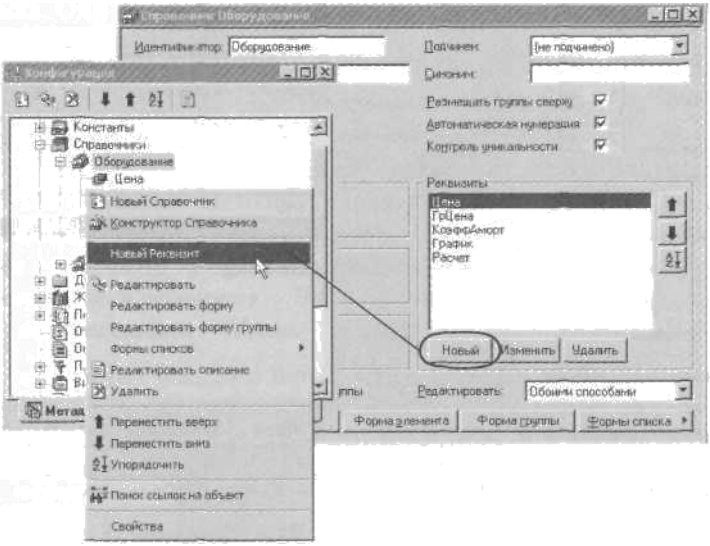
Чтобы к объекту можно было обращаться, ему надо задать идентификатор, который не может содержать пробелов и специальных символов, кроме подчеркивания, и не должен начинаться с цифры, его длина не должна превышать 128 символов; кроме идентификатора, можно также задать *комментарий и синоним. В* отличие от идентификатора, на них не накладывается никаких ограничений. Синоним описывает пользовательское представление объекта, если он не задан, то для представления используется идентификатор. В комментариях обычно записывают текст, Расшифровывающий идентификатор или поясняющий назначение создаваемого объекта. Для подчиненных справочников задается свойство подчинения (связь со справочником владельцем типа «много к одному»). Для справочников любого вида характерны реквизиты «Код» и «Наименование», названия которых не изменяются, длина кода (не более 24 символов) и наименования (не более 100 символов). Также здесь задаются основные параметры для нумерации и сортировки элементов - «Контроль уникальности» и «Серии кодов». Из данной формы доступны различные формы представления справочника: форма элемента (индивидуальная карточка), форма группы, Формы списков и устанавливаются формы для редактирования

элемента справочника: «в списке», «в диалоге» - в индивидуальной карточке или «обоими

способами».

По умолчанию справочнику дается идентификатор «Новый1», изменим его на - «Оборудование», установим тип кода — «Текстовый».

Создание нового реквизита



Создание нового реквизита справочника может осуществляться как из дерева метаданных, через контекстное меню конкретного справочника, так и из вышеописанной формы нажатием кнопки «Новый». При этом на экране появляется окно *палитры свойств* - «Свойства реквизита». В верхней части закладки «Общие» следует ввести идентификатор нового реквизита. Идентификатор реквизита является некоторым уникальным словом, которое позволит отличить данный реквизит от других реквизитов данного объекта. При создании нового рекви-зита программа сама подставляет идентификатор «новый1». Целесообразно вводить идентифика-торы, отражающие назначение реквизита. Введем в справочник «Оборудование» реквизит «Цена» Тип вводимого реквизита указывается выбором из списка возможных типов в поле «Тип значения». Стандартными типами являются «Число», «Строка», «Дата». Они позволяют вводить, соответственно, числовое значение, произвольную строку символов или календарную дату. Установим тип значения реквизита «Число», который располагается вы-ше типа «Строка».

Для числовых реквизитов, кроме самого типа «Число», нужно также указать длину и точность (количество знаков после запятой). Обратите вни-мание, что длина числового реквизита указывается с учетом целой части, дробной и

десятичной точки, на платформе 7.5 максимальная точность чисел - 5 знаков.

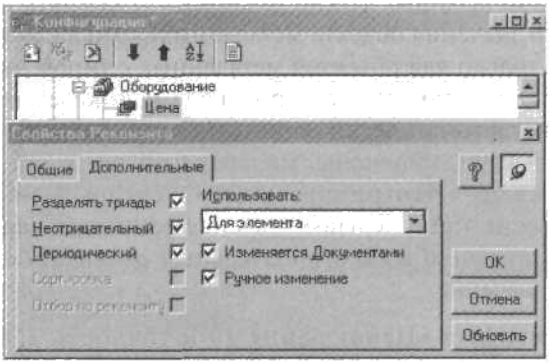
на платформе 7.7-9 знаков, максимальная длина чисел -19 знаков. На закладке «Дополнительные» можно задать разделение представления числа на триады -флажок «Разделять триады»,- которое не влияет на формат хранения чисел, то есть символ разделителя не включается в длину числа. Полностью данное свойство объекта метаданных можно именовать «Разделять триады цифр при показе числового значения объекта метаданных». Это свойство доступно для редактирования только для объектов метаданных с типом значения «число». Если это свойство включено, то при вводе и показе значения объекта метаданных будут автоматически вставляться разделители между тройками цифр, разделяя тысячи, миллионы, миллиарды и так далее. Здесь же можно установить флажок «Неотрицательный», который не позволит Вам интерактивно ввести отрицательное значение реквизита, причем при вводе отрицательного значения в «неотрицательный реквизит» система запишет положительное значение.

Укажем для реквизита «Цена» длину 10 и точность 2, это значит, что максимальное число, которое можно будет ввести в реквизит количество, составит 9999999.99, то есть, в целой части количества мы сможем вводить 10-2-1 = 7 разрядов. При выборе длины и точности нужно ориентироваться на текущие потребности, оценив, какие при-мерно максимальные значения цены и с какой точностью встре-чаются в хозяйственных операциях. В дальнейшем можно легко увеличить и длину, и точность для любого реквизита. Максимальная точность числового значения - 9 знаков.

Уменьшение длины и точности реквизита может привести к потере существующих значе-ний в уже введенных элементах, но программа предупредит *вас* ***об этом.***

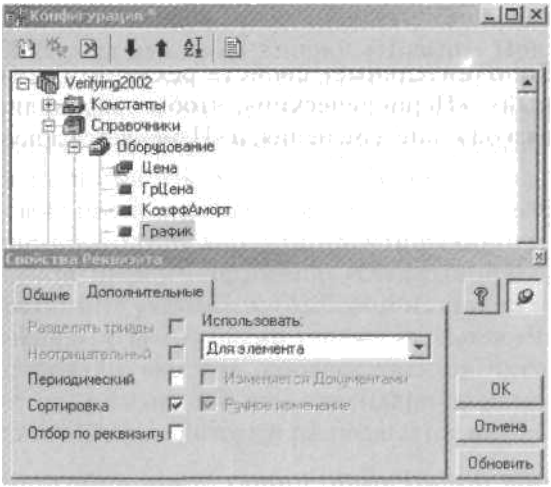
На закладке «Дополнительные» свойств реквизита установим флажки «Неотрицательный», «Периодический», чтобы сохранялись все значения реквизита с привязкой к дате изменения, и «Изменяется документами», чтобы

можно было отслеживать историю изменения значения по документам; флажок «Ручное изменение» устанавливается системой и служит для обеспечения возможности ручного ввода или изменения значений в истории реквизита.

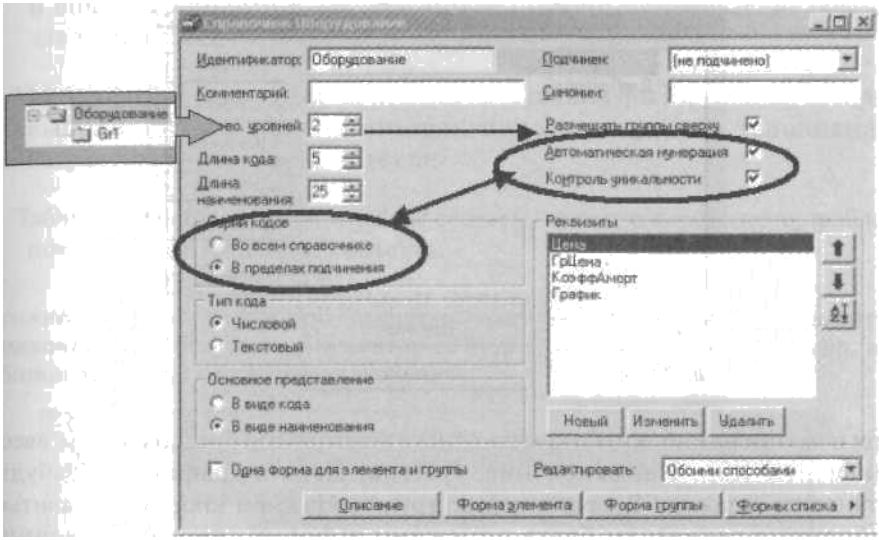


Для подтверждения введенных нами данных нужно нажать кнопку *«ОК.»*. При этом палитра свойств будет закрыта, а введенный реквизит будет располагаться в списке реквизитов, в окне редактирования реквизитов.

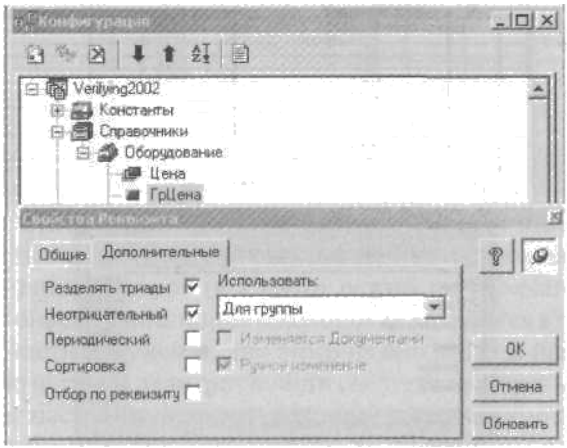
При необходимости использования выборки или поиска по реквизиту у последнего должен быть установлен на закладке «Дополнительные» флаг «Сортировка», а для возможности отбора элементов по значению реквизита - флаг «Отбор по реквизиту». Установим флаг сортировки для реквизита «График».



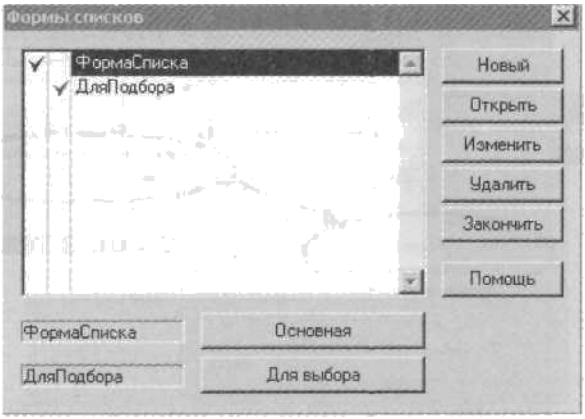
Вернемся в окно редактирования свойств справочника «Оборудование» и создадим возможность группировать элементы справочника по группам. Для этого установим значение поля ввода «Кол-во, уровней» больше единицы.



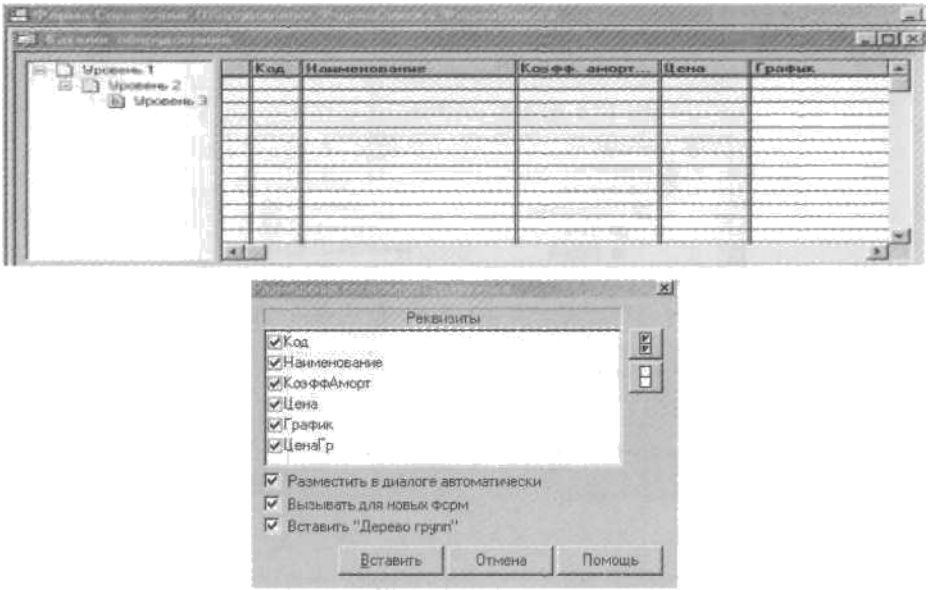
Свойство реквизита многоуровенного справочника «Использовать» позволяет определить, для каких записей будет использоваться данный реквизит: «для элемента», «для группы» или «для обоих».



Далее нажмем последовательно кнопки «Формы списка» и «Редактировать». Форм списка может быть несколько. Две из них можно определить для просмотра каталога и для подбора значений по умолчанию.



При нажатии на кнопку «Открыть» списка конструктор предложит нам ввести реквизиты Код, Наименование, ГрЦена, Цена и График. Не забудьте установить флажок «Вставить дерево групп». Нажмем кнопку «Вставить» и выбранные реквизиты будут помещены в форму списка справочника «Оборудование».



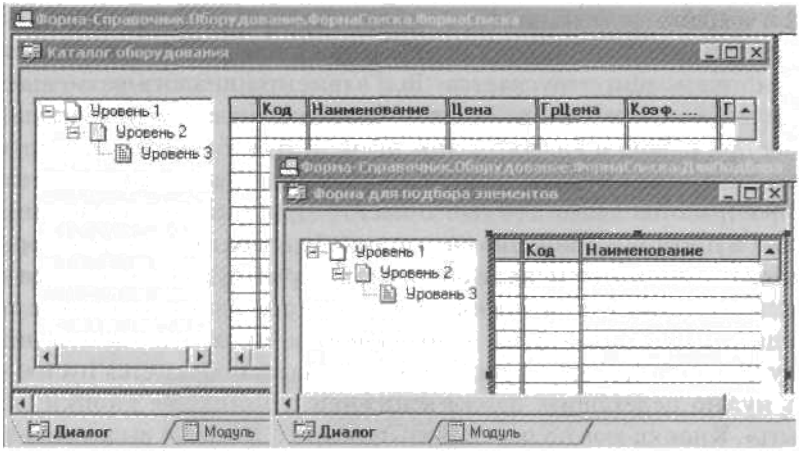
Напомню, что форма включает в себя:

1. Диалоговое окно, в котором мы задаем расположение реквизитов, кнопок и информационных полей (создаем визуальное представление объекта для пользователя);

2. Модуль формы, в который мы помещаем те процедуры и функции, которые выполняются в момент интерактивного открытия, заполнения, сохранения и изменения объекта;

3. Табличный шаблон, в котором мы создаем, если это необходимо, шаблон печатной формы данного объекта.

Упражнение 6. Самостоятельно создайте Форму списка для подбора элементов справочника «Оборудование»,которая будет содержать дерево групп, а в таблице атрибуты код и наименование.



Чтобы можно было установить вручную нужную ширину колонок многострочной части, следует сначала отключить режим автоматического определения ширины колонок (в 7.7 этот режим по умолчанию отключен). Автоматическое определение ширины колонок отключается в палитре свойств многострочной части документа, для этого нужно дважды щелкнуть мышью в многострочную часть. В палитре свойств следует щелкнуть мышью флажок «Автоматическая настройка ширины колонок» и нажать кнопку *«ОК»,* чтобы подтвердить произведенные изменения. После этого можно перетаскиванием вертикальных границ колонок табличной части установить для них желаемую Ширину. Для перетаскивания границ колонок необходимо поместить указатель Мыши над линией, разделяющей колонки в табличной части, чтобы курсор мыши принял соответствующую форму, затем нажать правую кнопу мыши и, не

опуская ее, перетащить разделитель колонок в новое место.

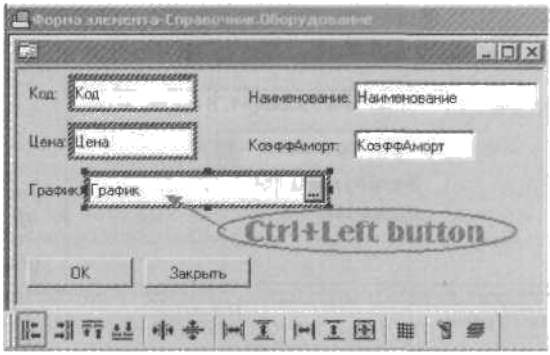
Если все колонки не помещаются в видимой области табличной части, то, для «прокрутки» колонок по горизонтали, следует щелкать мышью в левую или правую стрелку горизонтальной полосы прокрутки у нижней границы табличной части.

Можно изменить порядок следования колонок, перетаскивая их в пределах табличной части. Для перетаскивания колонки её следует выделить, щелкнув мышью в ее заголовок, и переместить указатель мыши, при нажатой левой кнопке, в то место, где должна располагаться колонка, при этом выделенной, т.е. редактируемой, колонкой останется та же колонка по номеру позиции; например, была выделена 5-я колонка и перемещена в 3-ю позицию, но выделенной станет колонка в 5-ой позиции, то есть та, которая была 4-ой.

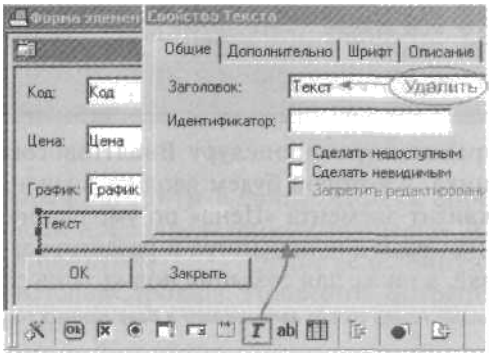
Упражнение 7. Создайте диалоговую Форму для элементов справочника «Оборудование».

Редактирование формы в основном выполняется мышью. Основным приемом работы является «перетаскивание». При перетаскивании щелкают левой кнопкой мыши в элемент, который нужно перетащить и, не отпуская левую кнопку, ведут указатель мыши к тому месту, куда перетаскивается элемент, а затем кнопка мыши отпускается. Все элементы диалога размещаются в пределах *шаблона окна диалога.* Также можно поместить элемент и вне шаблона, но тогда при работе с диалогом этот элемент будет невиден и недоступен. Сначала отодвинем вниз нижнюю границу диалога. Щелкните мышью в пустое серое пространство диалога (в любое место). При этом граница формы будет выделена пунктирной линией, на которой будут размещены маркеры. Эта линия показывает тот элемент диалога, над которым будут выполняться какие-либо действия. Для изменения вертикального размера диалога нужно перетащить мышью маркер на пунктирной линии обозначающей нижнюю границу диалога. При этом в нижней части диалога образуется пустое место. Теперь нужно перетащить ближе к нижней границе две кнопки: «ОК» и «Закрыть». Кнопки можно перетащить по одной, а можно выделить их обе и перетащить две одновременно. Для выделения кнопок щелкните мышью од-ну кнопку, затем нажмите клавишу Shift и, не отпуская ее, щелкните мышью другую кнопку. Вокруг каждой кнопки будет нарисована синяя пунктирная линия с маркерами. Перетащить такую группу элементов можно, перетаскивая любой элемент. Выделять элементы диалога можно также, просто «обводя» их мышью - границу выделяемой области будет показывать тонкая пунктирная линия. Но таким образом можно выделять только элементы, расположенные рядом. Выделенные элементы можно отформатировать по шаблону. Для выбора

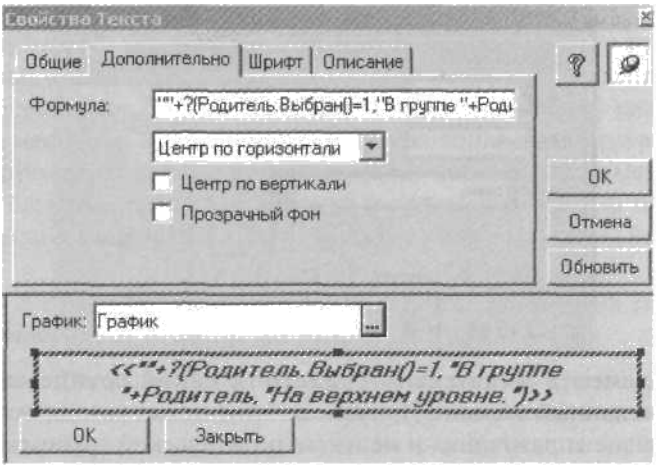
шаблона, нажав клавишу Ctrl, щелкните левой кнопкой мыши нужный элемент формы диалога.



В форме элемента желательно увидеть, к какой группе справочника принадлежит данный элемент. Причем, элемент может находиться на верхнем (первом) уровне справочника и не иметь родительской группы. Создадим в форме реквизит с соответствующим текстом. Выберем в меню «Вставить» или в панели инструментов инструмент «Текст» и наметим курсором место в форме, куда мы поместим реквизит формы. Для этого нужно поместить курсор в левый верхний угол предполагаемой области формы диалога, в которую предполагается их поместить, нажать левую кнопку мыши, не отпуская ее, переместить курсор к правому нижнему углу предполагаемой области размещения реквизита, и отпустить кнопку мыши. Так как текст в реквизите мы будем выводить с помощью выражения на встроенном языке в поле «Формула» на закладке «Дополнительные», то текст заголовка реквизита надо удалить.



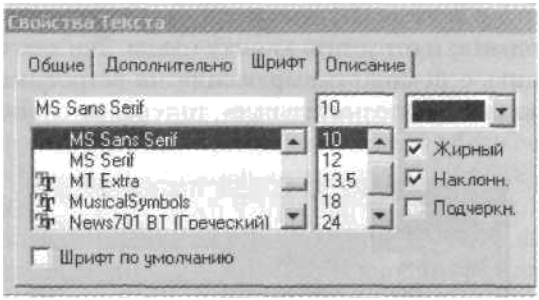
Запишем выражение в поле «Формула» свойств текста, используя, оператор конкатенации строк:



Метод **?**(<Логич_выраж>,<Выраж1>,<Выраж2>) возвращает значение первого выражения, если условие истинно, иначе - значение второго выражения.

Метод Выбран( ) возвращает флаг выбора элемента справочника; 1 - если элемент справочника выбран, О - если не выбран.

Шрифт отредактирован на закладке «Шрифт» текстового реквизита.



Создадим предопределенную процедуру ВводНового() в модуле формы элемента справочника, в которой будем вводить значение цены по группе справочника в реквизит элемента «Цена» по умолчанию. Форму элемента справочника мы открываем для просмотра и редактирования существующих записей справочника, а также для создания новых записей в справочнике.

Метод **Уровень()** возвращает номер уровня ***записанного в базу данных*** текущего элемента справочника.

В последнем случае во время работы в форме элемент ещё не записан, поэтому метод Уровень() вернет значение 0. Поэтому для определения принадлежности элемента группе справочника используем знакомый нам метод Выбран() для атрибута Родитель.

```
Процедура ВводНового()
        Если Родитель.Выбран()=1 Тогда
            Цена=Родитель.ГрЦена;
        КонецЕсли;
КонецПроцедуры
```

Предопределенная процедура при интерактивном вводе нового элемента справочника **ВводНового**(<ПризнКопирования>,<ОбъектКопирования>).
Параметры:<ПризнКопирования> - признак того, что объект введен копированием. Число: 1 - объект введен копированием, 0 - просто новый объект. Данный признак может быть использован для анализа необходимости инициализации реквизитов нового объекта.<ОбъектКопирования> - объект, который был скопирован.

При копировании записей справочника копируются все реквизиты, кроме кода записи справочника. Отредактируем принудительно атрибут элемента «Наименование», используя параметры предопределенной процедуры ВводНового() и метод ВвестиСтроку().

Метод **ВвестиСтроку**(<Строка>,<Подсказка>,<ДлинаСтроки>,<Признак>,<Таймаут>) Вызывает окно интерактивного диалога для ввода строки. Возвращает:
1 - если в диалоге нажата кнопка ОК;
О - если нажата кнопка Отмена;
-1 - если закончилось время ожидания ответа.
Параметры: <Строка> - имя переменной, объявленной в модуле для приема вводимого значения; <Подсказка> - текст заголовка окна диалога ввода; <ДлинаСтроки> - длина вводимой строки; <Признак> - если 0 или опущен - ввод одной строки, если 1 - ввод многострочного текста с разделителями строк; <Таймаут> - число секунд времени ожидания ответа (если опущен или 0, то без ограничения).

```
Процедура ВводНового(признак,копируемый)
        Если признак=1 Тогда
        ВвестиСтроку(Наименование,"Введите наименование",25);
        КонецЕсли;
        Если Родитель.Выбран()=1 Тогда
            Цена=Родитель.ГрЦена;
        КонецЕсли;
КонецПроцедуры
```

Запишем аналогичный алгоритм в форме списка справочника. В этом контексте свой список предопределенных процедур. Для ввода цены по Умолчанию используем предопределенную процедуру ПриРедактированииНовойСтроки(). Проверить операцию копирования в форме списка мы не можем.

```
Процедура ПриРедактированииНовойСтроки () Если
        Родитель.Выбран()=1 Тогда
            Цена=Родитель.ГрЦена;
        КонецЕсли;
```

```
КонецПроцедуры
```

Упражнение 8. Создайте справочник «Единицы». В окне редактирования справочника «Единицы» установите подчинение (в поле ввода «Подчинен») справочнику «Оборудование» и создайте реквизит «Коэффициент», в который будут записываться значения коэффициентов пересчета соответствующие единицам измерений. В форме элемента и в форме списка справочника в текстовом реквизите формы покажите владельца элемента справочника.

Для вставки в табличную часть формы списка текстового реквизита щелкнем мышью кнопку с синей буквой «Т» в панели элементов диалога или соответствующую строку в меню «Вставить», установим его на табличную часть, курсор мыши при этом примет специальный вид, и ещё раз щелкнем левой кнопкой мыши. Появится дополнительная колонка без названия в самой правой позиции табличной части формы, чтобы её увидеть, как правило, надо прокрутить табличную часть стрелкой вправо. Для ввода формулы нужно активизировать свойства колонки. Сначала щелчком мыши в табличную часть мы должны активизировать табличную часть, а затем двойным щелчком в заголовок колонки активизировать свойства этой колонки. При этом появляется палитра свойств колонки. Зададим колонке название в поле «Заголовок», а в поле «Формула» напишем — Владелец.

7.3. Использование подчиненных справочников

При создании конкретных конфигураций довольно часто возникают задачи, когда надо реализовать для каждого конкретного элемента справочника -список относящихся к нему объектов (отношение один-ко-многим). В нашем примере имеется справочник оборудования, и каждый элемент справочника может иметь произвольный список единиц измерения данного оборудования. Для того чтобы ввести новый или отредактировать существующий элемент справочника «Единицы», сначала откройте форму списка справочника владельца «Оборудование», выберите в нем курсором нужный элемент. Затем откройте форму списка подчиненного справочника «Единицы», он будет отображать список единиц измерения данного оборудования. Теперь можно добавлять, редактировать или удалять записи в справочнике «Единицы». Выбор элементов подчиненного справочника, в форме, при условии, что в форме существует элемент диалога, в котором выбран элемент справочника владельца, для которого производится выбор, реализуется в конфигурации при помощи указания в палитре свойств реквизита формы, предназначенного для выбора элемента подчиненного справочника, параметра «Связан с». В нем вводится имя идентификатора реквизита формы, который содержит элемент родительского справочника. В этом случае при выборе элемента подчиненного справочника автоматически будет производиться выбор из элементов, подчиненных выбранному элементу справочника-родителя. Другим способом установки элемента справочника владельца, по значению которого должен выполняться выбор подчиненного справочника является вызов метода *ИспользоватьВладельца(<?>)* для объекта подчиненного

справочника. Это позволяет определить выбор подчиненного справочника даже в том случае, когда в форме нет элемента диалога имеющего тип справочника - владельца. Для выборки элементов подчиненного справочника по владельцу, после метода *ИспользоватьВладельца(< ?>)* надо вызвать метод *ВыбратьЭлементы()* и далее методы *ПолучитьЭлемент()* или *НайтиЭлемент(< ?>), НайтиПоКоду(< ?>), НайтиПоНаименованию(< ?>).*

Упражнение 9. (Необязательное) Создайте в форме элемента и форме списка справочника «Оборудование» алгоритм, позволяющий при создании нового элемента программно создавать подчиненный элемент в справочнике «Единицы» с Наименованием «шт.» и Коэффициентом равным 1.

Опишем для этого предопределенные процедуры ПриЗакрытии() в модуле формы и в модуле формы списка справочника «Оборудование», потому что последовательность директив в кнопке «ОК» в форме следующая: #Записать? Закрыть.

Метод **ТекущийЭлемент()** возвращает значение ***записанного в базу данных*** позиционированного текущего элемента справочника.

В модуле формы:

```
Перем ФлагНового;

Процедура ВводНового(признак,копируемый)

      Если признак=1 Тогда

          ВвестиСтроку(Наименование,"Введите наименование",25);

      КонецЕсли;

      Если Родитель.Выбран()=1 Тогда

          Цена=Родитель.ГрЦена;

      КонецЕсли;

      ФлагНового-1;

КонецПроцедуры

Процедура ПриЗакрытии()

      Если ФлагНового=1 Тогда

          Ед=Создать0бъект("Справочник.Единицы");

          Ед.ИспользоватьВладельца(ТекущийЭлемент());

          Ед.Новый();

          Ед.Наименование="шт.";

          Ед.Коэффициент=1;

          Ед.Записать();

      КонецЕсли;
```

```
КонецПроцедуры

ФлагНового=0;
```

В модуле формы списка мы можем создать несколько элементов, поэтому создадим список значений новых значений.

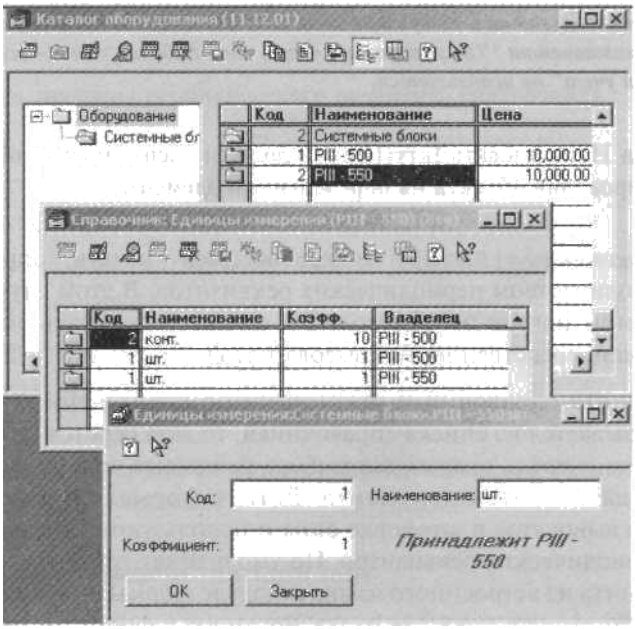
Метод **РазмерСписка**( ) возвращает число элементов в списке значений.
Метод ПолучитьЗначение(<Позиция>,<Строка>) позволяет получить значение из указанной позиции списка. Возвращает значение из списка.
<Позиция> - номер позиции в списке, из которой возвращается значение (изменяется от 1 до РазмерСписка); <Строка> - идентификатор переменной, в которой возвращается символьное представление получаемого значения.
Метод НаЙтиПоКоду(<Код>,<ФлагПоиска>) позволяет найти элемент справочника по коду.
Возвращает: 1 - если действие выполнено;
О - если действие не выполнено (элемент не найден).
<Код> - выражение со значением искомого кода,
<флагПоиска> - флаг поиска (необязателен):
0- поиск во всем справочнике вне зависимости от родителя;
1- поиск внутри установленного подчинения (родителя);
2- поиск по полному коду через разделитель.
Значение по умолчанию:
О - если код уникален во всем справочнике; 2 - если код уникален только'в группе. Его можно использовать только для объектов, созданных функцией СоздатьОбъект.

```
Перем СписокНовых;

Процедура ПриРедактированииНовойСтроки ()

    Если Родитель.Выбран()=1 Тогда

       Цена=Родитель.ГрЦена;

    КонецЕсли;

    СписокНовых.ДобавитьЗначение(Код,Наименование);

КонецПроцедуры

Процедура ПриЗакрытии()

    Перем Стр;

    Если СписокНовых.РазмерСписка()>0 Тогда

        Новый=СоздатьОбъект("Справочник.Оборудование");

        Для i=1 по  СписокНовых.РазмерСписка() Цикл

           Если Новый.НайтиПоКоду (СписокНовых.ПолучитьЗначение (i,

           Стр),0) =1  Тогда

                 Ед=СоздатьОбъект("Справочник.Единицы");

                 Ед.ИспользоватьВладельца(Новый.ТекущийЭлемент());

                 Ед.Новый();

                 Ед.Наименование="шт.";

                 Ед.Коэффициент = 1;

                 Ед.Записать();
```

```
КонецЕсли;
```

```
            КонецЦикла;//

        КонецЕсли;

    КонецПроцедуры

    СписокНовых=СоздатьОбъект("СписокЗначений");
```

На приведенном ниже рисунке показано, как выглядят формы списков обоих справочников и форма элемента справочника «Единицы». В форме списка справочника «Единицы» отключен режим иерархии, но благодаря текстовому реквизиту «Владелец» можно различить единицы измерения с одинаковыми наименованиями и кодами , принадлежащие разным владельцам, в форме списка.



7.4.Использование периодических реквизитов справочников

Использование периодических реквизитов существенно облегчает реализацию многих прикладных моментов конфигурации, однако, использование периодических реквизитов требует при конфигурировании особого внимания. Во-первых, не следует злоупотреблять этим свойством. Установка свойства «периодический» для каждого реквизита должна быть обоснованным и хорошо продуманным шагом, иначе это просто усложнит работу пользователей со справочниками, но не даст большого выигрыша. Во-вторых, при конфигурировании следует всегда помнить, что для

использования значения периодического реквизита справочника необходимо обязательно указывать дату, на которую потребуется выбрать или

записать значение. Установка даты использования периодических реквизитов задается либо методом ИспользоватьДату ([Дата]) , либо методами Получить([Дату])и Установить ([Дата] , [Значение]) (через точку после самого реквизита). Одновременно использовать оба эти способа задания даты для одного и того же объекта типа Справочник нельзя. Если метод ИспользоватьДату! [Дата]) не вызывался, то к периодическому реквизиту справочника следует обращаться с помощью методов Получить ([Дату]) и Установить ([Дата] ,[Значение]). Если метод ИспользоватьДату ([Дата]) был вызван, то можно обращаться только непосредственно к значениям реквизитов.

На платформе 7.7 можно не указывать значение даты для метода Получить(), по умолчанию устанавливаются следующие значения: значение точки актуальности - если используется компонента "Оперативный учет", или Рабочая дата - если компонента "Оперативный учет"не используется.

Вызов метода ИспользоватьДату([Дата]) должен располагаться обязательно до позиционирования объекта на определенный элемент.

Примером, целесообразного использования метода ИспользоватьДату ([Дата]) может являться печать справочника с большим количеством периодических реквизитов. В этом случае, один раз вызвав в самом начале этот метод, Вы сможете получать значения всех реквизитов непосредственно, без вызова метода Получить ([Дату]).

Форма элемента справочника открывается с определенной датой. Если элемент открывается из списка справочника, то дата берется из списка. Если список в свою очередь открыт для выбора реквизита документа, то его дата будет совпадать с датой документа. Даты в формах списка и элемента справочника выводятся в заголовке окна и используются только для вывода и записи периодических реквизитов. По умолчанию это Рабочая дата, но её можно изменить из встроенного языка в модуле формы списка или элемента (группы). При записи элемента редактируемого в форме, если существуют периодические реквизиты, то выдается специальный запрос, в котором пользователь отмечает какие значения периодических реквизитов нужно установить на дату формы, вызов и вид запроса можно откорректировать методом модуля формы элемента справочника СохранениеПериодическихРеквизитов().

При интерактивном редактировании формы элемента (группы) или формы списка справочника периодические реквизиты могут записываться или не записываться на дату формы, но не могут быть записаны на другую дату.

По умолчанию изменения реквизитов выставляются на основании сравнения значений со значениями, записанными на ближайшую предыдущую дату. Данный запрос выдается только для реквизитов, непосредственно

включаемых в форму. Если некоторый периодически реквизит непосредственно в форму не включен, а изменяется из встроенного языка, то он не будет выдаваться в этом запросе и сохраняться. Для того, чтобы в реквизит участвовал в запросе о записи периодических реквизитов можно вставить его в форму, установив ему признак «Сделать невидимым». Периодические реквизиты содержат значения только начиная с даты первой записи истории значений реквизита, однако и в историю реквизита можно записать «пустое значение» на какую-либо дату. Например, если самая первая запись периодического реквизита «Цена» отмечена датой '02.12.94' со значением 48.00, а мы в программном модуле запросили значение этого реквизита на '30.11.94', то система вам вернет «пустое значение», т.е. ноль; если «пустое значение» установлено на дату '04.12.94', а следующее не пустое значение на '14.12.94', то с 04 по 13.12.94 система также вернет вам «пустое значение».

Пример записи значения периодического реквизита справочника:

```
Словарь = СоздатьОбъект(«Справочник.Оборудование»);
Если Словарь.НайтиЭлемент(Элемент)  = 1  Тогда
     Периодич = СоздатьОбъект(«Периодический»);
     Периодич.ИспользоватьОбъект(«Цена», Словарь.ТекущийЭлемент());
     Периодич.Значение= 2 5;
     Периодич.ДатаЗнач = '17.05.00';
     Периодич.Записать();
КонецЕсли;
```

Здесь «Элемент» - переменная или реквизит объекта с типом значения Справочник. Оборудование.

7.5. Работа со слоями и закладками

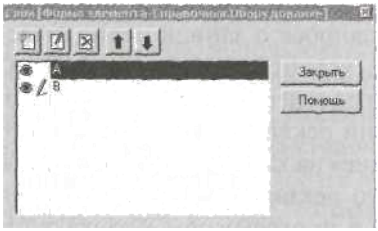
Закладки позволяют, не перегружая видимую часть формы объекта, расположить в ней максимальное число элементов формы.

Упражнение 10. В форме справочника «Оборудование» разместите реквизиты формы справочника по слоям, создайте закладки и алгоритм управления формой.

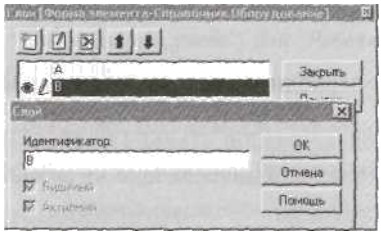
Разделим форму по слоям. Для этого (при активном окне формы диалога объекта) выберем пункт главного меню «Диалог», а в нём пункт «Слои». В появившемся окне «Слои» мы увидим один слой «Основной», создаваемый по умолчанию.

Нажмём кнопку «Изменить», в появившемся окне «Слой» изменим

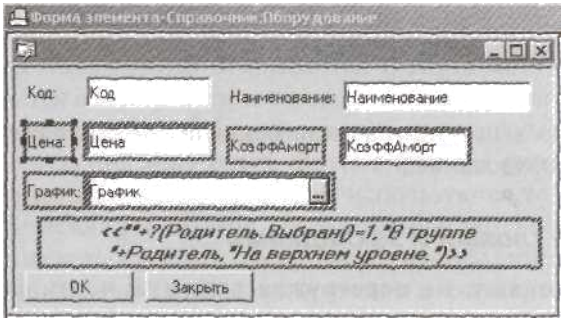
идентификатор слоя на «А». Создадим новый слой, для этого нажмём кнопку «Новый» в окне «Слои» и зададим ему идентификатор «В».



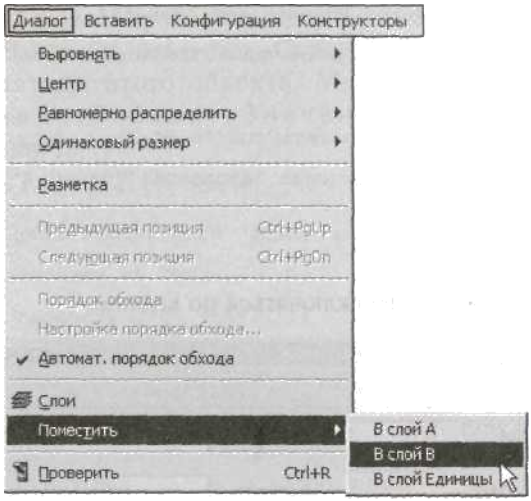
Новый слой автоматически становится Видимым и Активным, слой «А» при этом станет только Видимым.



Нажмём кнопку «Закрыть», чтобы закончить редактирование слоев. Выделим реквизиты формы диалога,



в пункте главного меню «Диалог» выберем пункт «Поместить» и поместим выбранные элементы диалога в слой «В».



Используя окна управления слоями, описанные выше, можно посмотреть, как расположились элементы формы диалога по слоям.

Для управления слоями используется метод формы

Форма. ИспользоватьСлой(«Имя_Слоя»,<Режим>)

<ИмяСлоя> - строковое выражение - название слоя формы, как оно задано в конфигураторе. Параметр может быть составным (указывать несколько слоев). В этом случае имена слоев перечисляются в строке через запятую.

<Режим> - необязательный параметр. Числовое выражение:

0- скрыть слой <ИмяСлоя> в форме;

1- показать слой <ИмяСлоя> в форме;

2- показать слой <ИмяСлоя> и скрыть все остальные.

Значение по умолчанию - 2. Доступ к методу возможен только в контексте Модуля формы через атрибут Форма.

Чтобы посмотреть, как переключаются слои можно создать кнопку для переключения слоев и поместить в неё вызов процедуры Переключить(). В качестве параметра переключения можно использовать значение текущего заголовка кнопки. Обратите внимание на свойства метода реквизитов формы Заголовок(): он работает как процедура , и как функция.

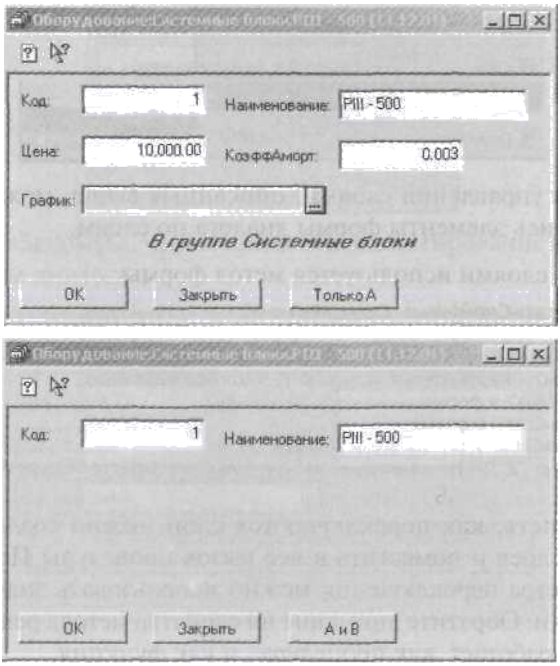
Метод **Заголовок**(<Название>) устанавливает заголовок элемента диалога. Возвращает текущий заголовок элемента диалога. <Название> - строковое выражение - новый заголовок колонки многострочной части формы, кнопки, рамки группы, текста, флажка, переключателя. Доступ к методу возможен только в контексте Модуля формы через атрибут Форма.

Чтобы к реквизиту формы можно было бы обратиться, его надо назвать, задать ему идентификатор. Зададим кнопке идентификатор - «Переключение» и Начальный заголовок — «Только А». Создадим в модуле формы справочника процедуру Переключить()

```
//Вызов процедуры в кнопке -
Переключить(Форма.Переключение.Заголовок())
```

```
Процедура Переключить(ТекущийЗаголовок)
    Если ТекущийЗаголовок="Только А" Тогда
        Форма.ИспользоватьСлой("А",2) ;
        Форма.Переключение.Заголовок("А и В");
    Иначе
        Форма.ИспользоватьСлой("А,В",2);
        Форма.Переключение.Заголовок("Только А") ;
    КонецЕсли;
КонецПроцедуры
```

В Предприятии слои будут переключаться по кнопке.



Как правило для переключения по слоям в форме создают закладки. Интерактивное действие — выбор закладки обрабатывается в предопределенной процедуре модуля формы ПриВыбореЗакладки (), которая запускается в момент выбора закладки пользователем.

ПриВыбореЗакладки(НомерЗакладки,ЗначениеЗакладки):

<НомерЗакладки> - числовое значение – номер выбранной закладки формы.
<ЗначениеЗакладки> - значение выбранной закладки формы.

Чтобы сделать закладки в форме доступными надо установить флаг

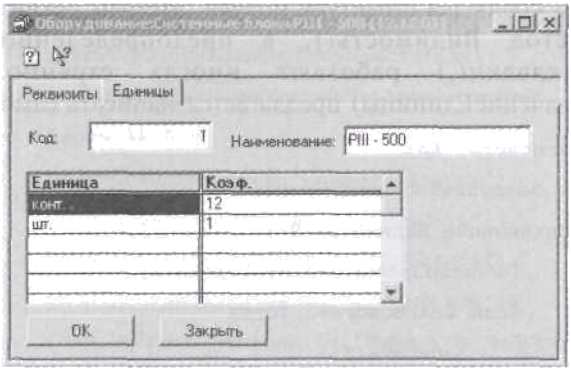
использования закладок.

Упражнение 11. В форме элемента справочника «Оборудование» создайте закладки и алгоритм управления слоями.

Напишем в модуле формы процедуры для управления Закладками и Слоями. Атрибут формы Закладки имеет тип «СписокЗначений», поэтому для него доступны все методы этого объекта. Мы будем использовать метод форма.Закладки.ДобавитьЗначение(ЗначениеЗакладки, «НазваниеЗакладки»).

```
//****************************************

Процедура ПриВыбореЗакладки (НомерЗакладки,ЗначениеЗакладки)

      Если НомерЗакладки =1 Тогда

         Форма. Использовать Слой («А», 2);

      ИначеЕсли НомерЗакладки =2 Тогда

         Форма. Использовать Слой («В», 2);

      Конецесли;

КонецПроцедуры

//****************************************

Форма. ИспользоватьЗакладки (1);

Форма. Закладки. ДобавитьЗначение («», «А»);

Форма. Закладки. ДобавитьЗначение («», «В»);
```

Упражнение 12. (Необязательное) В форме элемента справочника «Оборудование» создайте закладку и Слой «Единицы», алгоритм переключения между слоями, реквизит формы ТаблицаЗначений с идентификатором «ТаблицаЕдиниц» на слое «Единицы» и алгоритм редактирования подчиненных элементов справочника Единицы из формы элемента справочника владельца.(Аналогичный алгоритм мы создали в Упражнении 3)


Реквизиты
Единицы
Код:
1
Наименование:
PIII - 500
Единица
Коэф.
конт.
12
шт.
1
OK
Закрыть

В поле Формула таблицы значений вызывается процедура Редактироовать(ТаблицаЕдиниц.ТекущаяСтрока()), в параметр которой передается номер выбранной строки.

Метод **ТекущаяСтрока**(<Строка>) устанавливает или определяет текущую строку таблицы в элементе диалога типа "ТаблицаЗначений". Возвращает число, соответствующее индексу текущей строки поля диалога (до его изменения), или 0, если текущей строки нет. <Строка> - необязательный параметр. Числовое выражение с задаваемым индексом строки для элемента диалога типа "ТаблицаЗначений", на которую требуется установить курсор. Если параметр не задан, то положение курсора в поле диалога не меняется."

Метод **ПолучитьСтрокуПоНомеру**(<Номер>) устанавливает в качестве текущей строки строку с заданным номером.

Возвращает:

1 - если действие выполнено;

О - если действие не выполнено.

<Номер> - номер строки.

Метод **Видимость**(<Режим>) устанавливает режим отображения элемента диалога. Возвращает: текущее числовое значение режима отображения элемента диалога (на момент до исполнения метода). <Режим> - число: 1 - поле диалога отображается, 0 - поле диалога скрыто.

Доступ к методу возможен только в контексте Модуля формы через атрибут форма. Использование данного метода для колонки табличной части формы не может изменять видимость отдельно для совмещенных и многострочных колонок."

Метод **ВвестиСтроку**(<Строка>,<Подсказка>,<ДлинаСтроки>,<Признак>,<Таймаут>) вызывает диалог для ввода строки.

Возвращает:

1 - если в диалоге нажата кнопка ОК;

0 - если нажата кнопка Отмена;

-1 - если закончилось время ожидания ответа.

<Строка> - имя переменной, объявленной в модуле для приема вводимого значения; <Подсказка> - текст заголовка окна диалога ввода; <ДлинаСтроки> - длина вводимой строки;<Признак> - если 0 или опущен - ввод одной строки, если 1 - ввод многострочного текста с разделителями строк; <Таймаут> - число секунд времени ожидания ответа (если опущен или 0, то без ограничения).

Метод **ВвестиЧисло**(<Число>,<Подсказка>,<Длина>,<Точность>,<Таймаут>) вызывает диалог для ввода числа.

Возвращает:

1- если в диалоге нажата кнопка 'ОК';

О - если нажата кнопка 'Отмена';

-1 - если закончилось время ожидания ответа.

<Число> - имя переменной, объявленной в модуле для приема вводимого значения;<Подсказка> - текст заголовка окна диалога ввода;<Длина> - длина вводимого числа;<Точность> - число знаков после десятичной точки;<Таймаут> - число секунд времени ожидания ответа (если опущен или 0, то без ограничения).

Предложенный ниже алгоритм можно перенести в модуль формы справочника «Оборудование» соблюдая структуру модуля. Методы реквизитов формы, в частности метод Видимость(), в предопределенной процедуре ПриВыбореЗакладки(,) работают иногда странно. Процедуру ИзменитьЕд(ЗначениеЕдиницы) предлагается написать самостоятельно.

```
Перем ФлагНового, Ед;

Процедура ЗагрузкаТаблицыЕдиниц()

    Форма.Переключение.Видимость(0);

    ТаблицаЕдиниц.УдалитьСтроки();

    Если ФлагНового=1 Тогда
```

```
Записать();
```

```
        КонецЕсли;
        Ед.ИспользоватьВладельца(ТекущийЭлемент());
        Ед.ВыбратьЭлементы();
        Пока Ед.ПолучитьЭлемент()=1 Цикл
            ТаблицаЕдиниц.НоваяСтрока();
            ТаблицаЕдиниц.Наим=Ед.ТекущийЭлемент();
            ТаблицаЕдиниц.К=Ед.Коэффициент;
        КонецЦикла;
    КонецПроцедуры
    Процедура УдалитьЕд(ЗначениеЕдиницы)
        ТаблицаЕдиниц.ПолучитьСтрокуПоНомеру(ЗначениеЕдиницы);
        Ед.НайтиЭлемент(ТаблицаЕдиниц.Наим);
        Ед.Удалить();
        ЗагрузкаТаблицыЕдиниц();
    КонецПроцедуры
    Процедура ДобавитьЕд()
        Перем Стр,К;
        Ед . Новый ( ) ;
        ВвестиСтроку(Стр,"Введите наименование",25);
        Ед.Наименование=Стр;
        ВвестиЧисло(К,"Введите коэффициент",7,0);
        Ед. Коэффициент= К;
        Ед.Записать();
        ЗагрузкаТаблицыЕдиниц();
    КонецПроцедуры
    //В поле Формула таблицы значений вызывается процедура
    //Редактировать(ТаблицаЕдиниц.ТекущаяСтрока())
    Процедура Редактировать(НомерСтрокиТаблицыЗначений)
        Перем Зн,Поз;
        СпЗн=СоздатьОбъект("СписокЗначений");
        СпЗн.ДобавитьЗначение("","Удалить");
        СпЗн.ДобавитьЗначение("","Добавить");
        СпЗн.ДобавитьЗначение("","Изменить");
        СпЗн.ВыбратьЗначение(Зн,"Выберите действие",Поз,,1);
```

```
        Если Поз=1 Тогда
            УдалитьЕд(НомерСтрокиТаблицыЗначений);
        КначеЕсли Поз = 2 Тогда
            ДобавитьЕд();
        Иначе
            ИзменитьЕд(НомерСтрокиТаблицыЗначений);
        КонецЕсли;
    КонецПроцедуры
    Процедура ПриВыбореЗакладки(НомерЗакладки,ЗначениеЗакладки)
        Если НомерЗакладки=1 Тогда
            Форма.ИспользоватьСлой("А,В",2);
        Иначе
            Форма.ИспользоватьСлой("А,Единицы",2);
            ТаблицаЕдиниц.Очистить();
            ТаблицаЕдиниц.НоваяКолонка("Наим",,,,"Единица",10,,);
            ТаблицаЕдиниц.НоваяКолонка("К",,,,"Коэф.",10,,);
            ЗагрузкаТаблицыЕдиниц{};
        КонецЕсли;
        КонецПроцедуры
        //_____________________________________________
        Форма.ИспользоватьЗакладки(1);
        Форма. Закладки. ДобавитьЗначение ("", "Реквизиты");
        Форма. Закладки. ДобавитьЗначение ("", "Единицы");
        Форма.ИспользоватьСлой("А,В",2);
        Ед=Создать0бъект("Справочник.Единицы");
        ФлагНового=0;
```

Режим выборки групп в форме списка справочника.

По умолчанию, выборка элементов справочников для полей диалога в формах документов, журналов и формах списков справочников установлена без выбора групп, а в форме отчета с выбором групп. Если Вы хотите изменить режим выборки групп, используйте метод *ВыборГруппы(<?>).*

Методы определения префиксов авто нумерации

Префиксы авто нумерации создаются для удобства отбора или поиска записей в базе данных. В Предприятии есть два типа методов для определения префиксов авто нумерации. Метод **ПрефиксАвтоНумерации** (<ИмяВида>,

<Префикс>)

располагают, как правило, в глобальном модуле. Он позволяет установить префикс для автоматического создания новых номеров.Параметры: <ИмяВида> - строковое выражение с полным названием справочника или документа конфигурации. Для установки префикса сразу всем документам или справочникам используется символ "*" вместо идентификатора вида документа/справочника; <Префикс> - строковое выражение, задающее префикс номеров документов или кодов элементов справочника.

Методы описанные ниже вызывают обычно в предопределенных процедурах ВводНового() в модуле формы и ПриРедактированииНовойСтроки() в модуле формы списка справочника.

Метод **ПрефиксКода**(<Префикс>) позволяет установить текущий префикс кода для справочника. Возвращает строковое значение текущего префикса кодов элементов справочника (на момент до исполнения метода).

Метод **УстановитьНовыйКод**(<Префикс>) позволяет установить новый код элемента справочника с заданным префиксом. <Префикс> - строка с префиксом кода элемента справочника.

Вопросы для самоконтроля

Как задается область определения реквизитов справочников?

Для каких справочников определены атрибуты Владелец и Родитель?

Какое основное отличие в использовании и обращении к подчиненным и обычным справочникам?

Какие формы можно создать для интерактивного редактирования справочников?

В чем отличие конфигурирования периодических реквизитов справочников от конфигурирования периодических констант?

В чем отличие реквизитов формы и реквизитов объекта, как формата хранения данных?

Можно ли редактировать данные объекта, не открывая его интерактивную форму?

Можно ли управлять слоями в форме, не используя предопределенную процедуру ПриВыбореЗакладки()?

VIII. Документы и журналы документов

Документы - предназначены для регистрации, корректировки и просмотра актов хозяйственной деятельности предприятия. В конфигураторе системы мы создаем форму документа, которая используется для ввода информации. При проведении документа создаются записи в учетных базах: движения по регистрам, бухгалтерские операции и проводки, записи журналов расчетов.

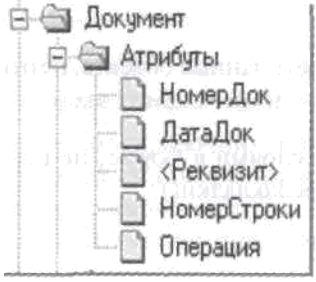
У документа существуют характеристики - дата и время, которые устанавливают строго определенную единую последовательность документов, которая должна отражать фактическую последовательность хозяйственных операций на предприятии, для обеспечения возможности корректного исправления и ввода новых документов задним числом. Для привязки документа к моменту времени при сохранении документа формируется его позиция на «оси времени». Позиция сохраненного документа уникальна. В момент ввода нового документа для программы он не имеет позиции, так как ещё не сохранен.

Сохраненный документ, в свою очередь может быть «проведенным» или «не проведенным». При проведении документа выполняются действия, описанные в модуле документа.

Документы, в отличие от справочников, не имеют формы списка и помещаются в определенный журнал документов, а также в журнал «Полный», содержащий все виды документов. Информация заносится в документ при заполнении реквизитов, которые могут находиться в шапке (заголовочной части) документа (относиться ко всему документу в целом) или в табличной части документа (перечень строк с однотипной информацией).

Атрибуты документов

Дня документов любого вида характерны атрибуты, например: дата и номер. Ввод атрибутов обеспечивается системой автоматически, наименования идентификаторов атрибутов не изменяются.



НомерДок и ДатаДок — это атрибуты шапки любого Документа. Тип ДатаДок — Дата. Тип НомерДок можно задать — число или строка.

НомерСтроки — это атрибут типа число многострочной части документа.

Атрибут Операция определен только для документов Бухгалтерского учета и возвращает ссылку на операцию созданную по документу.

Атрибут Операция определен только для документов Бухгалтерского учета и возвращает ссылку на операцию созданную по документу.

Реквизиты это дополнительные свойства документа, создаваемые при конфигурировании. Реквизиты можно создавать в шапке и в многострочной части документа.

8.1. Предопределенные процедуры модуля формы документа

Для модуля формы документа существует перечень предопределенных процедур:

ВводНового - процедура, которая отрабатывает в момент начала ввода пользователем нового документа.

ВводНаОсновании - процедура, которая отрабатывает в момент начала ввода пользователем данного документа на основании другого документа.

ПриЗаписи - процедура, которая отрабатывает в момент записи документа.

ПриНачалеРедактированияСтроки - процедура, которая отрабатывает в момент начала редактирования существующей строки в табличной части документа.

ПриВводеСтроки - процедура, которая отрабатывает в момент ввода пользователем новой строки табличной части документа.

ПриРедактированииНовойСтроки - процедура, которая отрабатывает в момент начала редактирования новой строки табличной части документа (после процедуры *«ПриВводеСтроки»}*.

ПриУдалении Строки - процедура, которая отрабатывает в момент удаления пользователем одной из строк табличной части документа.

8.2. Подчиненные документы

Система «1С:Предприятие» позволяет устанавливать между документами отношения подчиненности типа «много ко многим». Использование механизма подчиненности дает пользователю возможность автоматически Формировать список документов, подчиненных выбранному документу. Кроме того, к списку подчиненных документов можно обращаться на встроенного

языка системы «1С:Предприятие».

Чтобы сделать какой-либо документ (назовем его условно «Документ2») подчиненным другому документу («Документ 1»), необходимо:

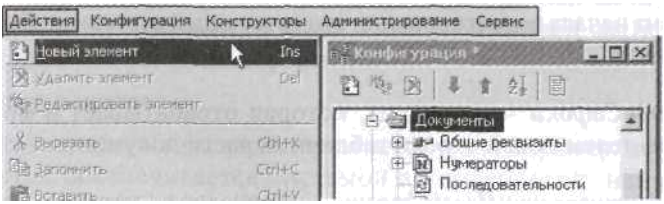
- при конфигурировании в структуре документа «Документ2» создать реквизит типа «документ», указав конкретный вид документов или «документ вообще»;
- при использовании конфигурации при вводе документа «Документ2» в качестве значения этого реквизита выбрать документ «Документ1».

Говорят, что Документ2 подчинен Документу 1, если Документ1 упомянут в каком-либо реквизите Документа2.

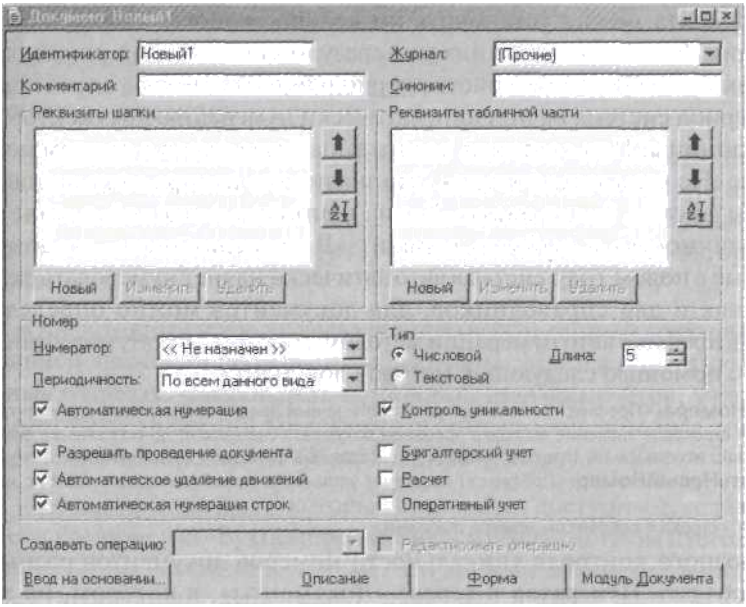
Для работы со списком документов, подчиненных какому-либо документу, используется журнал специального типа - журнал подчиненных документов.

8.3. Создание нового вида документов

Упражнение 13. Создадим документ «Приход» (Приходная накладная), который будет отражать хозяйственную операцию оприходования оборудования. В дальнейшем при проведении он будет изменять состояние регистра остатков и бухгалтерских итогов, а также формировать записи журнала расчетов,.



Выберем пункт «Новый элемент» из меню «Действия» главного меню Конфигуратора. Если снят флаг использования конструкторов (см. рис.) на экране появляется окно с заголовком «Документ Новый 1».



Как и в других окнах с полями ввода, в данном окне можно переме-щаться с помощью мышки или клавиш Tab и Shift+Tab.

Свойства документа

Прежде всего, для нового вида документов нужно указать *идентификатор.* Идентификатором объекта является некоторая уникальная последовательность символов. Если введенный Вами идентификатор совпадет с идентификатором уже существующего документа, то программа сама предупредит вас об этом. Зададим нашему документу идентификатор «Приход» и синоним - Приходная накладная. Далее можно посмотреть на различные свойства документа, установленные системой при его создании и изменить те из них, которые Вас не устраивают. Необходимо отнести Документы данного вида к определенному *журналу.* Щелкнув мышью в по-ле выбора «Журнал», Вы можете выбрать один из существующих журналов. Это означает, что документы дан-ного вида будут вводиться и просматриваться в этом журнале. В процессе конфигурирования можно создавать и новые Журналы. Если Вы работаете на платформе V7.5 и хотите для вновь созданного Документа определить соответствующий новый журнал, то сначала создайте этот журнал (пустой) или выберите для вновь созданного документа журнал «Прочие», создайте новый журнал и переопределите журнал в документе, на платформе V7.7. с помощью конструктора документов можно создать новый

журнал в процессе создания нового документа. Создадим журнал «Оборудование» для данного типа документа.

Нумерация документов

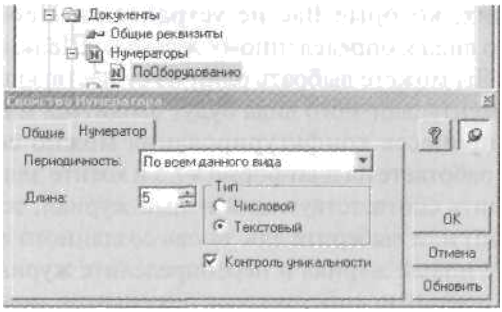
Для документа можно установить несколько свойств, относящихся к его нумерации. Большинство из них уже сразу имеют значения, подходящие для создаваемого документа. Свойство периодичности номеров определяет, через какой период система начнет автоматически нумерацию документов данного вида с единицы. При создании нового типа документов в данном поле стоит значение «По всем данного вида». Это значит, что номера документов данного вида все время будут расти. Если щелкнуть мышкой в поле выбора «Периодичность» и выбрать вариант «В пределах года», то документы, введенные в новом году, система автоматически начнет нумеровать с единицы. Так же как и для справочников, для документов можно определить или изменить префикс авто нумерации методом ПрефиксАвтоНумерации (?,""), а также с помощью следующих методов документа:

ПрефиксНомера(<Префикс>) позволяет установить новый префикс номера для документа.
Возвращает строковое значение текущего префикса документа (на момент до исполнения метода).
Метод можно использовать только для объектов, созданных функцией СоздатьОбъект.
УстановитьНовыйНомер(<Префикс>) позволяет установить новый номер документа с заданным префиксом.
<Префикс> - строка с префиксом номера документа.

Для сквозного контроля уникальности номеров документов разных видов можно создать **Нумератор** в дереве «Документы», в котором, на закладке Нумератор, задаются свойства нумерации документов.

Контроль уникальности. Если эта опция включена, то при вводе нового документа его номер проверяется на уникальность в пределах, установленных в реквизите «Периодичность». После завершения периода, установленного в реквизите «периодичность», нумерация документов начнется с 1.

Упражнение 14. Создадим нумератор - «ПоОборудованию», для нумерации документов, участвующих в движении оборудования, и зададим контроль уникальности в нумераторе, чтобы наши документы имели единую нумерацию.



После этого можно определить нумерацию документа по нумератору, указав его в поле нумератор свойств документа.

Проведение документов

Признак «Разрешить проведение» указывает системе, что данный документ можно не только сохранить в электронном виде и распечатать, но и создать документом записи в учетных базах. Если признак не установлен, то у документа не будет модуля документа, в котором описывается алгоритм его проведения.

Признак «Оперативный учет» указывает системе, что данный документ может создать записи движений по регистрам, в которых ведётся оперативный учёт в различных разрезах: товаров, мат. ценностей, взаиморасчётов, учёт партий товаров. Алгоритм движений по документу описывается в модуле документа в предопределённой процедуре *ОбработкаПроведения()*.

Признак «Бухгалтерский учет», который становится доступным, если системе установлена компонента «Бухгалтерский учет» и создан хотя бы пустой объект типа «ПланСчетов», указывает системе, что данный документ может иметь бухгалтерскую операцию, запи-сывать проводки, и, таким образом, отражать в бухгалтерском учете, ту хозяйственную операцию, которую введенный документ описывает. При этом в нижней части окна редактирования становится доступен признак «Создавать Операцию по документу». Для простых документов рекомендуется оставить предложенное значение «Всегда». В этом случае документы данного вида всегда будут иметь операцию. Другие значения признака «Создавать Операцию по документу» позволяют регулировать создание бухгалтерской операции по документу. Например, если установить значение «Только при проведении», бухгалтерская операция будет записываться по документу только в процессе проведения документа. Эти возможности используются при большом количестве документов и в сложных Конфигурациях. Алгоритм бухгалтерской операции также описывается в модуле документа в предопределённой процедуре *ОбработкаПроведения()*. Само по себе, включение документа в журнал операций влияет только на возможность просмотра журнала.

Если установлен признак «Расчет»,- документы данного вида смогут создавать записи журнала расчётов в процессе проведения документа.

Создание структуры Документа

Для того, чтобы документ мог содержать полезную информацию, характерную

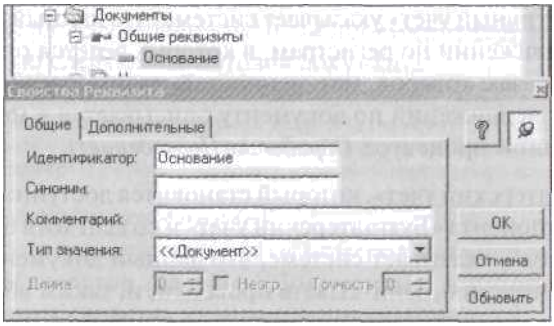
Именно для данного документа, мы должны определить в окне редактирования

набор *реквизитов.* Для ввода нового реквизита следует нажать кнопку «Новый» в группе «Реквизиты шапки».

Можно создать несколько *общих реквизитов,* которые будут характерны для любого вида документов - и существующих, и созданных в последствии.

Упражнение 15. Создадим общий реквизит - «Основание», тип значения -«Документ», который позволит любой документ сделать подчиненным другому документу.

Для этого откроем дерево метаданных на ветви «Документы» - «Общие реквизиты» - меню «Действие» - «Новый элемент». При этом на экране появляется окно *палитры свойств* - «Свойства реквизита». Далее новый общий реквизит документов вводится аналогично реквизиту справочника.



В структуре документа следует отражать только те данные, которые будут использоваться. В окне настройки вида документа расположены два поля: для ввода реквизитов шапки и для ввода реквизитов табличной части.

Реквизиты шапки могут хранить только одно значение, по каждому документу данного типа. В нашем примере, так как реквизиты «ДатаДок», «НомерДок» и «Основание» вводятся системой автоматически, другие реквизиты шапки нам не потребуются.

Кроме этого, в документе может присутствовать одна табличная часть. Реквизиты табличной части могут хранить множество значений, по каждому документу данного типа вводиться, соответственно, для каждой строки документа.

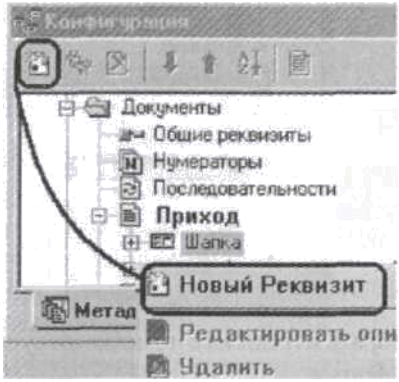
Если имеется бланк документа (накладной, счета), то из него видно, какие реквизиты будут относиться к табличной части, то есть являться колонками в многострочной части документа.

Ввод нового реквизита табличной части.

Упражнение 16. Создадим реквизиты табличной части - «Оборудование», тип значения - «Справочник.Оборудование», «Ед», тип значения -

«Справочник.Единицы», «Количество», «НовЦена» и «Сумма» », тип значения - Число. В реквизите «Сумма» на закладке «Дополнительные» установите флажок «Итог по колонке».

Для ввода нового реквизита в табличную часть следует нажать кнопку «Новый» в группе «Реквизиты табличной части». Ввод реквизита в табличную часть ничем не отличается от ввода других реквизитов. Конфигуратор позволяет вводить новые или редактировать существующие реквизиты в окне «Метаданные». Для этого в «дереве метаданных» щелкните левой кнопкой мыши плюс «+» слева от нужного документа, появятся две ветви «Шапка» и «Табличная часть», установите курсор на нужную ветвь и щелкните правую кнопку мыши, появится меню, в котором первым пунктом стоит «Новый Реквизит».



Щелкнем левой кнопкой мыши пункт меню «Новый Реквизит». В появившейся палитре свойств в поле «Идентификатор» укажем «Оборудование». По умолчанию новому реквизиту присваивается тип «Строка», выберем тип значения «Справочник.Оборудование» ниже в списке. Аналогично создадим реквизит «Единица» типа «Справочник. Единицы».

Аналогично введем числовые реквизиты «Количество», «НовЦена» и «Сумма» -, для реквизита «Сумма» на закладке «Дополнительные» палитры свойств установим флаг «Итог по колонке».

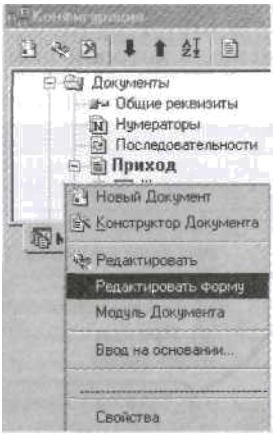
Только те реквизиты табличной части документа, для которых установлен флаг «Итог по колонке», могут быть выбраны для показа в графе журнала документов, также по этим реквизитам можно будет получить итоговую сумму, вызвав метод «Итог(«Сумма»);».

Укажем для числовых реквизитов длину и точность. «Цену» и «Сумму» будем вводить с копейками.

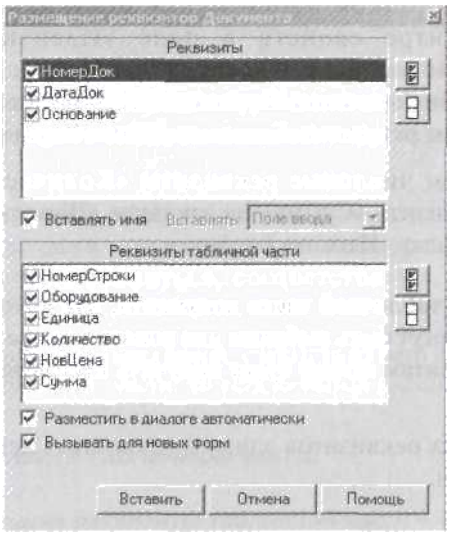
Редактирование формы документа

Упражнение 17. Создадим Форму для интерактивного редактирования документа аналогично тому, как мы создавали формы для справочников.

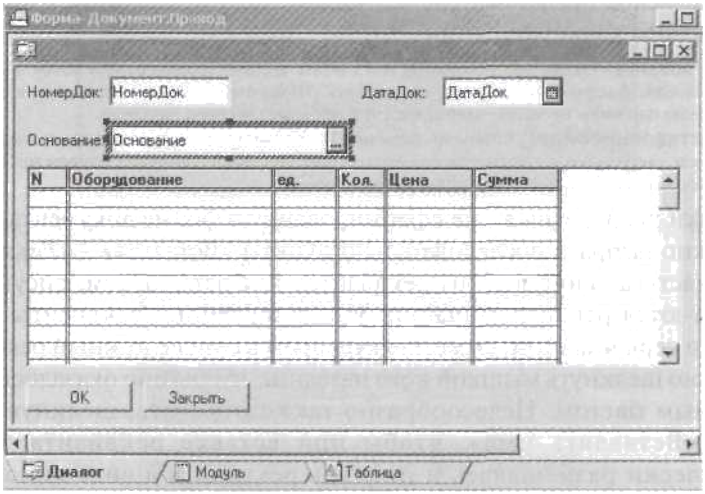
Нажмите кноп-ку *«Форма»* в окне редактирования документа или установите курсор на нужный документ в «Дереве метаданных», щелкните правой кнопкой мыши и, в появившемся меню, выберите пункт «Редактировать форму».



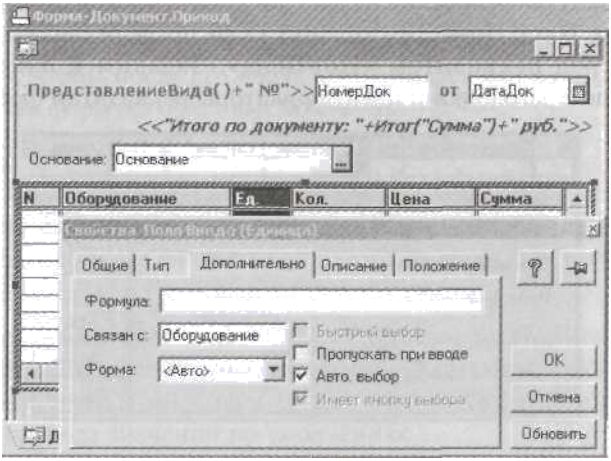
Система предложит выбрать из списка с пометками существующие реквизиты документа для ввода в форму,



после нажатия кнопки «Вставить» будет открыто окно *редактора форм* с введенными реквизитами документа.



Справочник Единицы у нас подчиненный, поэтому при выборе его значения необходимо определить значение элемента справочника владельца, для этого на закладке дополнительные панели свойств заполним поле «Связан с» ссылкой на элемент справочника владельца - «Оборудование», который выбирается в строке табличной части документа.



Метод **Итог(<"ИмяРеквизита">)** возвращает сумму значений реквизита по всем строкам табичной части документа.<ИмяРеквизита> - имя реквизита табличной части документа (в кавычках).

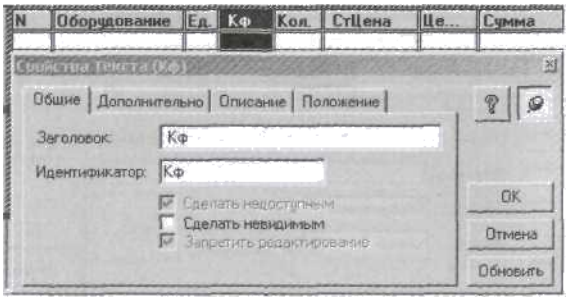
Метод можно использовать только для реквизитов табличной части документов с установленным свойством 'Итог по колонке'.

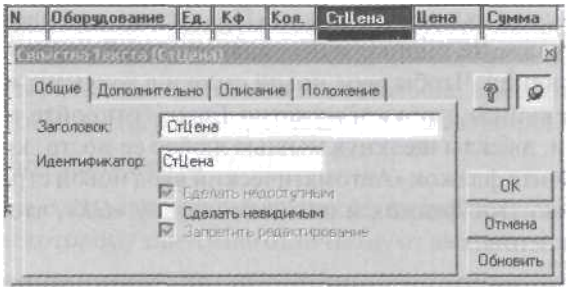
Для большинства видов метаданных определены методы, позволяющие установить или получить название (идентификатор) и получить пользовательское представление (синоним) вида метаданных в виде строки, как они заданы в конфигураторе.

Метод **Вид(<Название>)** позволяет установить или считать текущее название вида объекта метаданных. Возвращает текущее название вида объекта метаданных.<Название> - строка с названием вида объекта метаданных. Если параметр не задан - метод просто возвращает текущее название. Метод **ПредставлениеВида()** позволяет получить пользовательское представление вида объекта метаданных, как оно задано в конфигураторе.Возвращает строковое значение, содержащее пользовательское представление вида объекта метаданных.

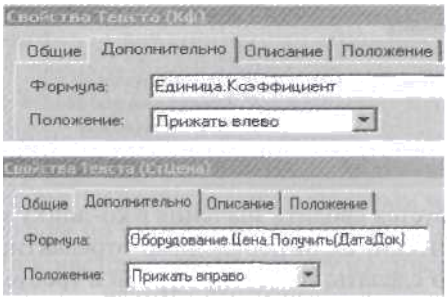
Для вставки реквизитов в уже сформированную форму документа в главном меню нужно выбрать последовательно пункты «Вставить» - «Реквизиты». В верхней части диалога выбора реквизитов для вставки (см. рисунок выше) располага-ются реквизиты шапки. Уже вставленные реквизиты отмечены галочкой и серым цветом. Отметьте курсором в списке нужный реквизит. Для этого нужно щелкнуть мышкой в его название, чтобы оно оказалось отмечено выделенным цветом. Целесообразно также отметить, щелкнув мышкой, флажок «Вставлять имя», чтобы при вставке реквизита в диалоге автоматически размещалась и подпись реквизита в виде синонима или идентификатора. Теперь нужно щелкнуть мышкой кнопку *«Вставишь».* При этом, диалог закрывается и курсор мыши принимает специальный вид, показывающий, что сейчас будет производиться вставка реквизитов в окно диалога.

Покажем в табличной части документа коэффициент выбранной единицы измерения и цену оборудования на дату документа - реквизиты формы «Кф» и «СтЦена» типа Текст. Поместим колонку «Кф» после реквизита «Единица», а «СтЦена» перед реквизитом «НовЦена». Зададим в панели свойств соответствующие заголовки и идентификаторы реквизитам формы.





На закладке Дополнительные панели свойств зададим формулу, по которой будут вычисляться значения в этой колонке.



Так как мы задали идентификаторы реквизитам формы, то по этим идентификаторам мы сможем обращаться к значениям этих реквизитов, в контексте формы документа.

Введём алгоритм вычисления реквизита «Сумма» в формулу реквизита «Количество», чтобы после его ввода выполнилась формула расчета суммы. В нашем случае формула будет выглядеть так:

```
Сумма=Количество*Кф*НовЦена;
```

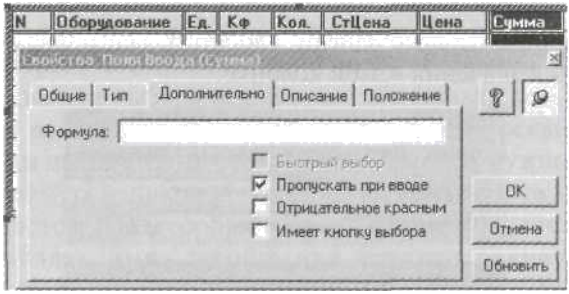
Действие этой формулы будет заключаться в том, что в реквизит «Сумма» будет занесено значение выражения справа от знака равно. После ввода формулы следует нажать кнопку «Обновить» в палитре свойств, для подтверждения ее ввода.

Эту же формулу можно внести в поля «Единица» и «НовЦена», чтобы при Редактировании этих полей сумма обновлялась автоматически. Для удобства Редактирования строки документа можно в поле «Оборудование» прописать операторы для ввода значений по умолчанию:

```
Количество=1; НовЦена=СтЦена;
```

Теперь, чтобы после ввода новой цены и количества сразу начать вводить новую строку, вызовите палитру свойств для колонки «Сумма», дважды Щелкнув в заголовок колонки «Сумма», затем перейдите к закладке

«Дополнительные» и включите флажок «Пропускать при вводе», щелкнув по нему мышью, нажмите кнопку «ОК» палитры свойств, чтобы подтвердить сделанные изменения. Чтобы ввод новой строки в документ начинался после ввода цены в текущую строку и нажатия Enter, откройте палитру свойств табличной части, дважды щелкнув мышью любое ее место, кроме заголовков колонок и включите флажок «Автоматический ввод новой строки». (Для этого щелкните мышью этот флажок и нажмите кнопку *«ОК»,* чтобы подтвердить сделанные изменения.)



Теперь нам не потребуется вводить значения в ко-лонку «Сумма» - программа делает это автоматически. Если необходимо отредактировать вычисленные значения, это можно сделать, поместив курсор в требуемую ячейку и нажав клавишу Enter. Заметим, что условие «Пропускать при вводе» срабатывает только при вводе новой строки: если выполняется редактирование существующей строки, то все работает обычным образом: при нажатии Enter ячейка переключается в режим редактирования, повторное нажа-тие Enter выключит режим редактирования, но курсор останется на месте - никакого перехода не будет.

Установим признак «Пропускать при вводе» для колонки «N», содержащей номер строки документа. В этом случае после ввода новой строки колонка «N» будет пропущена, а в режим редактирования будет переключена следующая ячейка в строке для ввода значения, в этой ячейке автоматически откроется соответствующий справочник, журнал, калькулятор или календарь в зависимости от типа реквизита.

Заметим, что в 7.7 при установленном флажке автонумерации строк в свойствах документа редактировать номер строки вручную нельзя, для этого существуют стрелки на панели инструментов.

Теперь зададим заголовок окна диалога. При создании диалога документа заголовок был оставлен пустым, поэтому при работе с этим документом в режиме запуска «1С:Предприятие» заголовок будет формироваться программой автоматически из идентификатора; кроме этого, в заголовок будет добавлен номер документа (или слово «Новый», если вводится новый документ). Для задания заголовка окна диалога следует вызвать палитру

свойств диалога, дважды щелкнув мышью в пустое серое пространство диалога. В поле «Заголовок» закладки «Общие» палитры свойств следует ввести текст, который будет выводиться в заголовке окна диалога и нажать кнопку *«ОК»,* чтобы подтвердить произведенные изменения.

Сортировка строк документа

В некоторых случаях у пользователей возникает потребность сортировки строк документов по некоторому значению отличному от введенного номера строки.

Кроме того, в модуле формы документа отсутствует возможность изменения номеров строк. Для решения данной задачи можно предложить сортировать строки документа в процессе печати. Для этого целесообразно использовать список значений. (В 7.7 существует метод документов сортироватьстроки (<колонки>) для сортировки строк документа по значениям некоторых реквизитов табличной части.) Вначале в список значений заносятся в качестве значений номера строк документа. При этом в качестве представлений значений указывается строковое значение, которое будет определять порядок сортировки, например, наименование товара. Далее список значений сортируется представлению штатным методом. После этого организуется цикл печати по списку значений. Причем позиционирование строки документа выполняется по полученному из списка значений номеру.

8.4. Создание печатной формы документа

В Конфигураторе создается не сама печатная форма, а только «заготовка» для её построения, состоящая из 2-х частей: *шаблона печатной формы* и алгоритма, описывающего порядок построения формы. Непосредственно сама печатная форма документа создается при работе документа в режиме «1С:Предприятие».

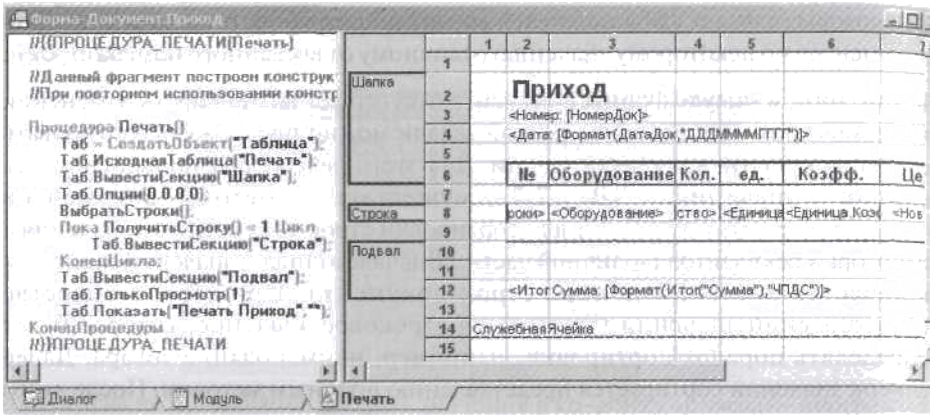
Табличный редактор располагается в закладке «Таблица». Для ее активизации нужно щелкнуть в эту закладку. Вообще, форма может иметь несколько табличных редакторов, но при ее создании в форму автоматически включается только один.

Упражнение 18. Создадим табличный шаблон и алгоритм создания печатного Документа по шаблону с помощью конструктора печати.

Конструктор печати документа можно запустить из меню Конструкторы при активном модуле формы документа. Работа с конструктором печати подробно описана во 2 т. Конфигурирования и администрирования 1С:Предприятия 7.7, стр. 201-209.

В результате работы конструктора мы получим текст процедуры печати, Табличный шаблон и кнопку в окне диалога. Конструктор позволяет отформатировать выражения в ячейках, используя метод Формат (<Форматируемое

значением <«Форматная строка»>) . Форматную строку в ячейках табличного шаблона можно задать непосредственно, не используя метод Формат(). Конструктор печати не использует идентификаторы реквизитов формы, в нашем примере — «Кф», и не видит у какого числового реквизита табличной части установлен «Итог по колонке».



Основные свойства табличного редактора

В системе «1С:Предприятие» в глобальном контексте и для объектов имеющих форму предусмотрен редактор таблиц, внешне похожий на Excel, в котором создаются шаблоны печатных форм. Информация, предназначенная для печати, располагается в ячейках шаблона и группируется по горизонтальным (и вертикальным) секциям. Вывод из исходного шаблона таблицы в результирующую таблицу осуществляется программно по секциям шаблона при помощи свойств служебного объекта «Таблица». Ячейки шаблона могут выполнять четыре типа преобразований: **Текст** - воспринимает символы в ячейке как строку текста, **Выражение** - воспринимает символы в ячейке как текст на встроенном языке, который транслируется и вычисляется, а затем преобразуется в текстовую строку результирующей таблицы, причем все незначащие символы отбрасываются, **Шаблон** и Фикс.шаблон позволяют в одной ячейке выводить и текст, и значение вычисленного выражения, текст, который должен быть проинтерпретирован как выражение берется в квадратные скобки []. **Фиксированный шаблон** позволяет задать определенную длину для значения выражения в квадратных скобках равную количеству знакомест между скобками. Для ячейки можно задать ширину столбца и высоту строки. Единицей измерения этих параметров является количество символов шрифта по умолчанию для табличных документов, последний задается в меню Сервис - Параметры -Интерфейс.

Оформление заголовка печатной формы

Прежде всего, введем заголовок документа. Заголовок будет состоять, очевидно, из названия документа, номера документа и даты документа. Для ввода заголовка щелкните мышью в ячейку в верхней части. Содержимое ячейки может быть оформлено различным образом: изменяются начертание и размер текста, обрамление ячейки, цвет фона и текста. Все эти изменения выполняются при помощи палитры свойств ячейки. Для вызова палитры свойств ячейки выберите пункт «Свойства» меню «Действия» главного меню Конфигуратора. Откроем палитру свойств данной ячейки. В появившейся палитре свойств в закладке «Текст» нам нужно изменить значение поля «Тип». Щелкнем в него мышкой и выберем строку «Шаблон». (Этот тип позволяет совместить в одной ячейке и обычный текст, и одно или несколько выражений. Для того, чтобы выделить выражения в тексте ячейки, нужно просто заключать их в квадратные скобки). Теперь мы указали, что в данной ячейке будет располагаться не обычный текст, а выражение типа Шаблон на встроенном языке. Результат этого выражения при построении печатной формы документа будет выводиться в ячейке вместо самого выражения. Если заголовок документа это просто текст, то номер и дата документа зависят от конкретного документа, то есть они будут меняться. Для отражения данных документа в ячейку должно быть внесено выражение. Как мы уже отмечали, у документа есть два предопределенных реквизита -«Дата» и «Номер», которые мы не вводим при создании вида документа, но они всегда существуют. Они автоматически были внесены в диалог формы ввода документа. Для использования в выражениях для этих реквизитов существуют спе-циальные идентификаторы *ДатаДок* и *НомерДок*.

Установите курсор в окно редактирования и наберите на клавиатуре следующую строку:

```
[ПредставлениеВида()]№ [НомерДок] от [ДатаДок #ДДДММММГГГГ]
```

и нажмите клавишу Enter, чтобы подтвердить введенный текст. Таким образом, в заголовок документа введены — строка названия документа и, в прямоугольных скобках, выражения, значения которых будут вставлены в символьную строку. Дата документа отформатирована в виде: 2 цифры - число, затем месяц прописью и 4 цифры - номер года (другие возможные значения форматной строки для различных базовых типов данных смотрите в I томе «Описания встроенного языка» », стр. 80-82 - метод «Формат»).

Разместим в печатном шаблоне реквизит «Основание». Введём поясняющий текст и выражение для вывода значения реквизита. Выберем ячейку (щелкнем в нее мышкой). Наберем на клавиатуре поясняющий текст «Основание». Разумеется этот текст может быть произвольным. Закончим ввод текста нажатием клавиши Enter. Теперь активизируем ячейку правее и введем в нее выражение. Для этого вызовем палитру свойств и изменим значение поля

«Тип». Щелкнем в него мышкой и выберем строку «Выражение». В качестве

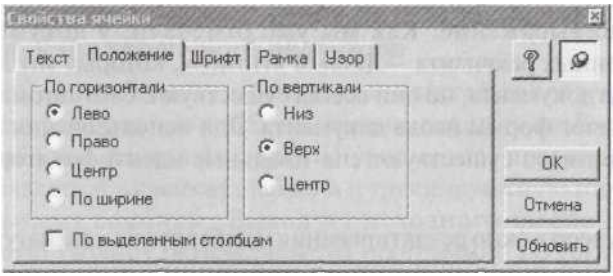
выражения введем слово «Основание», которое в данном случае является идентификатором общего реквизита (его мы создавали, когда разрабатывали структуру документа) и не может быть другим. Закроем палитру кнопкой *«ОК»*.

Создание табличной части печатной формы документа

Реквизиты табличной части документа вводятся в таблицу так же, как и реквизиты шапки, с той лишь разницей, что выражение будем размещать не справа от поясняющего текста, а на следующей строке, в другой секции. Фактически, сформирована таблица, которая повторяет табличную часть документа. Заголовки граф этой таблицы схожи с заголовками граф табличной части документа, а содержимое строк - это выражения для вывода реквизитов табличной части документа.

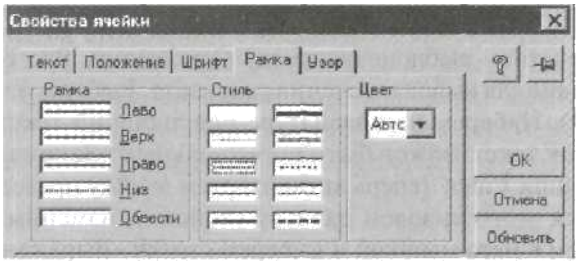
В качестве первого реквизита в форме расположен специальный реквизит, содержащий номер строки. Так же как номер и дата документа, реквизит «Номер строки» автоматически создается системой в документе, если существует хотя бы один реквизит табличной части. Его имеет смысл расположить в печатной форме, чтобы строки в ней были пронумерованы.

Выберите закладку «Положение», щелкнув мышью ее название.



Управляющие элементы этой закладки позволяют задавать вертикальное и горизонтальное выравнивание текста в ячейке. Зададим положение текста в середине ячейки, для этого щелкнем мышью слово «Центр» в группе «По горизонтали» и «По вертикали».

Изменим размер и атрибуты текста ячейки. Для этого используем закладку «Шрифт». Установим в закладке размер шрифта 10 и включим атрибут «жирности». Теперь перейдем к закладке «Рамка».


Свойства ячейки
Текст
Положение
Шрифт
Рамка
Узор
Рамка
Лево
Верх
Право
Низ
Обвести
Стиль
Цвет
Авто
OK
Отмена
Обновить

Управляющие элементы этой закладки предназначены для создания рамки вокруг ячейки или группы ячеек. На этой закладке выберем стиль рамки, щелкнув мышью нужный образец в группе «Стиль». В группе «Рамка» выберем вид рамки, точнее, вдоль каких границ ячейки будет идти рамка. Если мы щелкнем мышью поле «Обвести» - рамка будет идти вокруг ячейки.

Расположим следующие поясняющие тексты и выражения:

№ п/п	Оборудование	Кол.	Ед.	Цена	Сумма
<НомерСтроки>	<Оборудование>	<Количество*Кф>	<Единица>	< НовЦена>	<Сумма>

При установке свойств ячеек не забывайте для выражений устанавливать тип ячейки «Выражение», при этом текст в ячейке будет заключен в угловые скобки.

Табличный редактор позволяет легко изменять ширину колонок. Удобнее всего это сделать, перетащив мышью разделитель между колонками таблицы в верхнем заголовке (сером поле с номерами колонок). Похожую операцию мы проделывали, когда создавали диалог для ввода документа.

Очень важно, чтобы реальное содержимое ячейки таблицы соответствовало тому типу, который установлен для ячейки в поле Тип «Свойств ячейки», текст, выражение, шаблон. Несоответствие может вызывать различные неприятные эффекты. Например, если для ячейки с выражением установлен тип «Текст», то при построении печатной формы выражение будет выведено в форму, как обычный текст. То есть, вместо ожидаемых цифр в форме мы увидим какой-то непонятный текст.

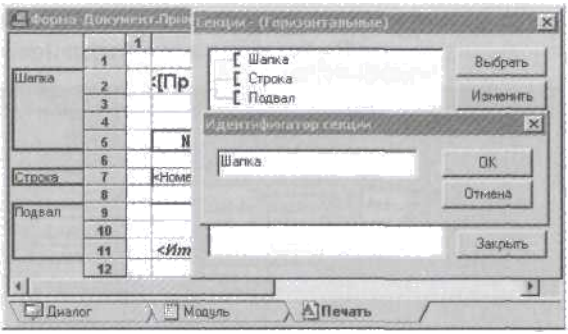
И наоборот: если для текстовой ячейки установлен тип «Выражение», то при построении печатной формы программа совершенно логично будет считать текст в такой ячейке выражением и попытается его вычислить. Результатом такого «вычисления» будет, скорее всего, сообщение об ошибке. К счастью, подобные «эффекты» легко обнаружить и устранить.

Создание секции

Выберем из меню «Таблица» пункт «Включить секцию».

Для вывода в печатную форму повторяющихся строк предусмотрен механизм *секций.* То есть вывод на печать табличного документа будет выполняться не сразу, а отдельными секциями.

Секция является частью табличного документа. Секции бывают вертикальные и горизонтальные. Горизонтальная секция является совокупностью нескольких строк; как минимум, секция состоит из одной строки. Фактически, секция нужна только для того, чтобы назвать одну или несколько строк некоторым именем. Для создания секции выделите строки или столбцы табличного шаблона. Выделенные строки закрашиваются инверсным цветом. В главном меню Конфигуратора выберем позицию «Таблица» пункт «Включить в секцию». Появится диалог, в котором нужно указать имя секции. Изменим предложенное имя «Секция 1» и нажмем кнопку *«ОК».* В левой части окна появится обозначение выделенной нами секции.


Секции - (Горизонтальные)
Шапка
Строка
Подвал
Выбрать
Изменить
Идентификатор секции
Шапка
OK
Отмена
Закрыть
Диалог
Модуль
Печать

Описание алгоритма печати документа

Для манипулирования табличным документом в языке существует служебный объект «Таблица».

```
Процедура Печать()

      Таб = СоздатьОбъект("Таблица");

      Таб.ИсходнаяТаблица("Печать");

      Таб.ВывестиСекцию("Шапка");

      Таб.Опции(0,0, Таб.ВысотаТабпицы(),0);

      ВыбратьСтроки();

      Пока ПолучитьСтроку() =1 Цикл

            Таб.ВывестиСекцию("Строка");

      КонецЦикла;

      //          Пропись("*.spl"); // Определение файла прописи

      Таб.ВывестиСекцию("Подвал");

      Таб.ТолькоПросмотр(1);

      Таб.Показать("Печать документа "+ПредставлениеВида()+"
      "+НомерДок,"");

КонецПроцедуры
```

Дадим пояснения по строкам.

Строка *Таб=СоздатьОбъект («Таблица»)* создает в модуле объект языка типа «Таблица». Он обязательно должен быть присвоен некоторой формальной переменной, которая в дальнейшем и будет использована для управления этим объектом. В отличие от реквизита документа, формальные переменные не хранятся в информационной базе, а «живут» до окончания процедуры. Использование объекта через создание переменной с некоторым именем позволяет использовать одном модуле несколько однотипных объектов. Итак, мы получили переменную Таб, которая содержит объект типа «Таблица».

Теперь мы можем выполнять различные действия над этим объектом, указывая методы этого объекта. Название метода записывается через точку после самого объекта и после него всегда ставятся круглые скобки. В скобках могут Указываться параметры. Опишем те действия, которые производятся с переменной Таб.

```
Таб.ИсходнаяТаблица(«Печать»);
```

В этой строке вызывается метод *ИсходнаяТаблица(«...»),* который назначает исходный табличный документ. Мы уже упоминали, что в форме может быть несколько табличных документов. Например, вдокументе «Счет» может быть 2 печатные формы: для печати рублевого счета и для печати валютного счета.

Метод *Исходная Таблица(«...»)* позволяет выбрать, какую печатную форму мы будем использовать в данной процедуре. Если в форме объекта всего один табличный документ, то вызов метода *ИсходнаяТаблица(«...»)* можно опустить.

```
Таб.ВывестиСекцию («Шапка») ;
```

Эта строка выполняет включение в результирующую таблицу подготовленной нами секции. Напомню, что мы выделили в табличном документе: две секции «Шапка» и «Строка».

```
Таб.Опции(0,0, Таб.ВысотаТаблицы(),0);

 Таб.ТолькоПросмотр(1);

 Таб.Показать("Печать документа "+ПредставлениеВида()+"
"+НомерДок,"");
```

Метод *Опции(,,,,)* отключает в готовом документе показ сетки ячеек и заголовков строк, определяет количество фиксированных строк и столбцов.

Метод *Таб.ВысотаТаблицы()* в данном случае устанавливает число фиксированных строк, равное числу строк секции «Шапка». Теперь эти строки всегда будут видны на экране, а многострочная часть таблицы, при изменении положения ползунка вертикальной линии прокрутки, будет, как бы, подворачиваться под Шапку.

Метод *ТолькоПросмотр(< ?>)* определяет, что печатная форма будет использоваться только для просмотра и печати, а не для редактирования.

Метод *Показать(<?>)* открывает сформированную форму печатного документа, чтобы ее можно было бы посмотреть и вывести на принтер. Если этот метод не вызывать, то мы при работе с данным документом так и не увидим печатной формы.

Вывод строк документа располагается после вывода шапки, но до вызова метода *Показать(<?>)*. Метод документа, начинающий обработку строк называется - *ВыбратьСтроки()*.

Для организации обхода (выборки) строк документа используется конструкция встроенного языка

```
Пока ПолучитьСтроку()=1 Цикл

    Таб.ВывестиСекцию(«Строка»);

КонецЦикла
```

Таким образом метод *ВывестиСекцию(«Строка»)* будет вызываться для каждой строки документа.

Чтобы «привести модуль в порядок», можно воспользоваться следующим приемом: выбрать из меню «Действия» пункт «Выделить все», а затем, из этого

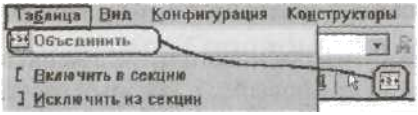
же меню, пункт «Форматировать блок». В результате все строки модуля будут выровнены в соответствии с принятым стандартом.

Редактирование в документе секции для печати итогов

В нижней части печатной формы документа находится область печати итогов по колонке «Сумма». Для записи выражения использован метод документа *Итог(«<Реквизит >»)* , который предназначен для получения суммы значений числового реквизита документа по всем строкам. В скобках в качестве параметра нужно указать в двойных кавычках идентификатор того реквизита, с установленным флагом «Итог по колонке», данные по которому суммируются. При печати денежных величин иногда необходимо выводить их прописью. Для этого используется форматная строка «ЧПДС». Если необходимо напечатать величину в альтернативной валюте, используется метод Пропись(<"Файл прописи.spl">) до установки форматной строки «ЧПДС».

Режим объединения ячеек табличного шаблона

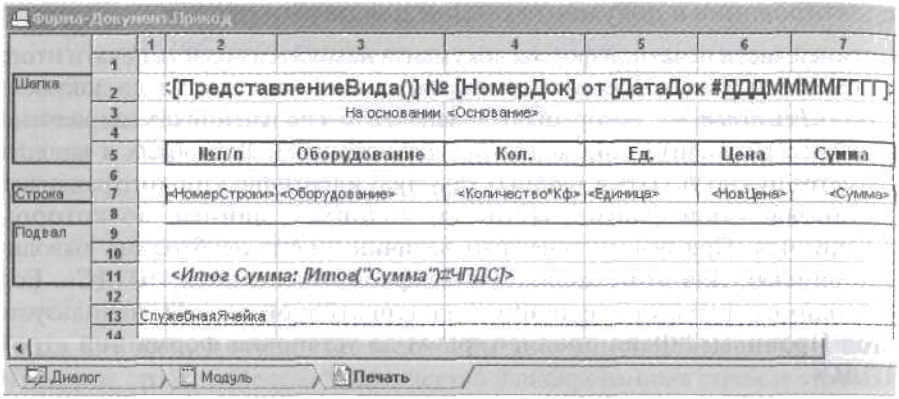
Разместим по центру заголовок документа. Сначала выделим область в строке, в которой расположен заголовок, начиная с его правой ячейки и до ячейки, расположенной над самой левой колонкой табличной части. Далее выберите из меню «Таблица» пункт «Объединить», чтобы выделенные ячейки воспринимались как одна область для центрирования.



Заголовок будет расположен по центру бланка. Теперь вызовите палитру свойств из меню «Действия» (пункт «Свойства»). В ней активизируйте закладку «Положение» и щелкнем мышью в поле «Центр» группы «По горизонтали», чтобы установить режим центрирования по горизонтали.

При расположении текста в пределах нескольких ячеек, объединяемых командой «Объединить», необходимо, чтобы текст размещался в крайней левой ячейке из группы объединяемых ячеек. Если это не так, то при выполнении команды

«Объединить» текст «пропадет». Увидеть «пропавший» текст можно, если вызвать палитру свойств для ячейки, в которой находится этот текст.



8.5. Включение документа в пользовательское меню

Упражнение 19. Создадим в пользовательском меню пункт в колонке «Документы» для вызова документа.

Для этого нужно раскрыть соответствующую ветвь в редакторе меню, щелкнув мышью знак «+» слева от слова «Документы». При этом раскрывается список документов в этой ветке.

Добавление в ветку «Документы» нового пункта меню выполняется двойным щелчком мышкой в элемент <новый...>, который располагается в самом низу ветки (если вся ветка не умещается в окне, следует воспользоваться вертикальной полосой прокрутки, чтобы увидеть этот элемент), название пункта меню, которое в итоге появится в меню «Документы» главного меню «1С:Предприятия.»

Редактирование свойств пункта меню

Прежде всего, в палитре свойств нужно выполнить выбор поля «Объект». Значение в этом поле показывает, с каким объектом метаданных будут выполняться действия при выборе пункта меню. Для выбора объекта нужно щелкнуть мышью в поле «Объект», и, используя полосу прокрутки, пролистать окно до строки с наименованием вводимого документа и щелкнуть по нему мышкой. В поле «Название» будет вставлена строка с наименованием вводимого документа.

Пока это название редактор сформировал сам, но его можно отредактировать. После того, как выбран «объект воздействия», необходимо выбрать команду, которую к этому объекту надо будет «применить». Для этого следует в поле «Команда» выбрать строку «Документ.<наименование вводимого

документах>.Ввести». Состав команд меняется в зависимости от того, какого вида объект выбран в поле «Объект». Например, для констант, справочников, журналов документов, отчетов доступна только команда «Открыть». Созданный нами пункт меню уже может вводить документ.

Для ввода нового документа можно создать кнопку на панели инструментов. Тогда для ввода нового документа требовалось бы всего лишь нажать кнопку, а не выбирать пункт меню, что заметно дольше. Можно назначить кнопке (и пункту меню) комбинацию «горячих клавиш», при нажатии на которые выполнялась бы команда, назначенная кнопке.

Во встроенном языке системы «1С:Предприятие» существует метод: *КнопкаПоУмолчанию()*. Эта команда позволяет указать, какая кнопка будет срабатывать при нажатии комбинации клавиш Ctrl+Enter. Если в модуль формы созданного нами документа ввести строку *КнопкаПоУмолчанию(«ОК»),* то при вводе документа нажатие комбинации клавиш Ctrl+Enter будет равносильно тому, как если бы мы щелкнули мышью кнопку «ОК» диалога.

8.6. Движения документа. Создание записей периодического реквизита Справочника по документу

Документы в системе «1С:Предприятие» являются инструментом для записи движений в учетные базы данных, в которых ведутся записи по различным разрезам учета. Эти записи вводятся только программно в модуле документа в предопределенной процедуре *ОбработкаПроведения()*.

Простейшим примером такой базы является совокупность записей периодических реквизитов справочников, создаваемая документами.

Для записи движений при создании документа необходимо в окне редактирования установить флажок «Разрешить проведение документа» и определить типы учетных баз, в которых данный документ будет создавать записи: Оперативный учет, Бухгалтерский учет или Расчет.

Модуль документа открывается из окна редактирования документа (не формы Документа) или через контекстное меню документа в дереве метаданных. Следует обратить внимание, что контекст модуля документа не связан с контекстом формы документа, поэтому в контексте модуля документа Доступны только атрибуты (реквизиты) документа.

Если в окне редактирования документа установлен флажок «Автоматическое

удаление движений», то при удалении или пометке к удалению документа движения созданные в учетных базах по данному документу удаляются.

Упражнение 20. Создадим в модуле документа алгоритм формирования записей истории периодического реквизита справочника «Оборудование» «Цена».

Новую цену оборудования мы задаем в приходном документе, определяя значение реквизита «ЦенаНов», поэтому создадим в модуле документа записи истории периодического реквизита справочника оборудование по каждой строке документа. Чтобы не записи в истории периодического реквизита справочника не дублировались, нужно проверить была ли изменена цена в документе.

Метод Установить**РеквизитСправочника**(ОлементСправочника>,<НазваниеРеквизита>,<Значение>, <ДатаУстановки>,<ИмяТипа>,<Длина>,<Точность>) позволяет записать значение периодического реквизита справочника с привязкой к проведению документа.
<ЭлементСправочника> - элемент справочника, в который будет запись;
<НазваниеРеквизита> - название периодического реквизита справочника;
<Значение> - новое значение периодического реквизита;
<ДатаУстановки> - дата установки нового значения периодического реквизита. Дата установки имеет смысл только для не оперативных документов.
<ИмяТипа> - необязательный параметр. Строковое выражение - название типа данных (или Вид субконто);
<Длина> - необязательный параметр. Число - длина числового или строкового значения;
<Точность> - необязательный параметр. Число знаков после десятичной точки.
Параметры <ИмяТипа>, <Длина> и <Точность> следует указывать при установке значения периодического реквизита справочника, имеющего неопределенный тип. Метод доступен только в Модуле документа в предопределенной процедуре ОбработкаПроведения.
Метод **ПривязыватьСтроку**(<НомерСтроки>) позволяет записывать номер строки документа в движениях документа. <НомерСтроки> - номер строки Документа.
Метод устанавливает номер строки для всех последующих движений регистров, при записи значений периодических реквизитов справочников с привязкой к документу, а также при записи бухгалтерских проводок. Метод доступен только в Модуле документа в предопределенной процедуре ОбработкаПроведения.

```
Процедура  ОбработкаПроведения()

    ВыбратьСтроки();

    Пока ПолучитьСтроку() =1 Цикл

        Если  НовЦенаОборудование.Цена.Получить(ТекущийДокумент())   Тогда

            ПривязыватьСтроку(НомерСтроки);

            УстановитьРеквизитСправочника (Оборудование , "Цена" , НовЦена) ;

        КонецЕсли;

    КонецЦикла;

КонецПроцедуры
```

8.7. Журналы документов.

Журналы документов - предназначены для просмотра и выбора документов, созданных в системе. Журналы документов могут иметь свои графы, значение которых заполняется из соответствующих реквизитов хранящихся в журналах документов. Это значительно облегчает возможность поиска нужного документа в журнале. Визуальное представление журнала задается в «Форме» журнала.

8.7.1. Виды журналов документов

В «1С:Предприятии»существуют следующие виды журналов документов:

Обычный журнал - в Обычном журнале может быть прописано несколько видов документов, но один и тот же вид документов не может быть прописан в разных Обычных журналах,

Общий журнал для всех документов, позволяет делать отбор по значениям общих реквизитов и граф отбора,

Дополнительный журнал служит для разделения по видам документов, как и обычный журнал, однако может быть прописан в разных Дополнительных журналах, в отличие от Общих журналов не позволяет делать отбор по значениям реквизитов,

Журнал подчиненных документов служит для работы со списком документов, подчиненных выбранному документу, может назначаться в конфигураторе для одного из Общих журналов, в противном случае создается системой автоматически и не редактируется, в пользовательском интерфейсе открывается через меню Действия для выбранного документа,

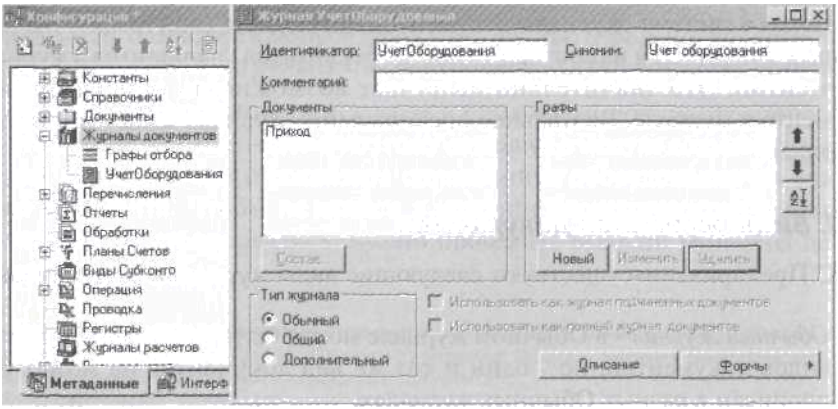
Полный журнал - общий журнал по умолчанию, может назначаться в конфигураторе для одного из Общих журналов, в противном случае создается системой автоматически и не редактируется,

Журнал «Прочие» - общий журнал для работы с документами, для которых не созданы свои обычные журналы, создается системой автоматически и не редактируется.

8.7.2. Создание вида документа в журнале документов

Для создания или редактирования журнала документов раскройте ветвь «Журналы документов» дерева метаданных, создайте нужный журнал, Щелкнув правой кнопкой мыши на любом ранее созданном журнале или на

названии ветви «Журналы документов» и выбрав пункт «Новый журнал» в появившемся меню.



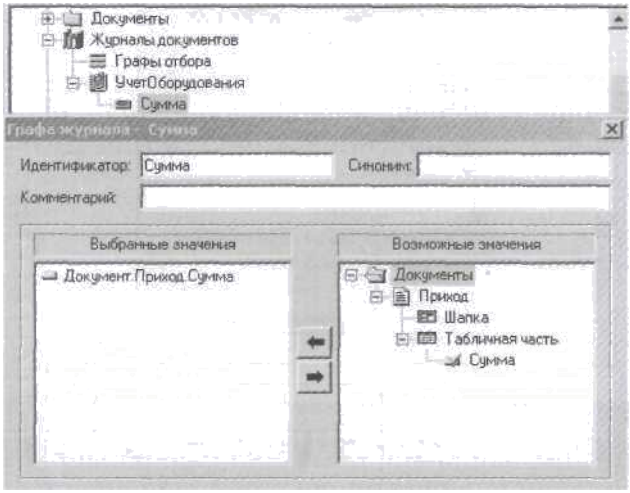
Чтобы посмотреть форму созданного журнала нажмите кнопку «Формы» и далее выберите пункт «Форма Списка» в появившемся меню.

1С:Предприятие автоматически создает 4 колонки в любом журнале: «Дата» - дата документа, «Время» - время документа, «Документ» -краткое наименование вида документа и «Номер» - номер документа. Какие-то из этих колонок в форме журнала могут отсутствовать - это значит, что в процессе конфигурирования их удалили из журнала. Кроме этих колонок, журнал может содержать произвольное число дополнительных колонок, называемых *графами,* для вывода в них любых других реквизитов документов, хранящихся в журнале. Состав этих граф, а также информация, которая будет выводиться в этих графах, полностью определяется в конфигураторе.

Новую графу журнала документов можно создать в окне свойств журнала (кнопка «Новый») или через контекстное меню пункт «Новая графа».

В существующем журнале, если щелкнуть мышью знак «+» слева от его идентификатора, откроется список граф журнала. Теперь, если дважды щелкнуть мышью идентификатор самой графы, будет вызвано окно для редактирования состава информации, отображаемой в этой графе. Создадим в нашем журнале графу «Сумма», в которой будем отображать итоговую сумму

по документу «Приход». Это возможно благодаря тому, что у этого числового реквизита табличной части документа установлен признак «Итог по колонке».



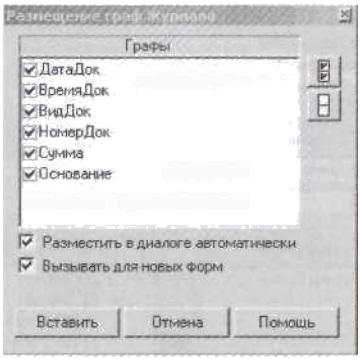
Для редактирования состава дополнительной графы журнала документов используются окна «Выбранные Значения» и «Возможные Зна-чения». В окне «Возможные значения» в виде дерева выдается список документов, для которых назначен редактируемый журнал. Последним документом в этом списке является последний созданный документ. Прочие документы уже существовали в конфигурации ранее. Если щелкнуть мышью знак «+» слева от идентификатора документа, а потом раскрыть ветви «Шапка» и «Таблица», мы можем увидеть идентификаторы реквизитов документа, который можно использовать для отображения в редактируемой графе. Окно «Выбранные Значения» содержит список реквизитов документов, уже назначенных для отображения в данной графе при работе с редактируемым журналом.

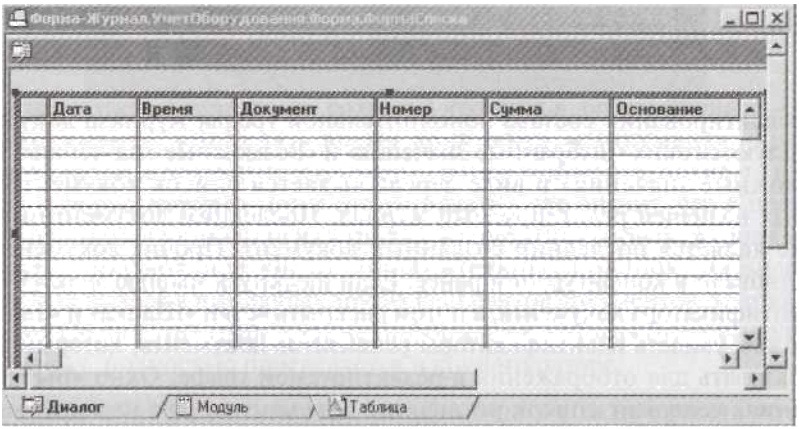
Исключить реквизит из показа в графе журнала можно, дважды щелкнув мышью его наименование в окне «Выбранные Значения».

Для вызова окна редактирования графы журнала следует дважды щелкнуть мышью ее название в дереве метаданных.

Для помещения реквизита документа в графу журнала дважды щелкнем его мышью в списке возможных значений и закроем окно редактирования графы, нажав кнопку *«ОК»*.

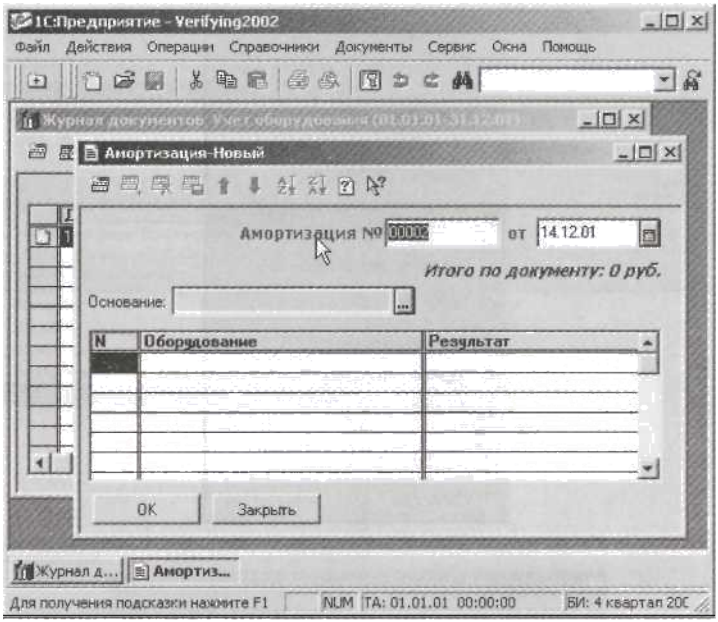
Создадим форму списка журнала документов. Это можно сделать, используя контекстное меню нашего журнала или его же окно свойств.





Упражнение 21. Создайте самостоятельно документ Амортизация, с реквизитами многострочной части Оборудование типа Справочник.Оборудование и Результат типа число, максимально до 9 999 999,99. Нумерацию документов этого вида задайте с помощью нумератора «ПоОборудованию». Покажите документ в журнале УчетОборудования, итог по колонке Результат должен отображаться в графе Сумма. Создайте форму документа и процедуру печати

документа. В форме документа покажите итог по колонке «Результат» в виде текстового реквизита формы.

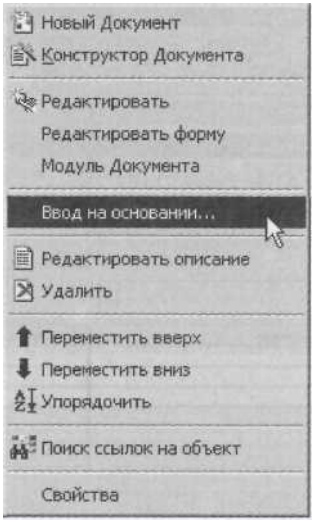


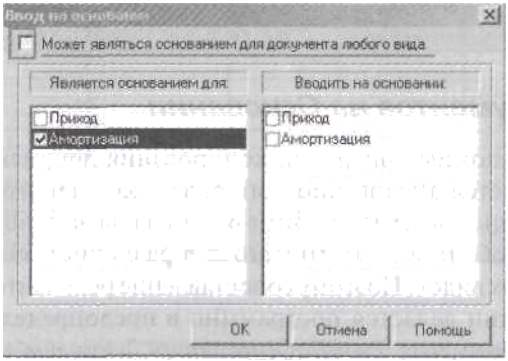
8.8. Ввод документов на Основании

Для документов, помимо операции копирования документов одного вида, которая выполняется аналогично копированию элементов справочников, определена операция ввода документов одного вида на основании документов другого вида. Особенность этого метода в различном составе реквизитов документов разных видов. Поэтому сопоставление реквизитов документов при вводе на основании делается программно в предопределенной процедуре модуля формы документа *ВводНаОсновании(<Документ-основание>).* Текст этой процедуры можно составить с помощью конструктора, работа с которым описана в 1 т. Конфигурирования и администрирования 1С:Предприятия 7.7, стр.161-163.

Чтобы конструктор работал правильно, желательно сначала отредактировать свойство документов «Ввод на основании». Это свойство определяет список видов документов, которые можно вводить на основании выбранного Документа.

Нажмем на кнопку «Ввод на основании» в окне свойств документа «Приход» или выберем соответствующий пункт в контекстном меню документа.

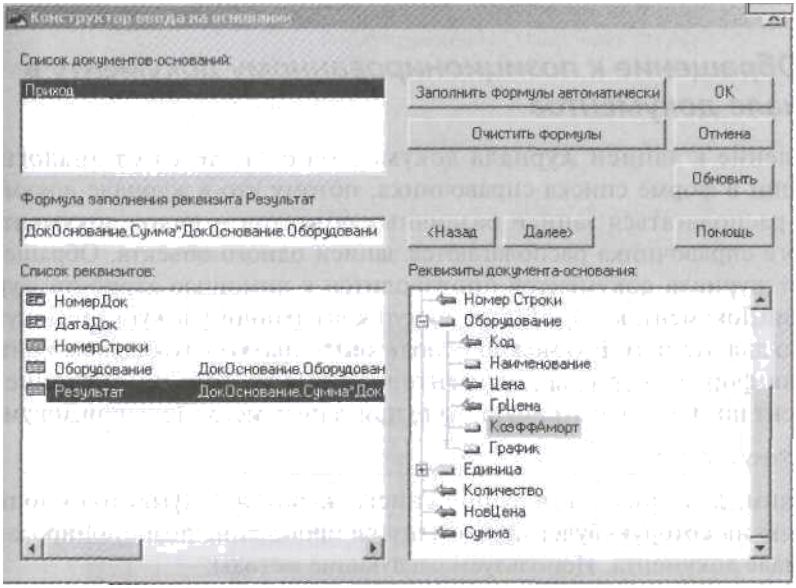




В списке документов «Является основанием для» выберем документ «Амортизация». Таким образом, мы определили, что документ «Амортизация» может быть введен на основании документа «Приход».

Обратите внимание на флажок в левом верхнем углу окна. Он устанавливается по умолчанию для любого нового документа, что позволяет вводить на основании этого документа любой существующий документ. Мы не будем вводить документ «Приход» на основании документа «Амортизация», поэтому отключите этот флажок в окне «Ввод на основании» документа «Амортизация».

Теперь войдите в модуль формы документа «Амортизация» и запустите конструктор ввода на основании.



Обратите внимание, что в списке реквизитов текущего документа нет общего реквизита «Основание» - конструктор не видит общих реквизитов, поэтому текст, введенный конструктором, мы отредактируем.

```
Процедура ВводНаОсновании(ДокОснование)

    Если ДокОснование.Вид() = "Приход" Тогда

        // !!!конструктор не видит общих реквизитов!!!

        Основание=ДокОснование;

        ДокОснование.ВыбратьСтроки();

        Пока ДокОснование.ПолучитьСтроку() = 1 Цикл

            НоваяСтрока();

            Оборудование = ДокОснование.Оборудование;

            Результат =
            ДокОснование.Сумма*ДокОснование.Оборудование.КоэффАморт*176/100;

        // 176 - это количество рабочих часов в месяце (22 р. дня)

        // Последнее выражение будет уточнено после конфигурирования
        Расчета.;

        КонецЦикла;

    КонецЕсли;
```

```
КонецПроцедуры
```

Упражнение 22. При вводе новых документов в информационную базу, установите префиксы для номеров документов. (Документы могут вводиться копированием и на основании документа другого вида).

8.9. Обращение к позиционированному документу в журнале документов

Обращение к записи журнала документов отличается от аналогичной операции в форме списка справочника, потому что в журнале документов могут располагаться записи различных объектов — видов документов. В каталоге справочника располагаются записи одного объекта. Обращение к записи журнала документов производится с помощью атрибута журнала ТекущийДокумент, который дает доступ к выбранному в журнале документу (только для чтения). Его можно использовать только в локальном контексте Модуля формы журнала документов. Чтобы возвратить значение типа документ по значению атрибута, надо применить метод ТекущийДокумент():

```
ТекущийДокумент.ТекущийДокумент()
```

Создадим, для примера, в форме списка журнала документов кнопку, по нажатию на которую будет производиться проведение позиционированного в журнале документа. Используем следующие методы:

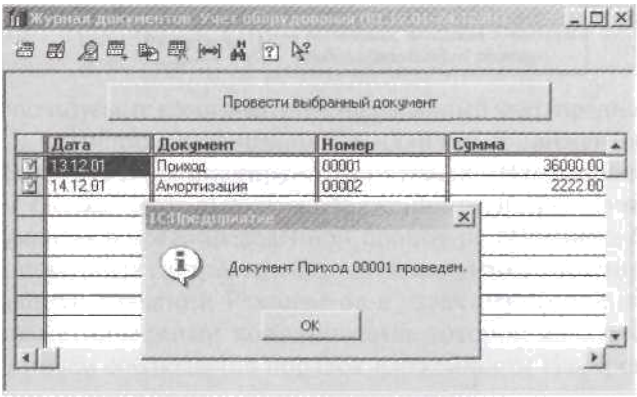
НайтиДокумент(<Документ>) позволяет найти документ по значению типа 'Документ'.
Возвращает: 1 - если действие выполнено (документ найден);
О - если действие не выполнено. <Документ> - выражение со значением типа 'Документ'. Метод можно использовать только для объектов, созданных функцией СоздатьОбъект. Провести(<Режим>,<Знач>) позволяет выполнить проведение документа. Возвращает: 1 - если проведение документа выполнено, 0 - иначе.
<Режим> - необязательный параметр. Число: 0 - проводить документ без сдвига ТА; 1 - проводить непроведенный документ реальным временем (со сдвигом ТА); 2 - перепроводить проведенный документ реальным временем (со сдвигом ТА); 3 - проводить любой (непроведенный, проведенный) документ реальным временем (со сдвигом ТА). Значение по умолчанию - 0. <Знач> - выражение произвольного типа, которое передается при запуске предопределенной процедуры ОбработкаПроведения (необязателен, по умолчанию - пусто).
Метод нельзя использовать в теле предопределенной процедуры ОбработкаПроведения. Если этот метод применяется в Модуле формы документа непосредственно к документу локального контекста, то данный метод отрабатывает те же действия, как интерактивное нажатие пользователем кнопки с формулой "#Провести". В этом случае, если параметр <Режим> опущен, то документ проводится в режиме, соответствующем установкам системы меню Сервис-Параметры.

Обратим внимание, что метод провести работает, как процедура — выполняет действие, и как функция возвращает значение. Воспользуемся этим свойством, чтобы показать пользователю, что документ действительно провёлся или нет. Это удобно особенно при перепроведении уже проведенных документов.

Алгоритм проведения приводится ниже.

```
Процедура ПровестиДок()

    Если ТекущийДокумент.Выбран()=1 Тогда

        Док=Создать0бъект("Документ");
```

```
            Док.НайтиДокумент(ТекущийДокумент.ТекущийДокумент());

            Если Док.Провести()=1 Тогда
                 Предупреждение("Документ "+ Док + "проведен.");

            Иначе

                 Предупреждение("Документ "+Док+ * НЕ ПРОВЕДЕН!!!!!!!!!");

            КонецЕсли;

       КонецЕсли;

КонецПроцедуры //ПровестиДок
```



Различия в использовании Общих реквизитов документов и Граф отбора

Общие реквизиты документов в основном предназначены для упрощения процесса создания нового документа и представляют собой набор реквизитов автоматически встраиваемых в новый документ.

Графы отбора это реквизиты документов, кроме Общих реквизитов - Общий реквизит не может быть Графой отбора, по которым ведётся сортировка и отбор документов. При включении реквизита документа в графу отбора по нему строится индекс в индексном файле, что существенно ускоряет сортировку документов и формирование запроса по этому реквизиту (см. 1 т. Конфигурирование и администрирование 1С:Предприятия 7.7, стр. 186-187).

Вопросы для самоконтроля

Какие отличия в структуре, в упорядочивании и в формах документов по сравнению со справочниками?

С помощью, каких объектов метаданных можно просмотреть существующие документы?

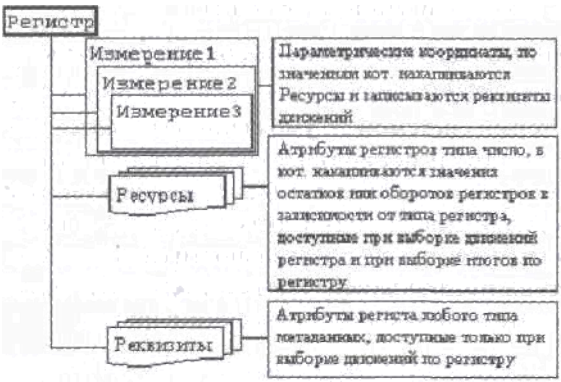
Чем отличается подчинение документов от справочников?

Какие реквизиты формы можно использовать в модуле документа?

Какие реквизиты и переменные можно использовать в шаблоне печатной формы?

Какой метод выводит на экран результирующую печатную форму?

IX. Основы построения «Оперативного учёта» в системе 1С:Предприятие



Регистры, используемые компонентой Оперативный учёт, предназначены для накопления и хранения информации о наличии и движении различных объектов учёта. Регистры оперируют понятиями «Измерение», «Ресурс», «Реквизит» и «Точка актуальности». Движения регистров записываются в «модуле документа» в предопределённой процедуре *ОбработкаПроведения()*. При записи движений регистров происходит накопление численных значений Ресурсов и записи значений Реквизитов в узлах некоторой многомерной таблицы, параметрическими координатами которой являются значения Измерений, причем соблюдается порядок вложенности Измерений.

В системе предусмотрено два типа регистров: оборотные регистры и регистры остатков. Для оборотных регистров характерен параметр «Периодичность», который задает период, для выбора итогов методом ИспользоватьПериод(). Для регистров остатков характерны типы движений «Приход» и «Расход», и параметр «Период актуальности», задающий период, в течение которого хранятся или восстанавливаются, для временного объекта, созданного методом СоздатьОбъект(), движения регистра. «Период актуальности» регистров остатков задается и редактируется, в отличие от оборотных регистров, в режиме «Предприятие», меню «Операции», пункт «Управление оперативными итогами...».

Оборотные регистры предназначены для хранения и оперативной выборки информации об общем объеме (обороте) учитываемого параметра за определенный период. Можно сказать, что оборотные регистры хранят информацию о том, как накапливались остатки.

Регистры остатков предназначены для хранения и оперативной выборки информации об остатках учитываемого параметра на определенный момент

времени.

На платформе V7.7 предусмотрена «Быстрая обработка движений», позволяющая ускорить операции чтения из регистров - запросы, временные расчеты итогов, обход движений регистров, однако установка этого признака замедляет запись движений данного регистра.

Точка актуальности, последовательность документов

Точка Актуальности (ТА) — это момент времени, определяемый позицией документа, на который системой по умолчанию выдаются значения остатков и итогов по регистрам, и периодические значения. Позиция документа формируется при его записи и представляет собой метку на оси времени. Если документ проводится после точки актуальности, и нет более поздних проведенных документов, то точка актуальности сдвигается на позицию этого документа. Система предполагает, что все документы оперативного учета вводятся последовательно, в хронологическом порядке.

Нарушение последовательности документов происходит из-за неправильнос вода хозяйственных операций. Если последовательность записи все; документов соответствовала фактической последовательности хозяйственных операций, то ввод Документа5, отражающего реальную хоз. операцию, которая была совершена, но не записана своевременно, не нарушит стройности учета Если же были и другие нарушения записи хоз. операций, например: некоторый приход был введен после того, как часть товаров из него была уже продана, то в промежутке между Документом 5 и ТА может быть обнаружен расходный документ, в котором недостаточно товаров для списания.

Для корректировки последовательности надо перепровести документы за период от введенного с нарушением последовательности документа до ТА:

Меню "Операции" - "Проведение документов".

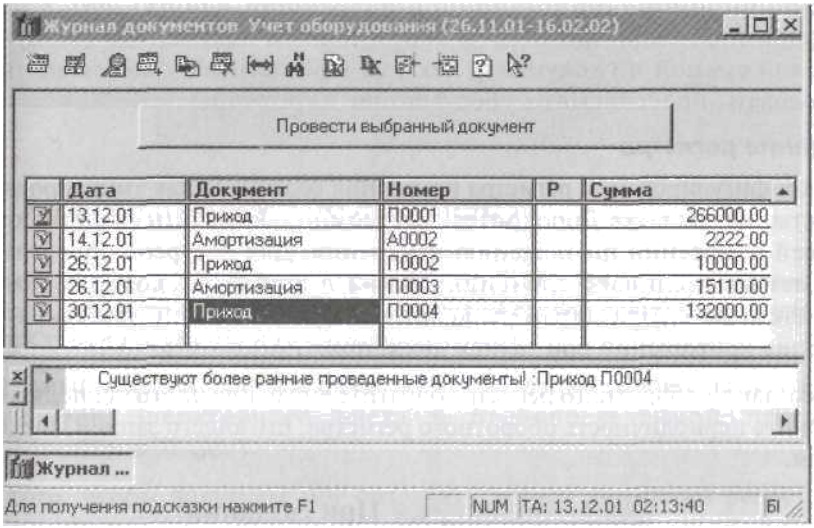
Точка актуальности, период итогов

На приведенном ниже рисунке показано, как формируются записи остатков или оборотов для регистров остатков или оборотных регистров соответственно. При переходе на следующий период фиксируется запись конечного остатка по периоду для регистра остатков или оборот за период

оборотного регистра. Имя таблицы (файла) остатков/оборотов имеет префикс RG***.dbf.

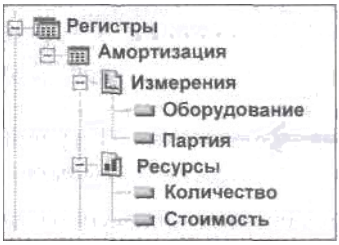
В журнале документов ТА отображается красной галочкой с подчеркиванием.

После ТА записи в регистрах недоступны ни для чтения, ни для записи. При попытке рассчитать регистры или провести документ после проведенных документов, расположенных после ТА (они помечены сиреневой галочкой в журнале документов), система выдаст сообщение: «Существуют более ранние проведенные документы::...».



Структура записей регистра

Записи накапливаемых значений ресурсов группируются по значениям измерений, причем существенен порядок измерений регистров, так как группировка значений ресурсов по второму измерению является вложенной в группировку по первому измерению. Другими словами, множество записей второго измерения является подмножеством множества записей первого измерения.

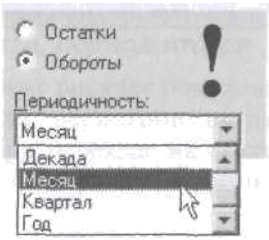


В нашем примере в регистре «Амортизация» по каждому элементу справочника «Оборудование» накапливаются записи по приходным документам, а не наоборот, как в документе. В результате, мы можем получить из регистра, например, количество и стоимость элемента оборудования, или список партий для выбранного элемента справочника «Оборудование» с ненулевыми значениями количества или стоимости, или текущую стоимость оборудования, оприходованного определенным документом (не путать с итоговой суммой по документу, так как в регистр мы сможем не только приходовать, но и списывать оборудование из регистра).

Создание регистра

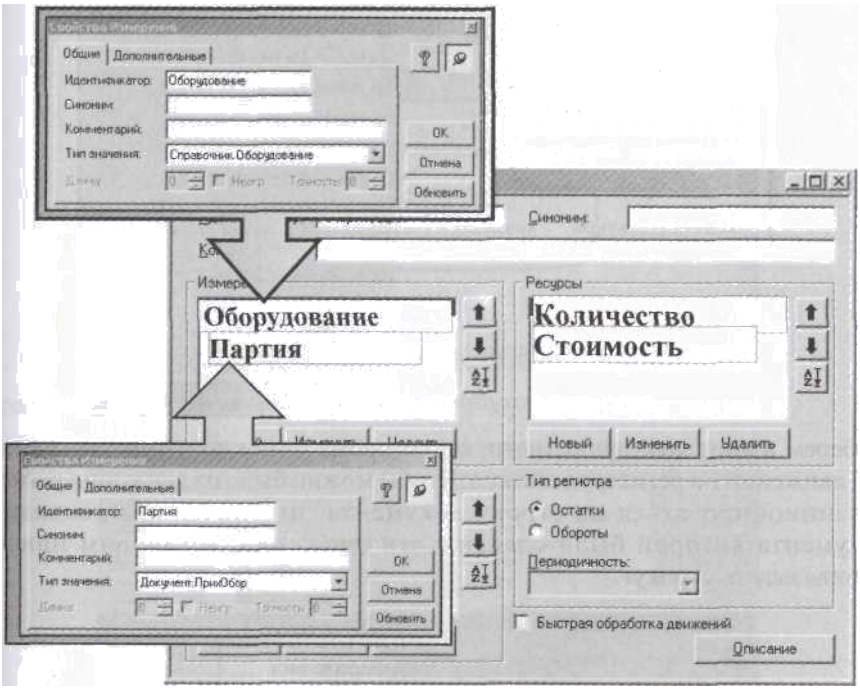
При конфигурировании регистра измерения создаются как типизированные объекты, на закладке Дополнительные можно обозначить свойство отбора записей измерений по значению измерения. Для ресурсов определяются параметры числового значения: длина и точность, которые должны соответствовать параметрам реквизитов документов, записывающих в них значения приращений при записи движений.

При создании оборотного регистра обратите внимание на его периодичность. Изменить периодичность оборотного регистра, имеющего записи движений, нельзя.



При создании оборотного регистра важно правильно задать его периодичность.

Упражнение 23. Создадим Регистр остатков «Амортизация» с измерениями «Оборудование» типа Справочник.Оборудование и «Партия» типа документ.Приход, ресурсами «Количество» и «Стоимость» и движение Приход по каждой строке документа «Приход».

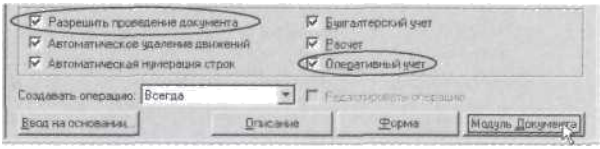


9.1. Запись движений по документу.

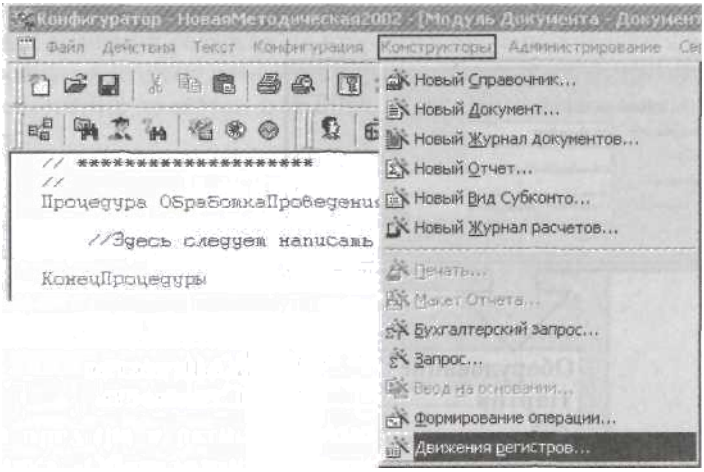
Упражнение 24. В документе «Приход» сформирйте записи движений по приходу в регистр «Амортизация» по каждой строке документа, привязывая НомерСтроки документа к каждому движению.

Программирование записей движений регистров производится в модуле Документов оперативного учета в предопределенной процедуре ОбработкаПроведения().

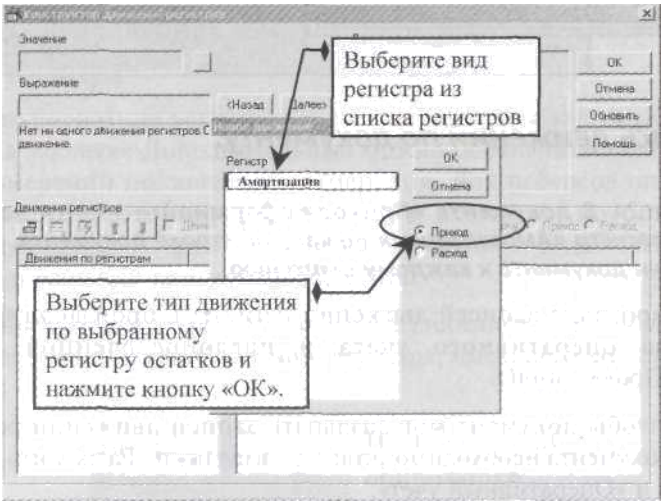
Для того чтобы документ мог создавать записи движений регистров, в свойствах документа необходимо установить флажки «Разрешить проведение Документа» и «Оперативный учет».

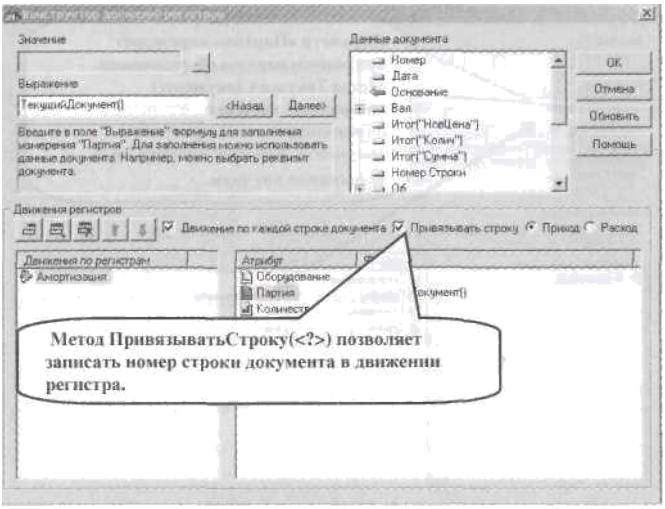

Разрешить проведение документа
Автоматическое удаление движений
Автоматическая нумерация строк
Бухгалтерский учет
Расчет
Оперативный учет
Создавать операцию:
Всегда
Ввод на основании
Описание
Форма
Модуль документа

В модуле документа «Приход» алгоритм записи движений регистров можно создать с помощью конструктора движений регистров.

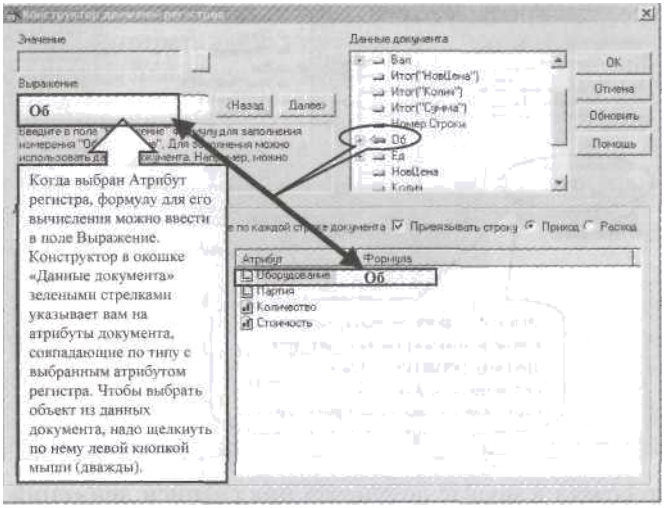


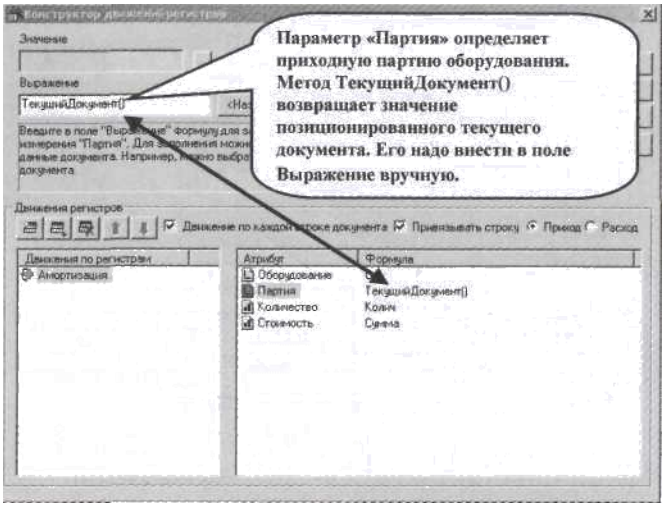
Выберем в конструкторе регистр, по которому будем создавать движения и тип движений по регистру остатков. Чтобы можно было из движения регистра спозиционироваться на строку документа, по значениям реквизитов документа которой были сделаны эти движения, установим признак «Привязывать строку».

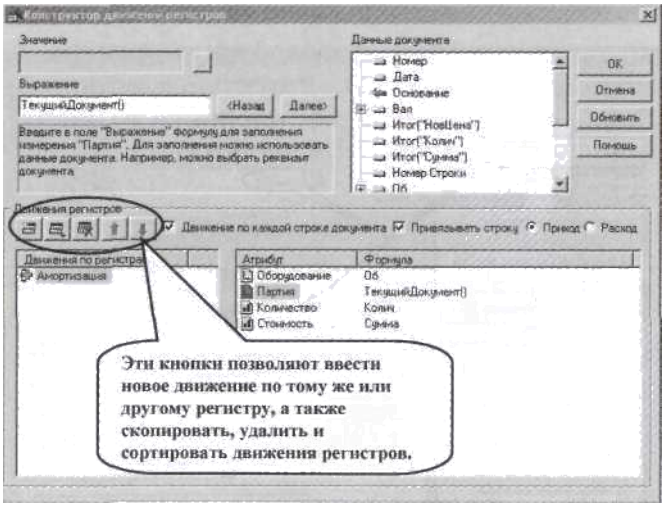




Определим выражения для значений атрибутов регистра.







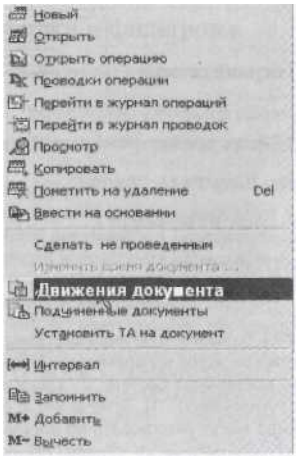
Конструктор создал в модуле документа цикл записи движений по строкам документа. В нашем случае это уже второй цикл по строкам, поэтому целесообразно отредактировать модуль документа после работы конструктора.

```
Процедура ОбработкаПроведения()

    ВыбратьСтроки();

    Пока ПолучитьСтроку() =1 Цикл

        //}}ДВИЖЕНИЯ_РЕГИСТРОВ

        Регистр.Амортизация.Оборудование - Оборудование;
```

```
Регистр.Амортизация.Партия = ТекущийДокумент();
```

```
            Регистр.Амортизация.Количество = Количество *
            Единица.Коэффициент;

            Регистр.Амортизация.Стоимость = Сумма;

            Регистр.Амортизация.ПривязыватьСтроку(НомерСтроки);

            Регистр.Амортизация.ДвижениеПриходВыполнить () ;

            //}}ДВИЖЕНИЯ_РЕГИСТРОВ

            // Запись истории периодического реквизита  справочника

            Если НовЦенаОборудование.Цена.Получить(ТекущийДокумент()) Тогда

                ПривязыватьСтроку(НомерСтроки);

                УстановитьРеквизитСправочника(Оборудование,"Цена",

                НовЦена);

            КонецЕсли;

            // Запись истории периодического реквизита  справочника

    КонецЦикла;

    КонецПроцедуры
```

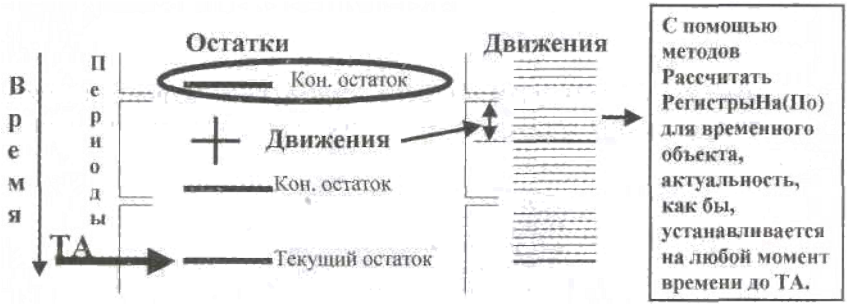
В Предприятии посмотреть движения регистров можно в журнале документов через контекстное меню документа или меню Действия — пункт Движения документа.



9.2. Временный расчет регистров

Рассмотрим, как можно получить из регистра остатков значения ресурсов на любой момент времени.

По структуре регистры остатков отличаются тем, что помимо таблицы остатков RG*.dbf (оборотов для оборотных регистров) формируют таблицу движений RA*.dbf , в которую записываются положительные или отрицательные значения приращений ресурсов и значения реквизитов регистра, причем, как мы уже отмечали, имеются два типа движений: "приход" и "расход". Записи движений регистров актуальны в течение определённого промежутка времени, называемого периодом актуальности итогов; возможные значения - месяц, пятнадцать дней, декада, пять дней. По умолчанию остатки, накопленные в ресурсах регистров, выдаются на точку актуальности - ТА. Если нужно получить остатки или итоги по регистру до ТА, то выполняется временный расчет регистров. Период актуальности итогов необходим для того, чтобы при каждом обращении к регистрам на момент до точки актуальности не вычислять значения остатков или итогов от начала работы предприятия по движениям. Для расчета остатков или итогов на любой момент времени берется остаток на начало периода и по движениям вычисляется остаток на данный момент времени.

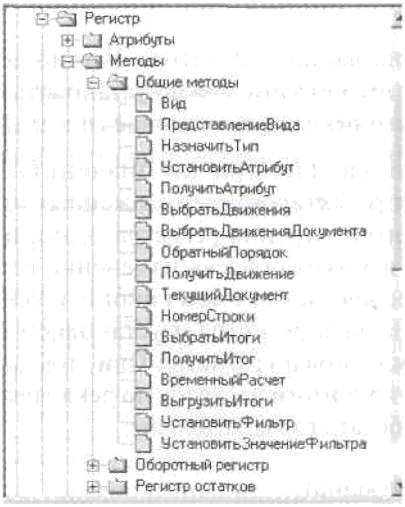



```
РегП=Создать0бъект("Регистр.Амортизация");

Если НашаДата<ПолучитьДатуТА() Тогда

     РегП.ВременныйРасчет();

     РассчитатьРегистрыНа(НашаДата);

КонецЕсли;
```

Метод **ВременныйРасчет (<Флаг>)** позволяет установить флаг участия регистра во временном расчете Возвращает текущее значение флага участия регистра во временном расчете. Значения параметра <флаг>: 1 - установить флаг участия регистра во временном расчете; 0 - сбросить флаг участия регистра во временном расчете (необязателен, по умолчанию - 1). Замечание: В один момент времени только по одному объекту регистров каждого вида могут участвовать во временном расчете.

Метод **РассчитатьРегистрыНа** (<ГраницаРасчета>,<ГрафаОтбора>) позволяет рассчитать все регистры **с установленным флагом временного расчета** на начало события, заданное границей расчета.

9.3. Методы регистров



Обратим внимание на методы регистров, позволяющие организовывать циклы выборки.

- Выборка движений регистров;
- Установка порядка выборки и фильтров:

```
Регистр.Амортизация.ОбратныйПорядок();

Регистр.Амортизация.УстановитьФильтр(ЗначениеОборудования);
```

Метод **ОбратныйПорядок**(<Режим>)позволяет установить порядок выборки движений документов. Возвращает:
1 - обратный порядок выборки документов,
О - выборка документов в порядке возрастания даты и времени.
<Режим> - число: 1 - выбирать движения документов в порядке убывания даты и времени; 0 -выбирать движения документов в порядке возрастания даты и времени, (по умолчанию - 1). Данный метод нельзя применять с методом ВыбратьДвиженияСОстатками. Метод можно использовать как процедуру и как функцию. Как функция метод возвращает значение, соответствующее порядку выборки до вызова.
Метод **УстановитьЗначениеФильтра**(<«Идентиф»>,<Значен>,<Вариант>) позволяет установить Фильтр по значению одного атрибута регистра, установить вариант отбора и принадлежность некоторому множеству значений <Идентиф> - идентификатор измерения или реквизита в кавычках. <Значен> - значение или список значений. <Вариант> - необязательный параметр. Число:
0- не фильтровать;
1- фильтровать по значению; . Значение по умолчанию
2- искать вхождение. Для варианта "2":
- если параметр <Значен> - это группа справочника, то осуществляется проверка вхождения в группу;
- если параметр <Значен> - это простой элемент справочника или другой тип значения, то осуществляется просто фильтрация по значению;
- для списка значений осуществляется проверка вхождения в список; - если пустое значение или пустой список значений, то условие не проверяется.

Метод **УстановитьФильтр**(<Измерение1>,<Измерение2>...,<Рекв1>,<Рекв2>) позволяет установить Фильтры по всем измерениям и реквизитам регистра. <Измерен1>,<Измерен2>... - значения измерений

регистра. Измерения могут задаваться с пропусками (неуказанное значение - просто запятая), фиксируются только указанные измерения. <Рекв1>,<Рекв2>... - необязательные параметры. Выражения со значениями реквизитов регистра. Заданные значения реквизитов будут влиять только на отбор движений регистра.

Обратите внимание на вариант 2 метода УстановитьЗначениеФильтра, благодаря которому этим методом можно установить фильтр на множество значений измерения или реквизита в отличие от метода УстановитьФильтр.

Возможность установки фильтра по значению реквизита регистра существенно упрощает редактирование отлаженных «чужих» конфигураций, так как создание нового измерения регистра — это, по сути дела, создание нового разреза учета, и может повлечь изменение всей структуры учета на предприятии. Для решения частных задач иногда можно обойтись созданием дополнительного реквизита регистра и установкой фильтра по нему при выборке движений или итогов по регистру. При этом следует учитывать, что в записях регистра остатки и итоги ресурсов по реквизиту не накапливаются, и такой фильтр будет работать гораздо дольше.

- Цикл выборки движений

```
Регистр. Амортизация . ВыбратьДвижения (ДатаНач, ДатаКон);

Пока Регистр.Амортизация.ПолучитьДвижение() =1 Цикл

      ЗначениеПартии= Регистр.Амортизация.Партия;

      // и прочие атрибуты регистра

КонецЦикла;
```

Метод **ВыбратьДвижения**(<ДатаНачала>,<ДатаКонца>,<ГрафаОтбора>) позволяет выбрать все движения регистра по датам в заданном интервале дат. <ДатаНачала> - дата, документ или позиция начала временного интервала выбора движений регистра; <ДатаКонца> - дата, документ или позиция конца временного интервала выбора движений регистра (если не указана или 0, то конец - ТА). <ГрафаОтбора> - необязательный параметр. Строковое выражение. Идентификатор графы отбора -установка использования определенной графы отбора. "*" - автоматический выбор графы отбора. Пустая строка - не использовать графу отбора. Если не указан - автоматический выбор графы отбора.

Движения можно выбрать также методами ВыбратьДвиженияДокумента(<Документ>) и ВыбратьДвиженияСОстатками(<ДатаКонца>, <ГрафаОтбора>).

Метод **ПолучитьДвижение()** позволяет выбрать очередное движение регистра. Возвращает: 1 - если следующее движение регистра выбрано, 0 - иначе. Метод можно использовать только для объектов, созданных функцией СоздатьОбъект.

- **Выборка итогов** — ненулевых остатков ресурсов регистров.

```
Рег=СоздатьОбъект("Регистр.Амортизация");

Рег.ВыбратьИтоги();

Пока Рег.ПолучитьИтог()=1 Цикл

            ЗначениеПартии= Рег.Партия;

            ЗначениеСтоимости= Рег.Стоимость;
```

```
//.....
КонецЦикла;
```

Перед вызовом метода ВыбратьИтоги() можно также установить фильтры и порядок выборки. **Метод ВыбратьИтоги() можно использовать только для переменной определенной с помощью метода СоздатьОбъект.**

На платформе V77 можно выгрузить итоги в таблицу значений.

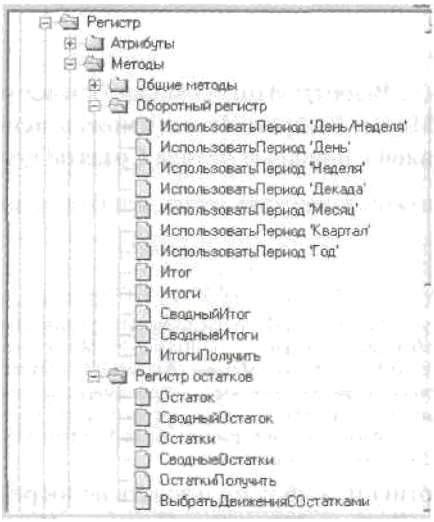
```
ТабЗн=Создать0бъект("ТаблицаЗначений");

Регистр.Амортизация.ВыгрузитьИтоги(ТабЗн) ;
```

Метод **ВыгрузитьИтоги**(<ТаблЗнач>,<ВклФильтр>,<Очищать>) позволяет выгрузить все итоги регистра с текущим фильтром в таблицу значений. <ТаблЗнач> - объект типа "ТаблицаЗначений", куда система выгрузит все итоги регистра, колонки таблицы создаются в соответствии с типами и идентификаторами атрибутов регистра. <ВклФильтр> - необязательный параметр. Число: 1 - в получаемую таблицу включаются измерения, закрепленные фильтром ; 0 - не включаются. Значение по умолчанию - 0. <Очищать> - необязательный параметр.Число:1 - перед выгрузкой таблица значений очищается;0 - не очищается.Значение по умолчанию -1.

Метод ВыгрузитьИтоги() можно использовать непосредственно для регистра в отличие от методов ВыбратьИтоги() и ПолучитьИтог(). Таблицу значений можно дописывать (см. параметр Очищать) - это позволяет выгрузить, например, в таблицу итоги из нескольких регистров, а отсортировать её или «свернуть».

Метод **Свернуть**(<ГруппКолонки>,<СуммКолонки>) позволяет свернуть таблицу значений по соответствующим значениям колонок, т.е. заменяет на одну строку все дублирующие (по значениям группировочных колонок) строки, суммируя значения по суммируемым колонкам. <ГруппКолонки> -группировочные колонки (номера или идентификаторы колонок через запятую), по которым группировать данные. <СуммКолонки> - суммируемые колонки (номера или идентификаторы колонок через запятую), по которым суммировать данные.

9.3.1. Методы регистров остатков



Метод **ОстаткиПолучить**() позволяет получить все ресурсы по регистру. Измерения записываются в атрибуты, полученные ресурсы считываются из атрибутов регистра. Метод используется только для регистров остатков.

```
Регистр.Амортизация.Оборудование= Оборудование;

Регистр.Амортизация.ОстаткиПолучить();

ЗначениеСтоимости= Регистр.Амортизация.Стоимость;
```

Методы **Остатки**(<?>) и **СводныеОстатки**(<?>) также позволяют получить остатки по всем ресурсам регистра. Параметры: [<Значение измерения1>, <Значение измерения2>,...] . При вызове метода Остатки (<?>) необходимо указать значения всех измерений регистра. Для метода СводныеОстатки(<?>) можно указать значения только необходимых измерений, а остальные опустить, поставив соответствующие запятые.

```
Регистр. Амортизация.СводныеОстатки(ЗначениеОборудования,);

ЗначениеСтоимости= Регистр.Амортизация.Стоимость;

ЗначениеКолл_во= Регистр.Амортизация.Количество;
```

Методы **Остаток**(<?>, «») и **СводныйОстаток**(<?>, «») возвращают значение остатка заданного ресурса регистра. Параметры:[<Значение измерения1>, <Значение измерения2>,... , <Имя ресурса в кавычках>]. При вызове метода Остаток(<?>,«») необходимо указать значения всех измерений регистра. Метод СводныйОстаток(<?>, «») возвращает значение сводного остатка заданного ресурса регистра по неполному списку значений измерений, как и метод СводныеОстатки(<?>).

```
ЗначениеСтоимости=
Регистр.Амортизация.СводныйОстаток(ЗначениеОборудования , ,"Стоимость ");

ЗначениеКолл_во=
Регистр. Амортизация.СводныйОстаток(ЗначениеОборудования , ,
"Количество");
```

9.3.2. Методы оборотных регистров

Методы получения значений ресурсов оборотных регистров имеют списки параметров и назначения аналогичные соответствующим методам регистров остатков. Перед получением итогов необходимо указать за какой период надо получить значения ресурсов с помощью метода ИспользоватьПериод(<?>). Список параметров этого метода зависит от Периодичности оборотного регистра, которая задается при конфигурировании регистра. Для регистра с периодичностью Месяц и аналогичными атрибутами справедлива запись:

```
Регистр.Обортный.ИспользоватьПериод(2001,ЗначениеМесяца);

ЗначениеОборота=Регистр.Обортный.СводныйИтог(Зн_еИзмерения,
"ИдентификаторРесурса");
```

9.4. Списание по партиям

Партионный учет подразумевает ведение учета с аналитикой по партиям. Как правило, партии формируются приходными документами по каждой позиции учитываемого объекта. Списание учитываемого объекта при этом производится также в разрезе сформированных партий.

№ партии	Текущий Остаток Партии	Новый остаток партии	Всего списать		50,00
			Списали = Если(ТОП <= Списать; ТОП; Списать)	Списали	Осталось списать
Партия 1	10,00	0,00		10,00	40,00
Партия 2	12,00	0,00		12,00	28,00
Партия 3	14,00	0,00		14,00	14,00
Партия 4	16,00	2,00		14,00	0,00
Партия 5	18,00	18,00		0,00	0,00

В зависимости от порядка списания партий выбирается очередная партия для списания и списывается или требуемое количество, уменьшая остаток партии, либо весь текущий остаток партии, если требуемое количество больше остатка партии.

При учете по партиям списание производится не из общего количества или стоимости объекта списания, а по сформированным партиям, что позволяет более точно вычислять себестоимость продукции. Партии могут формироваться

- приходными документами, когда каждая строка документа формирует новую партию,
- в зависимости от свойств объекта списания: например по сроку реализации.

Порядок списания также может быть различный:

- по выбранной (в диалоге) партии,
- по времени формирования партии — **FIFO** (первый пришел - первый ушел), LIFO (последний пришел - первый ушел),
- по размеру партии — по максимальному или минимальному остатку партии и прочее.

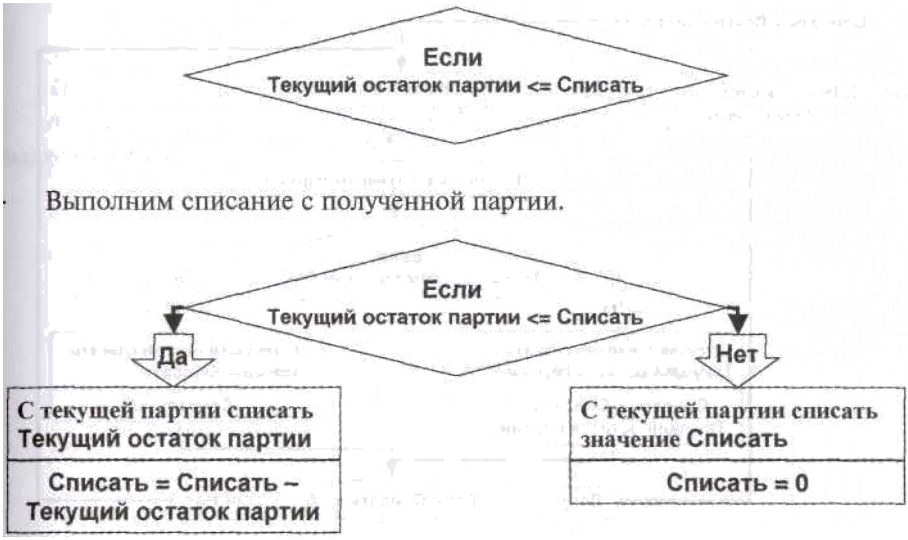
В нашем примере мы рассмотрим списание по методам **FIFO** и **LIFO.**

При списании по партиям, в отличие от списания по средней себестоимости, необходимо сначала сформировать список партий, подлежащих списанию, для выбранного объекта списания, а затем организовать списание по партиям.

Списание партии производится следующим образом:

Создадим переменную, в которой будет храниться списываемое с партии значение. Первоначально оно должно быть равно полному значению списания.

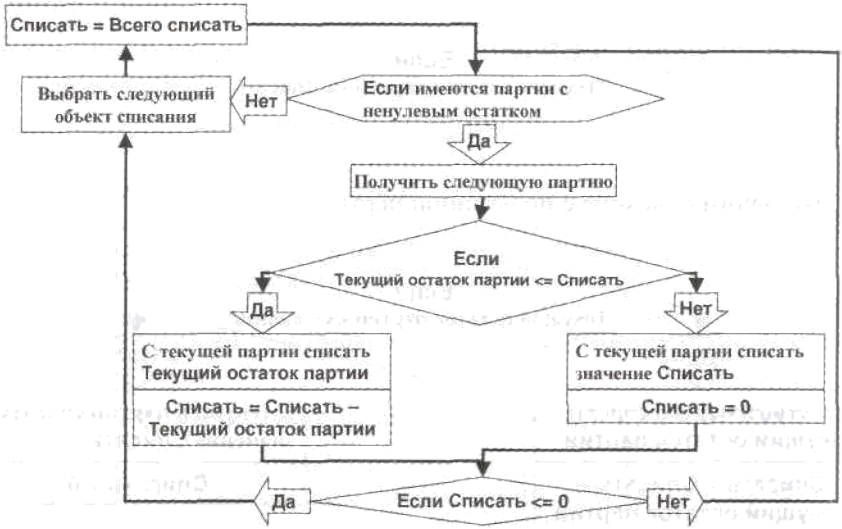
Проверим остаток по партии.



В этом случае партию списали целиком, но имеются другие несписанные партии, с которых можно списать остаток списания. В этом случае остаток списания полностью списан с текущей партии. Остаток по партии уменьшился на величину списания.

Перед завершением цикла, обязательно проверим - всё ли мы списали?

Так строится любой алгоритм списания по партии.



Упражнение 25. Предусмотрите в документе «Амортизация» выбор типа списания «По средней себестоимости», FIFO или LIFO типа перечисление. Предусмотрите ввод значения типа списания по умолчанию при создании нового документа.

Не забудьте создать и реквизит объекта (формата хранения), и соответствующий элемент формы.

Сформируем записи движений списания оборудования по партиям, в модуле документа «Амортизация», в регистр «Амортизация».

Установим флажок «Оперативный учет» в окне редактирования документа «Амортизация».

Войдем в модуль документа «Амортизация».

Чтобы документ можно было вводить задним числом с нарушением последовательности, создадим временный объект типа Регистр .Амортизация и рассчитаем его на текущий документ.

Партии запишем в таблицу значений.

Метод **ИтогиАктуальны**() возвращает флаг актуальности итогов: 1 - итоги актуальны, 0 - иначе. Метод доступен только в Модуле документа в предопределенной процедуре ОбработкаПроведения.

```
Рег=Создать0бъект("Регистр.Амортизация");
ТаблицаПартий=Создать0бъект("ТаблицаЗначений");

Если ИтогиАктуальны()=0 Тогда
```

```
        Рег.ВременныйРасчет() ;
        РассчитатьРегистрыНа(ТекущийДокумент());

КонецЕсли;
```

Организуем цикл выборки строк документа, определим списываемое значение и проверим остатки оборудования по стоимости.

РазделительСтрок - символ перевода строки текста.

```
ВыбратьСтроки();

Пока ПолучитьСтроку() =1 Цикл

      ВсегоСписать=Результат; // определим списываемое значение

      Дефицит=Рег.СводныйОстаток(Оборудование,,"Стоимость")-

      ВсегоСписать;

      Если Дефицит<0 Тогда

         Предупреждение("Дефицит по оборудованию "+Оборудование+" = "+

         Дефицит +РазделительСтрок+" Документ нельзя проводить!!");

         // Отмена проведения документа

         СтатусВозврата(0);

         Возврат;

      КонецЕсли;

КонецЦикла;
```

Сформируем партии оборудования. С помощью метода ВыбратьСтроку удобно просмотреть созданную таблицу значений.

```
Рег.УстановитьФильтр(Оборудование,);
Рег.ВыгрузитьИтоги(ТаблицаПартий,1,1);//Создали партии
оборудования    //ТаблицаПартий.ВыбратьСтроку(); //отладка
```

Партии отсортируем в зависимости от выбранного в документе типа списания.

Метод таблицы значений **Сортировать**(<Колонки>,<ДокумПоДате>) позволяет сортировать таблицу значений по колонкам.
<Колонки> - строковое выражение, которое определяет колонки, порядок и направление сортировки. Формат передаваемой строки - это разделенные запятыми номера или идентификаторы колонок со знаком направления сортировки ("+" - сортировать по возрастанию; "- " - сортировать по Убыванию; "*" - сортировать по внутреннему значению). Знак направления сортировки можно Указывать до или после обозначения колонки через пробел или без пробела. По умолчанию направление сортировки принимается по возрастанию.
<ДокумПоДате> - необязательный параметр. Имеет смысл только в том случае, если значениями таблицы значений являются документы. В этом случае можно задавать сортировку документов по их хронологии. Число: 1 - сортировка по хронологии Документов; 0 - нет. Значение по умолчанию - 0.

```
Если ТипСписания=Перечисление.ТипыСписания.FIFO Тогда
```

```
    ТаблицаПартий.Сортировать("Партия",1);
ИначеЕсли ТипСписания=Перечисление.ТипыСписания.LIFO Тогда
    ТаблицаПартий.Сортировать("Партия -" ) ;
Иначе // по средней себестоимости
    //Алгоритм списания по средней себестоимости
КонецЕсли;
```

Организуем цикл выборки партий оборудования из таблицы значений по каждой строке документа.

```
// Списание по партиям
ТаблицаПартий.ВыбратьСтроки();
Пока ТаблицаПартий.ПолучитьСтроку()=1 Цикл
     //Алгоритм списания по партиям
КонецЦикла;
```

В цикле выборки партий сформируем условие списание партии.

```
Если ТаблицаПартий.Стоимость<=ВсегоСписать Тогда
    //Партию списываем целиком и уменьшим списываемое значение
Иначе
    //С партии списываем списываемое значение целиком и обнуляем его
КонецЕсли;
//Поверяем: всё ли списали?
Если ВсегоСписать <=0 Тогда
    Прервать;
КонецЕсли;
```

Для определения списываемого количества разделим списываемую стоимость на себестоимость оборудования.

При списании по средней себестоимости:

```
Результат*Рег.СводныйОстаток(Оборудование,,"Количество")/
Рег.СводныйОстаток(Оборудование,,"Стоимость")
```

При списании по партии: если партию списываем целиком, то количество списываем из парии, иначе пересчитываем его через себестоимость

партии:

```
ВсегоСписать* ТаблицаПартий.Количество/ТаблицаПартий.Стоимость
```

Запишем движения по регистру Амортизация.

```
Процедура ОбработкаПроведения()
Рег=Создать0бъект("Регистр.Амортизация");
ТаблицаПартий=Создать0бъект("ТаблицаЗначений");
Если ИтогиАктуальны()=0 Тогда
      Рег.ВременныйРасчет();
      РассчитатьРегистрыНа(ТекущийДокумент());
КонецЕсли;
ВыбратьСтроки();
Пока ПолучитьСтроку() =1 Цикл
      // Списание по Оперативному учету
      ВсегоСписать=Результат; // определим списываемое значение
      Дефицит=Рег.СводныйОстаток(Оборудование,,"Стоимость")-ВсегоСписать;
      Если Дефицит<0 Тогда
            Предупреждение("Дефицит по оборудованию "+Оборудование+" = "+
            Дефицит +РазделительСтрок+" Документ нельзя проводить!!!");
            // Отмена проведения документа СтатусВозврата(0) ;
            Возврат;
      КонецЕсли;
      Регистр.Амортизация.Оборудование = Оборудование;
      Регистр.Амортизация.ПривязыватьСтроку(НомерСтроки);
      Рег.УстановитьФильтр(Оборудование,);
Рег.ВыгрузитьИтоги(ТаблицаПартий,1,1);//Создали партии
оборудования
//ТаблицаПартий.ВыбратьСтроку(); //отладка
      Если ТипСписания=Перечисление.ТипыСписания.FIFO Тогда
            ТаблицаПартий.Сортировать("Партия", 1);
      ИначеЕсли ТипСписания=Перечисление.ТипыСписания.LIFO Тогда
            ТаблицаПартий.Сортировать("Партия -") ;
      Иначе // по средней себестоимости
            //Алгоритм списания по средней себестоимости
            Регистр.Амортизация.Количество =
```

```
            Результат*Рег.СводныйОстаток(Оборудование,,"Количество")/
            Рег.СводныйОстаток(Оборудование,,"Стоимость");

            Регистр.Амортизация.Стоимость= Результат;

            Регистр.Амортизация.ДвижениеРасходВыполнить();

            Продолжить;

        КонецЕсли;

        // Списание по партиям ТаблицаПартий.ВыбратьСтроки();

        Пока ТаблицаПартий.ПолучитьСтроку()=1 Цикл

            Регистр.Амортизация.Партия = ТаблицаПартий.Партия;

            Если ТаблицаПартий.Стоимость<=ВсегоСписать Тогда

                //Партию списываем целиком и уменьшим списываемое значение

                Регистр.Амортизация.Количество = ТаблицаПартий.Количество;
                Регистр.Амортизация.Стоимость= ТаблицаПартий.Стоимость;
                ВсегоСписать=ВсегоСписать-ТаблицаПартий.Стоимость;

            Иначе

                //С партии списываем списываемое значение целиком и
                обнуляем его
                Регистр.Амортизация.Количество - ВсегоСписать*
                ТаблицаПартий.Количество/ТаблицаПартий.Стоимость;

                Регистр.Амортизация.Стоимость = ВсегоСписать;

                ВсегоСписать=0;

            КонецЕсли;

            Регистр.Амортизация.ДвижениеРасходВыполнить();

            //Проверим: всё ли списали?

            Если ВсегоСписать < = 0 Тогда

                Прервать;

            КонецЕсли;

        КонецЦикла;

    КонецЦикла;

    КонецПроцедуры
```

9.5. Особенности временного расчета регистров оперативного учета

Если мы откроем форму объекта, например: отчета, который создает временный расчет по регистру «Амортизация», и, не закрывая эту форму, откроем документ «Амортизация», который тоже создает временный расчет по регистру «Амортизация», то, при неправильном конфигурировании,

программа может сообщить об ошибке. Такая ситуация возникает, если переменная типа регистр создана в глобальном модуле или объявлена в модуле формы и после выполнения временного расчета продолжает существовать. Чтобы избежать таких ситуаций следует либо использовать переменные создаваемые непосредственно в момент выполнения процедуры, либо отключать регистр от временного расчета, после, того как он был использован.

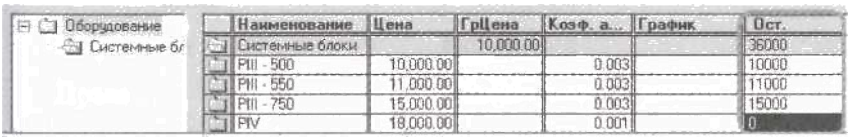
Еще одну особенность временных расчетов следует учитывать при проведении документов. Если в алгоритме проведения документа анализируются остатки и выполняются движения регистра, то при использовании текущих итогов регистра после записи каждого движения итоги изменяются, чтобы поддерживать временный расчет регистров в актуальном состоянии необходимо использовать метод *Актуальность()* со значением параметра равным 1 - *Актуальность(1).*

Упражнение 26. В Форме списка справочника «Оборудование» вывести текущие остатки по стоимости конкретного оборудования (элементу справочника) из регистра «Амортизация».

В ФормеСписка справочника «Оборудование», в табличной его части создадим реквизит типа текст с заголовком «Ост.», а в поле формула на закладке «Дополнительные» выше описанного реквизита введем соответствующее выражение. Напоминаю, что получить элемент по строке формы списка можно с помощью метода ТекущийЭлемент().

Упражнение 27. (Необязательное) В выше описанном поле показать остатки по элементам и суммарные остатки по группам справочника.

Метод **ЭтоГруппаО** для справочников возвращает флаг группы: 1 - если элемент является группой, 0 -если это обычный элемент.



Наименование	Цена	ГрЦена	Коэф. а...	График	Ост.
Системные блоки		10,000.00			36000
PIII - 500	10,000.00		0.003		10000
PIII - 550	11,000.00		0.003		11000
PIII - 750	15,000.00		0.003		15000
PIV	18,000.00		0.001		0

Вопросы для самоконтроля

На какой момент времени возвращаются остатки по регистрам, если не задан временный расчет регистров?

Когда необходимо для получения остатков выполнить временный расчет регистров?

В каких текстовых модулях определен метод ИтогиАктуальны(). В

чем отличия регистров остатков от оборотных регистров?

Что произойдет с базой данных, если изменить тип реквизита документа, участвующего в формировании аналитики по регистру?

Что произойдет с базой данных, если изменить тип измерения регистра?

Назовите преимущества и недостатки партионного учета.

X. Основы построения «Бухгалтерского учёта» в системе 1С:Предприятие

10.1. Принцип двойной записи. Баланс

Двойная запись — прием учета, когда сумма и др. атрибуты каждой хозяйственной операции фиксируются дважды:

по дебету одного счета записывается изменение актива или обязательства,

по кредиту другого (корреспондирующего) счета указывается источник, за счет которого произошло это изменение.

Дебет - он должен

Кредит - он верит

Впервые описана Лукой Пачоли в 1494 г. в книге «Сумма арифметики, геометрии, учения о пропорциях и отношениях», в разделе «Трактат о счетах и записях» , посвященной применению математики в коммерции.

Баланс

Слово «баланс» происходит от латинского выражения bi lanx — «две чашки» и означает равновесие.

Баланс — система показателей, отражающая поступление и расходование средств путем их сопоставления на определенную дату.

Бухгалтерский баланс состоит из двух частей: пассива и актива.

Актив	**Пассив**
Имущества	Источники
Права	Обязательства к уплате
Обязательства к получению	

Таким образом, общий принцип двойной записи заключается в том, что каждая операция одновременно фиксируется как изменение активов (приход) и пассивов (расход). Если после суммирования записей в каждом разделе суммы не совпадают, следует искать ошибку.

Равновесие бухгалтерского баланса - это равенство итогов актива и пассива, так как в активе баланса показываются хозяйственные средства предприятия, а в пассиве - источники этих средств, т.е. те же самые средства, но сгруппированные по другому признаку - по источникам их формирования.

10.2. Бухгалтерские счета

Все хозяйственные операции находят свое отражение на счетах бухгалтерского учета. Если зафиксировать показатели учета на определенную дату, они будут представлены в виде остатков (сальдо) по счетам. При рассмотрении показателей во времени, принято оперировать понятием отчетного периода, в котором можно выделить начальные остатки, обороты и конечные остатки по счетам.

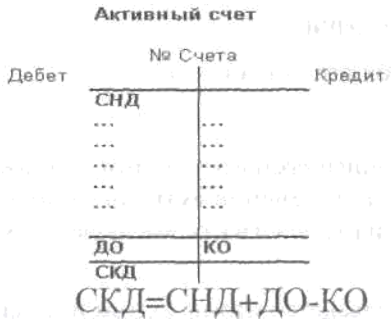
Другими словами бухгалтерский счет - это "регистр", на котором ведется учет некоторого разреза хозяйственной деятельности предприятия. Отличие от регистров оперативного учета в том, что балансовые счета — это взаимосвязанные разрезы учета. Взаимосвязь счетов обусловлена корреспонденциями — то есть двойными записями.

Сальдо и обороты по счетам бухгалтерского учета (самолетики)

Оборот - приход или расход объекта учета на счете за определенный период времени.

Сальдо — остаток объекта учета на счете на дату сальдирования (расчета остатка).

Балансовые счета можно разделить на активные, пассивные и активно-пассивные. Активные счета имеют дебетовое сальдо (остаток), пассивные кредитовое сальдо (остаток). Красным «неправильным» сальдо называют ненулевое конечное кредитовое сальдо на активном счете или конечное дебетовое сальдо на пассивном счете в конце балансового периода. Активно-пассивные счета позволяют в одном разрезе учета вести и активные, и пассивные статьи баланса.



СНД — сальдо начальное дебетовое, СКД — сальдо конечное дебетовое, ДО — дебетовый оборот, КО — кредитовый оборот, СНК — сальдо конечное кредитовое, СКК — сальдо конечное кредитовое.

10.3. Виды метаданных компоненты «Бухгалтерский учет»

План счетов - представляет собой таблицу, каждая строка которой отражает определенный счет или субсчет бухгалтерского учета.

Бухгалтерский счет — (далее «Счет») это "регистр", на котором ведется учет некоторого разреза хозяйственной деятельности предприятия. Счета бывают Активные, Пассивные и Активно-Пассивные.

Виды Субконто предназначены для определения объектов аналитического учёта по счетам, могут принимать значения справочников, документов, перечислений и базовых типов данных.

Проводка (корреспонденция) — это связь счетов б/у обусловленная одной хозяйственной операцией. Она предназначена для изменения состояния средств, отражаемых в бухгалтерском учёте. Содержит корреспондирующие счета, сумму, информацию по синтетическому и аналитическому учёту, а также дополнительные реквизиты.

Операция - совокупность проводок, отражающих в бухгалтерском учёте хозяйственную операцию.

В системе «1С: Предприятие» счета предназначены для хранения планов счетов бухгалтерского учета, то есть объектов синтетического учета средств предприятия. Если в конфигурации не определены планы счетов - нельзя создать бухгалтерский документ.


Поддерживается одновременно несколько планов счетов
Задается длина кода и наименования счета
Для счетов настраиваются дополнительные реквизиты
Для счетов настраиваются формы просмотра списка и редактирования счетов
Поддерживается многомерный и многоуровневый аналитический учет
Поддерживается многовалютный учет
Настраивается возможность использования *разделителя учета*
Учебный
Новый
Количественный учет
Только по аналитике
Аналитический учет
Валютный учет
Форма счета
Формы списка

Представленные в окне планов счетов свойства устанавливаются для любого плана счетов, создан ного в конфигурации.Например, нельзя создать два плана счетов с различным набором реквизитов счета или с различными разделителями учета.

В качестве разделителя учета можно выбрать любой реквизит журнала проводок.

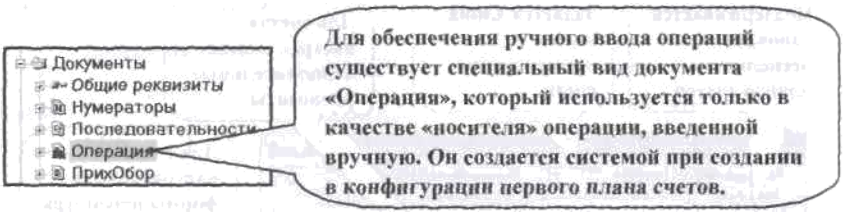
Интерактивные формы тоже являются общими для всех планов счетов. Обычно, для различных планов счетов, настраивают формы списка с различными наборами элементов формы и при открытии плана счетов жестко (программно) указывают, какую форму списка открывать для данного плана счетов.

Для ведения валютного учета (вычисления рублевого покрытия валютных сумм) необходимо предусмотреть в конфигурации Справочник с реквизитами Курс и Кратность, а в документах, формирующих бухгалтерские проводки, реквизит, ссылающийся на этот справочник.

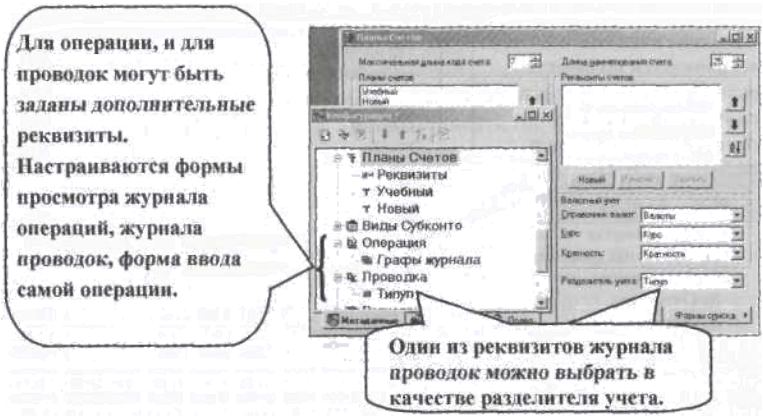
Чтобы можно было вести на счете аналитический учет в разрезе субконто, необходимо установить значение «Макс. Количество субконто» больше нуля.

Очень важный параметр «Количественный учет» «Только по аналитике». Если он установлен, Вы не получите ни сальдо, ни обороты по количеству в разрезе счетов. В этом случае количественный учет возможен только в разрезе субконто. Если Вашему бухгалтеру необходимо видеть количественные сальдо и обороты по субсчетам, необходимо снять этот флажок и перепровести документы бухгалтерского учета.

Операции и проводки

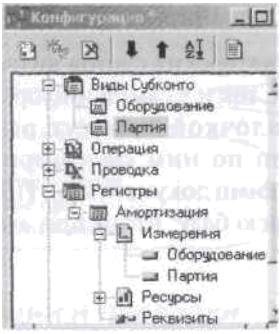


Создадим реквизит проводки «ТипУчета» типа перечисление ТипыУчета и разделитель учета в свойствах планов счетов.



В нашем примере мы создадим структуру бухгалтерского учета для списания оборудования по партиям, аналогичную структуре оперативного учета.

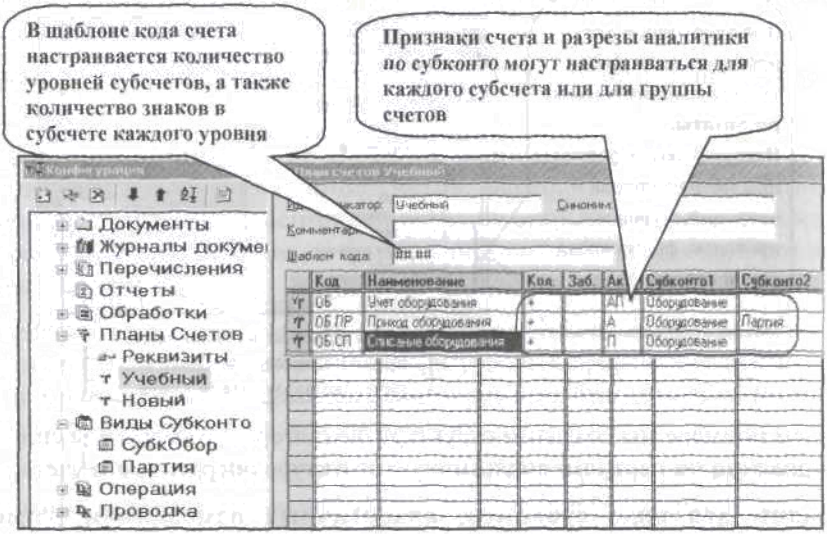
Создадим два вида субконто, аналогичные измерениям регистра «Амортизация». Эту операцию можно сделать методом drag & drop в дереве метаданных. Перетащим последовательно измерения регистра на ветку Виды Субконто.



В окне планов счетов создадим План счетов «Учебный» и определим максимальное количество субконто — 2. В учебном плане счетов создадим группу счетов с субсчетами. Иерархия в плане счетов определяется шаблоном кода.

Новый счет удобно вводить клавишей Ins или через меню «Действия» - «Новая строка». Учет не списанного оборудования будем вести на Активном счете, а списанного - на Пассивном. Количественный учет и Субконто1-

Оборудование определим по группе счетов, а Субконто2 — Партия только на активном счете.



При создании бухгалтерских запросов надо внимательно просмотреть, как заданы параметры счета в плане счетов. Если параметр счета задан только на субсчете, как Субконто2 — Партия, то итоговые значения сальдо и оборотов по группе счетов для этого параметра не накапливаются.

Важной особенностью бухгалтерских счетов является возможность создания счетов, как в конфигурации, так и в самой информационной базе. Последние не помечены красной галочкой и могут редактироваться в режиме «Предприятие», даже если по ним сформированы проводки ручными операциями или бухгалтерскими документами. После такого редактирования, необходимо перепровести всю базу; итоги при этом изменятся.

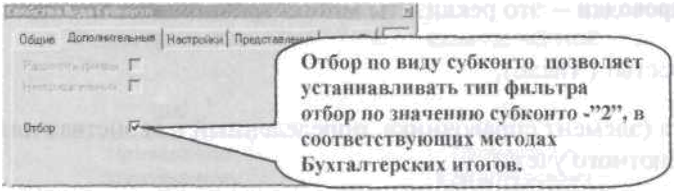
10.3.1. Виды субконто

Термин *субконто* используется для обозначения набора значений, используемых для ведения аналитического учета по счету. *Вид субконто* идентифицирует *совокупность объектов* конкретного типа, которые могут использоваться для ведения аналитического учета. Значение субконто в проводке - это ссылка на объект аналитики.

Созданные виды субконто указываются при настройке аналитического учета по счетам. Для каждого счета допускается использование до 5 видов субконто, что позволяет вести многомерный аналитический учет. Ведение многоуровневого аналитического учета реализуется путем использования

многоуровневых справочников.

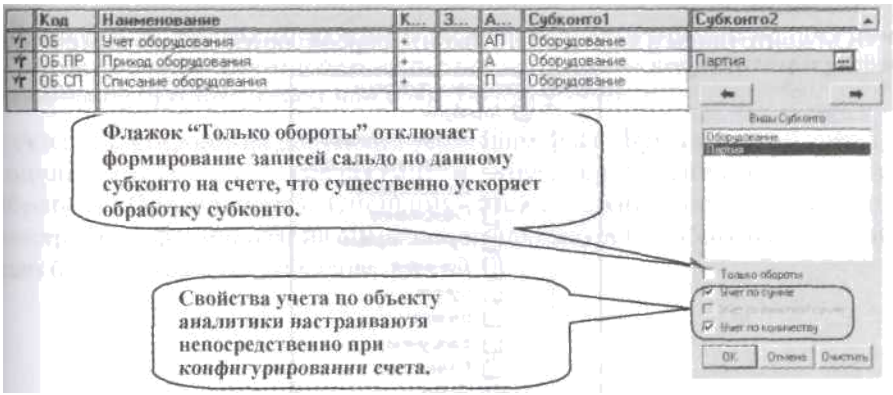
Свойства объекта аналитики: тип значения, свойство отбора, ссылки на его цены и представления объекта, настраиваются при конфигурировании вида субконто



В нашем примере установим признак отбора для субконто «Оборудование».

Свойства количественного, валютного и суммового учета по субконто определяются непосредственно на счетах.

Субконто на счетах имеют назначение аналогичное измерениям оборотного регистра. Отличие заключается в том, что в корреспонденции мы задаем одну и ту же ссылку на объект аналитики на разных счетах, что позволяет получить обороты между корреспондирующими счетами по объекту аналитики.



10.3.2. Атрибуты операции и проводки

Бухгалтерская операция имеет аналогию с документом, это просто другое представление документа.

Атрибуты операции — это реквизиты шапки, они, как правило, и берутся из Шапки бухгалтерского документа:

ДатаОперации (Дата, обычно это дата документа),

Содержание (обычно произвольная строка),

СуммаОперации (Число, обычно это итог по сумме — Итог(«Сумма»)),

Документ (ссылка на документ, создавший операцию).

Атрибуты проводки — это реквизиты многострочной части.

Количество (Число),

Валюта (элемент справочника, определенный в свойствах планов счетов для валютного учета),

ВалСумма (Число),

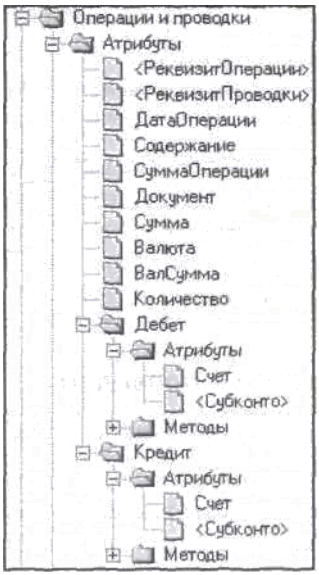
Сумма (Число, для валютных счетов — это рублевое покрытие).

Параметры синтетического и аналитического учета задаются в проводке по дебету и кредиту:

Атрибуты Дебета и Кредита:

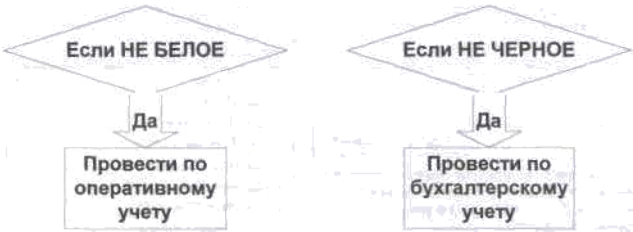
- Счет,

  Субконто (виды и количество субконто определяются выбранным счетом).



Упражнение 28 В документе Приход создайте реквизит шапки ТипУчета типа перечисление ТипыУчета. Предусмотрите ввод его значения по умолчанию -текущее значение константы ТекТипУчета, при создании нового документа в информационной базе.

В зависимости от значения этого реквизита мы будем выполнять движения по документу. У этого перечисления три значения: Красное, Белое и Черное. Определим условия проведения документа.



В результате если значение перечисления Красное — документ будет проведен и по оперативному, и по бухгалтерскому учету.

Описание алгоритма Операции в модуле документа.

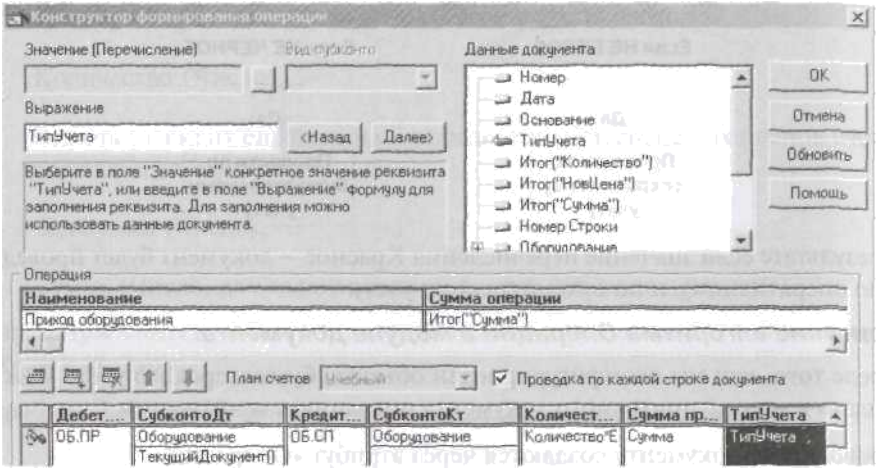
После того, как мы сконфигурировали объекты бухгалтерского учета можно создать в модуле приходного документа операцию с проводками.

Проводки по документу создаются через атрибут «Операция».

Упражнение 29. Создадим алгоритм формирования операции по документу «Приход» с учетом формирования партии приходным документом и условия списания по бухгалтерскому или оперативному учету.

В окне редактирования документа установим флаг «Бухгалтерский учет», а в модуле документа отредактируем предопределенную процедуру ОбработкаПроведения(). Операцию также можно создать с помощью конструктора формирования операций, но процедуру ОбработкаПроведения() надо будет потом отредактировать.

В конструкторе заполним атрибуты операции и проводки. Определяемый атрибут проводки выделяем в нижней таблице. Формулу, вычисляющую его значения, определяем по значению или берем из данных документа.



После редактирования модуль документа должен выглядеть примерно так:

```
//********************

Процедура ОбработкаПроведения()

ВыбратьСтроки();

Пока ПолучитьСтроку() =1 Цикл
     // Запись истории периодического реквизита справочника
     Если НовЦенаОборудование. Цена. Получить (ТекудийДокумент () ) Тогда
          ПривязыватьСтроку(НомерСтроки);
          УстановитьРеквизитСправочника(Оборудование,"Цена",НовЦена);
     КонецЕсли;
     Если ТипУчетаоПеречисление.ТипыУчета.Бел Тогда
          //}}ДВИЖЕНИЯ_РЕГИСТРОВ
          Регистр.Амортизация.Оборудование = Оборудование;
          Регистр.Амортизация.Партия = ТекущийДокумент();
          Регистр.Амортизация.Количество = Количество*
          Единица.Коэффициент;
          Регистр.Амортизация.Стоимость = Сумма;
```

```
            Регистр.Амортизация.ПривязыватьСтроку(НомерСтроки);
            Регистр.Амортизация.ДвижениеПриходВыполнить();
            //}}ДВИЖЕНИЯ_РЕГИСТРОВ
        КонецЕсли;
        Если ТипУчетаоПеречисление.ТипыУчета.Чер Тогда
            //{{ФОРМИРОВАНИЕ_ОПЕРАЦИИ
            Операция.НоваяПроводка();
            Операция.Дебет.Счет =СчетПоКоду("ОБ.ПР",ПланыСчетов.Учебный);
            Операция.Дебет.Оборудование = Оборудование;
            Операция.Дебет.Партия = ТекущийДокумент();
            Операция.Кредит.Счет=СчетПоКоду("ОБ.СП",ПланыСчетов.Учебный);
            Операция.Кредит.Оборудование = Оборудование;
            Операция.Количество = Количество*Единица.Коэффициент;
            Операция.Сумма = Сумма;
            Операция.ТипУчета = ТипУчета;
        КонецЕсли;
    КонецЦикла;
    Если ТипУчетаоПеречисление.ТипыУчета.Чер Тогда
        Операция.Содержание = "Приход оборудования";
        Операция.СуммаОперации = Итог("Сумма");
        Операция.Записать();
        //}}ФОРМИРОВАНИЕ_ОПЕРАЦИИ
    КонецЕсли;
    КонецПроцедуры
```

Обратите внимание, что, в отличие от движений регистров, для которых записывается каждое движение по строке документа, проводки по документу записываются в информационную базу все сразу при записи операции. Если возникает необходимость записать сформированные проводки, например, Чтобы посмотреть бухгалтерские итоги, используется метод **Операция.ЗаписатьПроводки().** Метод может использоваться только для атрибута "Операция" документа в момент проведения (в процедуре ОбработкаПроведения). При этом происходит обновление бухгалтерских итогов. Это позволяет при проведении документа обращаться к бухгалтерским итогам, уже измененным проводками, записанными этим документом. После

выполнения метода ЗаписатьПроводки() и до окончания процедуру ОбработкаПроведение уже невозможно изменять или удалять проводки добавленные до вызова этого метода. Данный метод имеет смысл использовать, только в том случае, если существует необходимость обращения к итогам, измененным проводками записываемой операции. При записи сложной проводки, если у главной корреспонденции сложной проводки не указана сумма (равна 0), то она автоматически вычисляется на основании подчиненных корреспонденции.

При добавлении нового субконто или валютного учета («+») по счёту в плане счетов, в модулях документов, по которым проводки формируются интерактивно, необходимо прописать движения по созданным атрибутам счёта: <тип субконто> или <ВалСумма>.

```
Операция. ВалСумма=...;
```

10.4. Бухгалтерские итоги

Бухгалтерские Итоги (далее БИ) - это служебный объект позволяющий читать и отбирать информацию из объектов метаданных бухгалтерского учета. Он имеет два режима: режим итогов и режим бухгалтерского запроса.

10.4.1. Режим

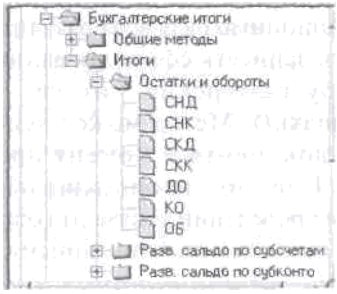
В режиме Итогов выводится информация текущего периода или любого произвольного периода. В последнем случае БИ надо рассчитать на нужный период с помощью метода Рассчитать(ДатаНачала, ДатаКонца)

БИ.Рассчитать(<НачалоПериода>,<КонецПериода>,<ФильтрПоСчетам>,<ТолькоСинтетика>, <ПланСчетов>,<РазделительУчета>);

<ФильтрПоСчетам> - необязательный параметр. Счета, для которых будет выполняться временный расчет итогов. Задается значением типа "Счет" или объектом типа "СписокЗначений", содержащим значения типа "Счет", либо строкой, содержащей список кодов счетов, разделенных символом "," или ";" ("АМ.ПР;АМ.СП") .

<ТолькоСинтетика> - необязательный параметр: 1 - рассчитывать сальдо только по счетам; 0 - или не указан - рассчитывать сальдо по счетам и по субконто.

Метод **БИ. ОсновныеИтоги()** переводит БИ в режим работы с итогами текущего периода (основными итогами). Вызов этого метода имеет смысл тогда, когда ранее был выполнен расчет временных итогов или запрос, и нужно вернуть объект к работе с основными итогами. При этом результаты запроса или расчета временных итогов теряются.



БИ.СНД(<Счет>,<ТипСуммы>,<Валюта>,<Субконто1...>)
Возвращает дебетовое сальдо по счету на начало периода.
<Счет> - значение типа "Счет" или строка - код счета.
<ТипСуммы> - необязательный параметр.
Значения: 1 ("С") - сумма(по умолчанию);
2 ("В") - валютная сумма;
3("К") - количество.
<Валюта> - необязательный параметр.
<Субконто1...> - необязательные параметры. Значения субконто. Если параметры не указаны, то итоги выдаются без учета аналитики.

Список параметров остальных бухгалтерских функций в режиме итогов такой же, кроме оборотов между счетами:

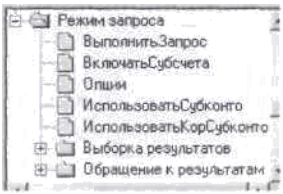
БИ.ОБ(<СчетДеб>,<СчетКр>,<ТипСуммы>,<Валюта>). Возвращает число - оборот с дебета счета <СчетДеб> в кредит счета <СчетКред>

Упражнение 30. В Форме списка справочника «Оборудование» вывести текущее конечное дебетовое сальдо по сумме конкретного оборудования (элемента справочника), по счету «Приход оборудования».

Код	Наименование	Цена	ГрЦена	Коэф. а...	График	Ост.	Сальдо
2	Системные блоки		10,000.00			36000	36000
1	PIII - 500	10,000.00		0.003		10000	10000
2	PIII - 550	11,000.00		0.003		11000	11000
5	PIII - 750	15,000.00		0.003		15000	15000
3	PIV	18,000.00		0.001		0	0

10.4.2. Режим

В режиме запроса с помощью метода БИ.ВыполнитьЗапрос(ДатаН,ДатаК) можно «построить» в оперативной памяти таблицу выборки и затем выбирать из нее необходимые данные.



БИ.ВыполнитьЗапрос(<НачалоПериода>,<КонецПериода>,<Счет>,<КоррСчет>,
<Валюта>,<ТипИтогов>,<Периодичность>,<ТипСуммы>)
<ТипИтогов> - число - тип отбираемых итогов. Значения: 1 - остатки и обороты по счету в цепом; 2 - обороты между счетами; 3 - первое и второе вместе. По умолчанию: 1.
<Периодичность> - число или символьная строка (см. документацию).Позволяет получить Дополнительный разрез итогов по периодам. По умолчанию периодичность не задана. Значения: 1("Период"); 2("Операция"); 3("Проводка"); 4 ("День"); 5 ("'Неделя'); 6 ("Декада"); 7 ("Месяц"); 8 ("Квартал"); 9 ("Год").
<ТипСуммы> - число или строка - тип рассчитываемых итогов. Значения: 1("С") рассчитывать суммы;2 ("В") рассчитывать валютные суммы; 4 ("К") рассчитывать количество. Если требуется одновременно Рассчитывать разные суммы, значение параметра получается путем сложения допустимых значений, например: 5 (1+4) - рассчитывать суммы и количество.

Перед вызовом этого метода можно установить признак отбора по субсчетам, определить фильтры по субконто и опции запроса.

БИ.Опции(<ВклЗабалансовСуммы>,<ВклОборСубкСуммы>)

Значения: 0 - суммы не включаются;
1 - суммы включаются.

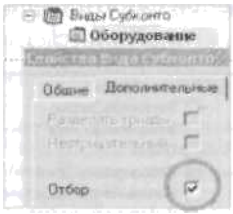
БИ.ВключатьСубсчета(<ФлагСчета>,<ФлагКоррСчета>)_Устанавливает режим отбора итогов методом "ВыполнитьЗапрос" по субсчетам.
<ФлагСчета> - необязательный параметр. Признак развертывания сальдо по субсчетам основного счета Значение:О - не разворачить по субсчетам (по умолчанию); 1 - разворачивать по субсчетам; -1 (минус единица) - не выдавать итоги по счетам-группам.

При установке признака отбора по субсчетам следует иметь в виду, что при установленном флажке «Количественный учет» «Только по аналитике» в свойствах Счетов при выборке субсчетов Вы не получите сальдо и обороты по количеству.

БИ.ИспользоватьСубконто(<ВидСубконто>,<Субконто>, <ТипФильтра>,<ПоГруппам>)
<ВидСубконто> - значение типа "ВидСубконто" (ВидСубконто.<ВидСубконто>)- расчет итогов будет выполнен только для субконто указанного вида.
<Субконто> - значение Субконто, по которому будут отобраны итоги по аналитике. Если параметр не задан - то считается пустым значением субконто.
<ТипФильтра> - Значения: 1 - разворачивать по данному субконто (по умолчанию), 2 - отбирать по данному субконто, 3 - не учитывать это субконто вообще. <ПоГруппам> - число - группировка итогов по субконто. Значения: 0 - не показывать итоги по группам справочника (по умолчанию);
1 - показывать итоги по группам справочника.

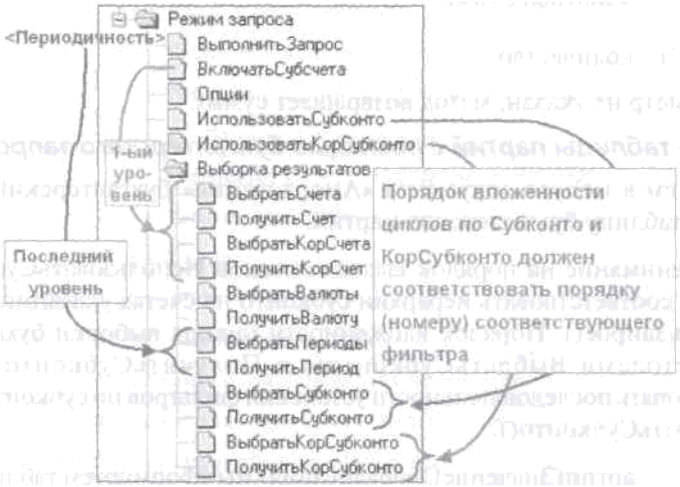
При установке фильтров по субконто надо обратить внимание на порядок их установки. Вложенность циклов выборки по субконто должна соответствовать порядку фильтров.

Тип фильтра 2 - отбор по субконто, имеет смысл использовать только для видов субконто с установленным флагом отбора (закладка «Дополнительные» свойств вида субконто).

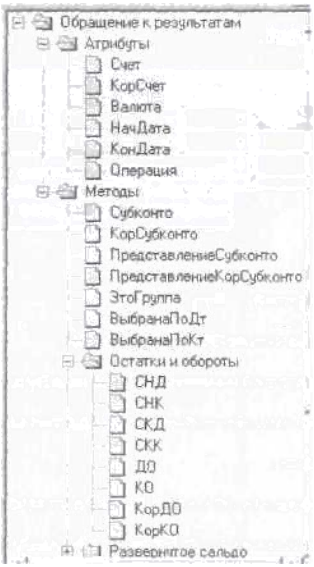


Порядок вложенности циклов выборки

Данные из бухгалтерского запроса выбираются в циклах выборки. На рисунке ниже показаны правила вложенности циклов выборки бухгалтерских итогов из запроса.



Обращение к результатам запроса



При обращении к атрибутам выборки, в отличие от методов, не надо ставить скобки - ().

При обращении в запросе к остаткам, оборотам и развернутому сальдо указывается только один параметр - ТипСуммы:

1("С") - сумма;

2("В") - валютная сумма;

3("К") - количество.

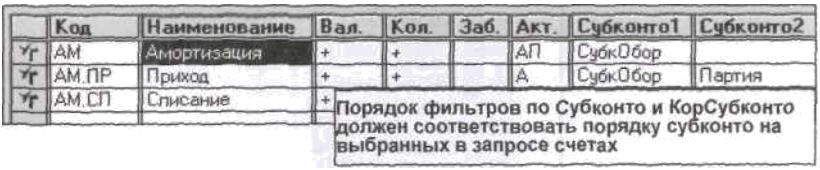
Если параметр не указан, метод возвращает сумму.

Создание таблицы партий с помощью бухгалтерского запроса

Сформируем в модуле документа «Амортизация» бухгалтерский запрос и заполним таблицу бухгалтерских партий.

Обратите внимание на порядок вызова методов ИспользоватьСубконто() -он должен соответствовать иерархии субконто на счетах указанных в методе ВыполнитьЗапрос(). Порядок вложенности циклов выборки бухгалтерских итогов методами ВыбратьСубконто() и ПолучитьСубконто() должен соответствовать последовательности установки фильтров по субконто методом ИспользоватьСубконто().

В процедуре Партии(ЗначениеОборудования) мы сформируем таблицу партий оборудования, по которым конечное сальдо больше нуля. При вызове процедуры в параметр надо передать значение оборудования, по которому необходимо получить партии.



	Код	Наименование	Вал.	Кол.	Заб.	Акт.	Субконто1	Субконто2
	АМ	Амортизация	+	+		АП	СубкОбор	
	АМ.ПР	Приход	+	+		А	СубкОбор	Партия
	АМ.СП	Списание	+					

Порядок фильтров по Субконто и КорСубконто должен соответствовать порядку субконто на выбранных в запросе счетах

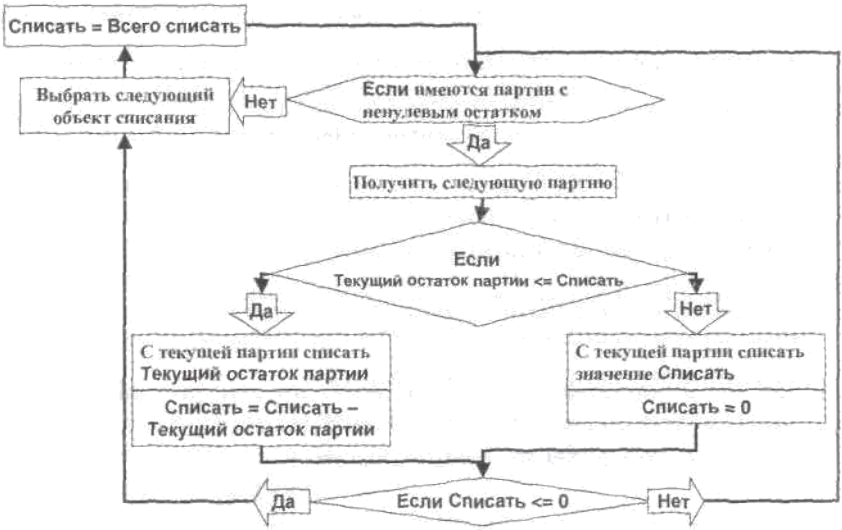
```
Перем ТаблицаБухгПартий, Ит;

Процедура Партии(ЗначениеОборудования)

    Если ПустоеЗначение(ТаблицаБухгПартий)=0 Тогда

        ТаблицаБухгПартий.УдалитьСтроки();

    КонецЕсли;

    Ит = СоздатьОбъект("БухгалтерскиеИтоги");

    Ит.ИспользоватьСубконто(ВидыСубконто.Оборудование,

    ЗначениеОборудования, 2);// Отбор

    Ит.ИспользоватьСубконто(ВидыСубконто.Партия,, 1);
```

```
Ит.ВыполнитьЗапрос(НачКвартала(ДатаДок),ДатаДок,"ОБ.ПР",,,1,,"СК");
```

```
        // Цикл по субконто 1 (Оборудование) не делаем,
        // т.к. указан отбор по субконто
        Ит.ВыбратьСубконто(2);
        Пока Ит.ПолучитьСубконто(2) = 1 Цикл
              ТаблицаБухгПартий.НоваяСтрока();
              ТаблицаБухгПартий.Партия=Ит.Субконто(2);
              // Сальдо конечное дебетовое;
              ТаблицаБухгПартий.СКДСум = Ит.СКД();
              ТаблицаБухгПартий.СКДКол = Ит.СКД(3);
        КонецЦикла;
  КонецПроцедуры
  Ит=СоздатьОбъект("БухгалтерскиеИтоги»);
  Ит.ИспользоватьПланСчетов(ПланыСчетов.Учебный);
  ТаблицаБухгПартий=Создать0бъект("ТаблицаЗначений");
  ТаблицаБухгПартий.НоваяКолонка("Партия");
  ТаблицаБухгПартий.НоваяКолонка("СКДСум");
  ТаблицаБухгПартий.НоваяКолонка("СКДКол");
```

Упражнение 31. Создадим алгоритм формирования операции по документу «Амортизация» с учетом списания оборудования по партиям.

Отредактируем процедуру ОбработкаПроведения() в документе «Амортизация». В цикле выборки строк документа, для списания по бухгалтерским партиям, будем вызывать процедуру Партии(), в цикле выбирать партии из таблицы значений и списывать их по соответствующим счетам.



Так как Оборудование списывается и по оперативному и по бухгалтерскому учету, запрет на списание, если имеется дефицит, надо снять.

```
Процедура ОбработкаПроведения()

    Перем Сообщение1;

    Рег=СоздатьОбъект ( "Регистр. Амортизация" ) ;

    ТаблицаПартий=СоздатьОбъект ( "ТаблицаЗначений" ) ;

    Если ИтогиАктуальны()=0 Тогда

        Рег.ВременныйРасчет() ;

        РассчитатьРегистрыНа(ТекущийДокумент());

    КонецЕсли;

    ВыбратьСтроки();

    Пока ПолучитьСтроку() = 1 Цикл

        // Списание по Оперативному учету

        // определим списываемое значение
```

```
ВсегоСписать=Результат;
```

```
Дефицит=Рег.СводныйОстаток(Оборудование,,"Стоимость")-
ВсегоСписать;

Если Дефицит<0 Тогда
     Сообщение1="Дефицит по остаткам "+Оборудование+" =
     "+Дефицит+"!";
     Предупреждение(Сообщение1);
     Перейти ~БухгалтерскийУчет; // Переход по метке
     // Отмена проведения документа //
     СтатусВозврата(0);
     //Возврат;
КонецЕсли;

Регистр.Амортизация.Оборудование = Оборудование;
Регистр.Амортизация.ПривязыватьСтроку(НомерСтроки);
Рег.УстановитьФильтр(Оборудование,);
Рег.ВыгрузитьИтоги(ТаблицаПартий,1,1);//Создали партии
оборудования
//ТаблицаПартий.ВыбратьСтроку(); //отладка

Если ТипСписания=Перечисление.ТипыСписания.F1FO Тогда

     ТаблицаПартий.Сортировать("Партия",1) ;

ИначеЕсли ТипСписания=Перечисление.ТипыСписания.LIFO Тогда

     ТаблицаПартий.Сортировать("Партия -") ;

Иначе // по средней себестоимости

     //Алгоритм списания по средней себестоимости

     Регистр.Амортизация.Количество =

     Результат*Рег.СводныйОстаток(Оборудование,,"Количество")/
     Рег.СводныйОстаток(Оборудование,,"Стоимость") ;

     Регистр.Амортизация.Стоимость= Результат;

     Регистр.Амортизация.ДвижениеРасходВыполнить();

     Продолжить;

КонецЕсли;

// Списание по партиям оперативного учета

ТаблицаПартий.ВыбратьСтроки();

Пока ТаблицаПартий.ПолучитьСтроку()=1 Цикл

   Регистр.Амортизация.Партия = ТаблицаПартий.Партия;

   Если ТаблицаПартий.Стоимость<=ВсегоСписать Тогда

        //Партию списываем целиком и уменьшим списываемое
        значение

        Регистр.Амортизация.Количество =
        ТаблицаПартий.Количество;

        Регистр.Амортизация.Стоимость= ТаблицаПартий.Стоимость;
        ВсегоСписать=ВсегоСписать-ТаблицаПартий.Стоимость;
```

```
        Иначе
            //С партии списываем списываемое значение целиком и
            обнуляем его
            Регистр.Амортизация.Количество = ВсегоСписать*
            ТаблицаПартий.Количество/ТаблицаПартий.Стоимость;
            Регистр.Амортизация.Стоимость = ВсегоСписать;
            ВсегоСписать=0;
        КонецЕсли;
        Регистр.Амортизация.ДвижениеРасходВыполнить();
        //Проверим: всё ли списали?
        Если ВсегоСписать <=0 Тогда
            Прервать;
        КонецЕсли;
    КонецЦикла;
    // Списание по Бухгалтерскому учету
    // определим списываемое значение
    // Установим метку безусловного перехода
    ~БухгалтерскийУчет: Списать=Результат;
    Партии(Оборудование);
    // Проверим Сальдо по сумме
    Дефицит=ТаблицаБухгПартий.Итог("СКДСум")-Списать;
    Если Дефицит<0 Тогда
        Сообщить("Бухгалтерский дефицит по оборудованию " +
        Оборудование+" = "+Дефицит ) ;
        // Операцию не формировать
        Продолжить;
    КонецЕсли;
    Если ТипСписания=Перечисление.ТипыСписания.FIFO Тогда
        ТаблицаБухгПартий.Сортировать("Партия",1);
    ИначеЕсли ТипСписания=Перечисление.ТипыСписания.LIFO Тогда
        ТаблицаВухгПартий.Сортировать("Партия -") ;
    Иначе // по средней себестоимости
        //Алгоритм списания по средней себестоимости
        Операция.НоваяПроводка();
```

```
  Операция.Кредит.Счет=СчетПоКоду("ОБ.ПР",ПланыСчетов.Учебный)
  ;
  Операция.Кредит.Оборудование = Оборудование;
  Операция.Кредит.Партия = ТекущийДокумент();
  Операция.Дебет.Счет=СчетПоКоду("ОБ.СП".ПланыСчетов.Учебный);
  Операция.Дебет.Оборудование = Оборудование;
  Операция.Количество=Результат*
       ТаблицаБухгПартий.Итог("СКДКол")/
       ТаблицаБухгПартий.Итог("СКДСум");
  Операция.Сумма = Результат;
  Продолжить;
КонецЕсли;
// Списание по партиям Бухгалтерского учета
ТаблицаБухгПартий.ВыбратьСтроки();
Пока ТаблицаБухгПартий.ПолучитьСтроку()=1 Цикл
  Операция.НоваяПроводка();
  Операция.Кредит.Счет=СчетПоКоду("ОБ.ПР",
  ПланыСчетов.Учебный);
  Операция.Кредит.Оборудование = Оборудование;
  Операция.Кредит.Партия = ТаблицаБухгПартий.Партия;
  Операция.Дебет.Счет=СчетПоКоду("ОБ.СП".ПланыСчетов.Учебный);
  Операция.Дебет.Оборудование = Оборудование;
  Операция.ТипУчета = ТаблицаБухгПартий.Партия.ТипУчета;
  Если ТаблицаБухгПартий.СКДСум <=Списать Тогда
     Операция.Количество= ТаблицаБухгПартий.СКДКол;
     Операция.Сумма = ТаблицаБухгПартий. СКДСум;
     Списать = Списать -ТаблицаБухгПартий. СКДСум;
  Иначе
     Операция.Количество= Списать*
     ТаблицаБухгПартий.СКДКол/ТаблицаБухгПартий.СКДСум;
     Операция. Сумма = Списать;
     Списать = 0;
```

```
            КонецЕсли;
            //Проверим: всё ли списали?
            Если Списать <=0 Тогда
                  Прервать;
            КонецЕсли;
         КонецЦикла;
      КонецЦикла;
      Операция.Содержание = "Списание оборудования";
      Операция.СуммаОперации = Итог("Результат");
      Операция.Записать();
КонецПроцедуры
```

Вопросы для самоконтроля

Что общего у бухгалтерского и оперативного учета?

Чем отличается бухгалтерский учет от учета на оборотных регистрах?

Чем отличается бухгалтерский учет от учета на регистрах остатков?

Что представляет из себя служебный объект Бухгалтерские Итоги?

XI. Основы построения объектов компоненты «Расчет»

Виды метаданных, используемые компонентой «Расчёт»

Журналы расчетов предназначены для хранения записей сложных периодических расчетов и их предыстории.

Календари предназначены для вычисления интервалов времени по различным графикам учета времени.

Виды расчетов - предназначены для написания алгоритмов расчетов в системе и определения приоритетов расчета и вытеснения для записи расчета.

Группы расчетов - предназначены для объединения видов расчетов, по каким - либо признакам для облегчения дальнейшей работы с ними.

11.1. Журналы расчетов

Журнал расчетов служит для просмотра и редактирования расчетов, произведенных в системе. В журнале расчетов накапливаются записи расчетов. Каждый журнал предназначен для выполнения расчетов по одному типу объектов - определенному справочнику системы. Для каждой записи журнала расчетов в качестве объекта указывается один из элементов этого справочника. Обязательно определяется вид расчета, а также как правило, указывается дата начала и дата окончания действия записи, рассчитанное значение записи записывается в атрибут Результат.

Для одного справочника могут быть созданы несколько журналов расчетов, каждый из которых будет содержать данные определенной предметной области.

Дата отсчета журнала расчетов определяет точку отсчета периода, заданного в журнале расчетов.

Если журнал расчетов уже содержит записи, система не позволяет изменить периодичность журнала расчетов.

В одной конфигурации может быть произвольное количество журналов расчетов. После создания журнала расчетов необходимо вставить в меню регламентную процедуру смены расчетного периода журнала.

Каждая строка Журнала расчетов содержит результат расчета для одного из объектов. Такие записи называются строками журнала расчетов и характеризуются следующими данными:

Объектом, для которого произведен расчет,

Видом расчета - определенным алгоритмом расчета, записанным в модуле вида расчета (начисление или удержание при расчете заработной платы),

Документом, который «породил» данную запись журнала расчетов с помощью методов ВвестиРасчет(<?>,,,,) или ЗаписатьРасчет(<?>,,,,). Например: документ начала месяца, отпуск, больничный и т.д.,

Родительским документом, который ввел данную запись журнала расчетов методом ВвестиРасчетНаОсновании(<?>,,,,,) или ЗаписатьРасчетНаОсновании(<?>,,,,,).

Протяженностью действия записи журнала расчетов (определяется датой начала и датой окончания данного расчета),

Периодом регистрации (в каком периоде журнала расчетов зарегистрирована данная запись) и периодом действия записи журнала расчетов (запись может действовать (рассчитываться) в нескольких подряд идущих периодах). Периоды журнала расчетов — это агрегатные объекты, характеризующиеся в свою очередь датой начала и датой окончания данного периода.

Результатом расчета, который рассчитывается по алгоритму, предусмотренному в модуле вида расчета, соответствующего данной записи или устанавливается при вводе записи расчета. Результат может быть исправлен пользователем.

Первичной записью, которая указывает на запись журнала расчетов, перерасчетом которой является данная запись.

Статус записи можно прочитать по значениям флажков типа число, с возможными значениями 0 и 1.

Перерасчет - показывает, что запись является перерасчетом другой (первичной) записи прошлого периода, и введена методом методом ВвестиПерерасчет() или ВвестиПерерасчетНаОсновании()

Эти методы вводят копию первичной записи, не заполняя атрибут Результат. При расчете записи перерасчета Результат будет вычислен за вычетом значения результата первичной записи.

Сторно — показывает, что запись была сторнирована. Признак 'сторно' равен 1 не только для простых сторно-записей, но и для рассчитанных, отредактированных вручную или зафиксированных (не подлежащих редактированию) сторно-записей. Если сторнирующая запись введена

программным образом, т.е. атрибут Сторно задан применением метода

УстановитьРеквизит () или непосредственным присвоением,

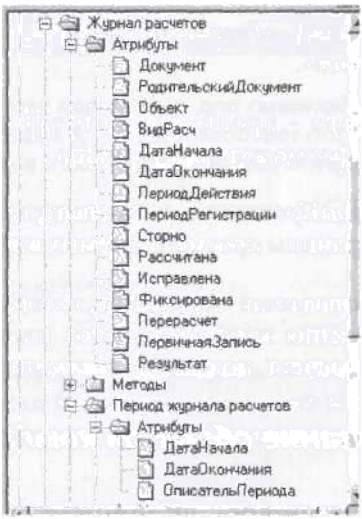
```
ЖрнРасчета = СоздатьОбъект("ЖурналРасчета.Зарплата");
ЖрнРасчета.Сторно = 1;
```

тогда он может быть переопределен программным образом.

Рассчитана — указывает, что запись рассчитана. Исправлена —

указывает, что запись исправлена вручную.

Фиксирована — указывает, что результат расчета записи защищен от исправления.



Период расчетов

Журнал расчетов предназначен для проведения повторяющихся через равные промежутки времени расчетов. Все записи журнала расчетов расположены во времени в том или ином временном интервале, называемом расчетным периодом. Расчетный период, как говорилось выше, — это агрегатный объект, характеризующиеся в свою очередь датой начала и датой окончания периода.

Величина расчетного периода может принимать одно из следующих значений: день,

неделя,

месяц,

квартал,

год.

Если журнал расчетов уже содержит записи, система не позволяет изменить периодичность журнала расчетов.

Предопределенные процедуры модуля формы журнала расчетов

ПриИсправленииРезультата - процедура, которая отрабатывает в момент исправления результата расчета записи журнала расчетов пользователем.

ПриОтменеИсправления - процедура, которая отрабатывает в момент отказа пользователя от редактирования.

ПриРасчете - процедура, которая отрабатывает в момент выполнения одной из трех команд *«РасчитатьЗапись», «РассчитатьОбъект»* или *«РассчитатьДокумент».*

ПриУстановкеОтбора - процедура, которая отрабатывает в момент установки отбора в форме журнала расчетов.

ПриУстановкеГраницыПросмотра - процедура, которая отрабатывает в момент установки границы просматриваемых в журнале расчетов записей.

ПриУстановкеПредставления - процедура, которая отрабатывает в момент установки представления журнала расчетов (по всем объектам расчетов, по одному объекту расчета, по одному документу расчета).

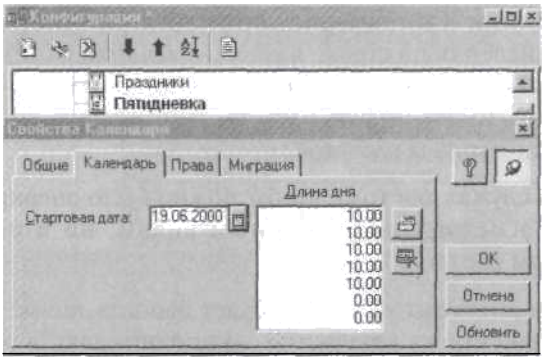
11.2. Конфигурирование объектов компоненты «Расчет»

11.2.1. Объекты метаданных - «Календарь»

Календари хранят данные о структуре периода расчетов. При конфигурировании календарей описывается порядок их автозаполнения.

В дереве метаданных выделим ветвь «Календари», в главном меню выберем пункт «Действие» и в появившемся подменю пункт «Новый элемент». В появившейся форме на закладке «Общие» введем идентификатор «Пятидневка», а на закладке «Календарь» график работы оборудования семидневку с 10-ти часовым рабочим днем и двумя выходными, стартовую

дату, начиная с которой будет вестись расчет по календарю, установим на любой понедельник. Нажав на кнопку «ОК», сохраним объект.



Упражнение 32. Создайте два календаря: для сменной работы: - 12 часов и два выходных и для оборудования, которое работает постоянно - не выключается. Введите все три календаря в пользовательский интерфейс.

В жизни иногда случаются праздники. В системе 1С:Предприятие праздники можно задать программно,

```
Праздник=СоздатьОбъект("Праздники");
Праздник.Новый('01.01.02',КоличествоРабочихЧасов);
```

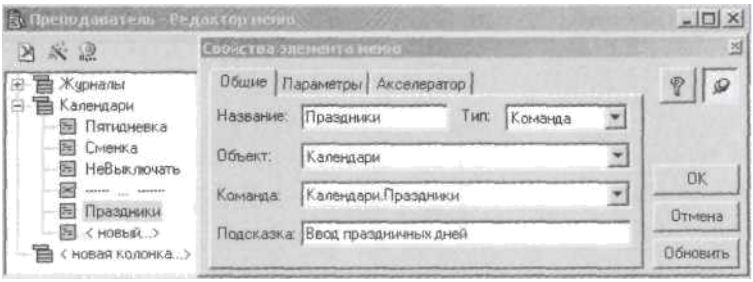
Подробнее смотрите «1С:Предприятие Версия 7.7 Описание встроенного языка Часть 2,стр.607-609. Также можно создать пункт меню в пользовательском интерфейсе, который позволит вводить праздники. Создадим такой пункт меню.

Откроем свойства нового элемента меню,

Выберем объект «Календари», Выберем команду

«Календари.Праздники», Отредактируем Название

и Подсказку.


Преподаватель - Редактор меню
Свойства элемента меню
Журналы
Календари
Пятидневка
Сменка
НеВыключать
Праздники
< новый...>
< новая колонка...>
Общие
Параметры
Акселератор
Название:
Праздники
Тип:
Команда
Объект:
Календари
Команда:
Календари.Праздники
Подсказка:
Ввод праздничных дней
OK
Отмена
Обновить

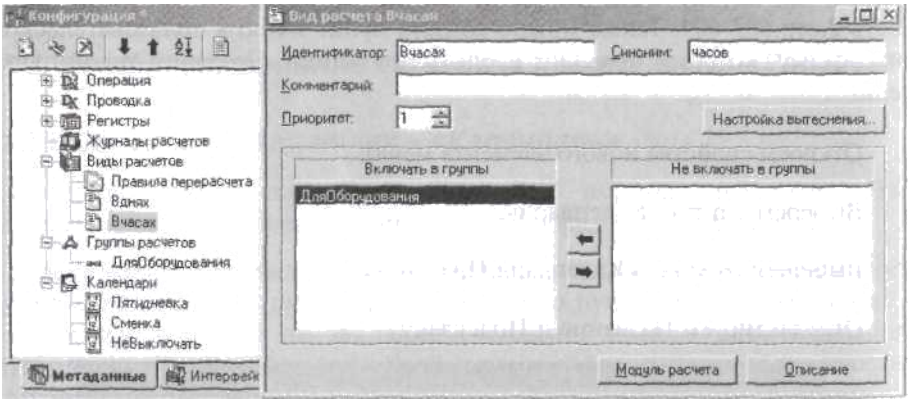
11.2.2. Создание объектов метаданных - «Группа расчетов» и «Вид расчета».

Виды расчетов служат для описания алгоритмов, по которым выполняются те или иные вычисления по строке журнала Расчетов. Объект вид расчета не имеет интерактивной формы, но, аналогично документу, у которого есть модуль документа, имеет модуль расчета, в котором доступны атрибуты и реквизиты записи журнала расчетов.

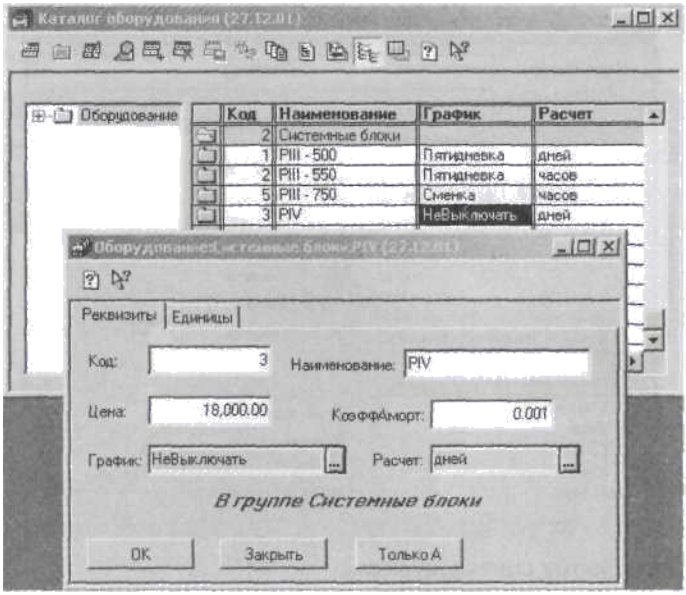
Группы расчетов служат для того, чтобы можно было оперировать не только результатами расчетов по конкретным видам, но и результатами по нескольким видам расчетов

При проведении документ «Приход» будет вводить новые записи журнала расчетов по каждой строке документа, нужно определить объекты типа вид расчета, в модулях которых будут описаны алгоритмы расчетов соответствующих записей журнала расчетов

В дереве метаданных выделим ветвь «Группы расчетов», в главном меню выберем пункт «Действие» и в появившемся подменю пункт «Новый элемент». В появившейся форме введем идентификатор «ДляОборудования». Отработанное время мы будем рассчитывать в днях и в часах. Аналогично, выделив в дереве метаданных ветвь «Виды расчетов», создадим виды расчетов «Вчасах» с синонимом «часов» и «Вднях» с синонимом «дней», и включим их в группу «ДляОборудования» с помощью стрелки.



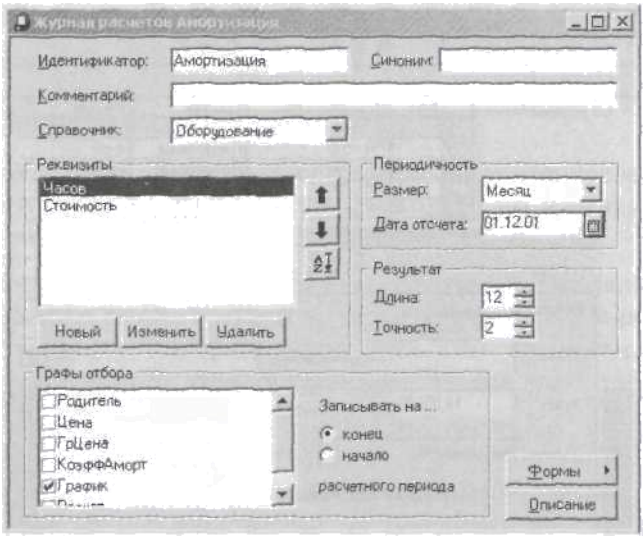
Упражнение 33. Создайте в справочнике «Оборудование» реквизит «Расчет» типа «ВидРасчета» для элемента справочника. Отредактируйте интерактивные формы и заполните все реквизиты в существующих записях справочника.



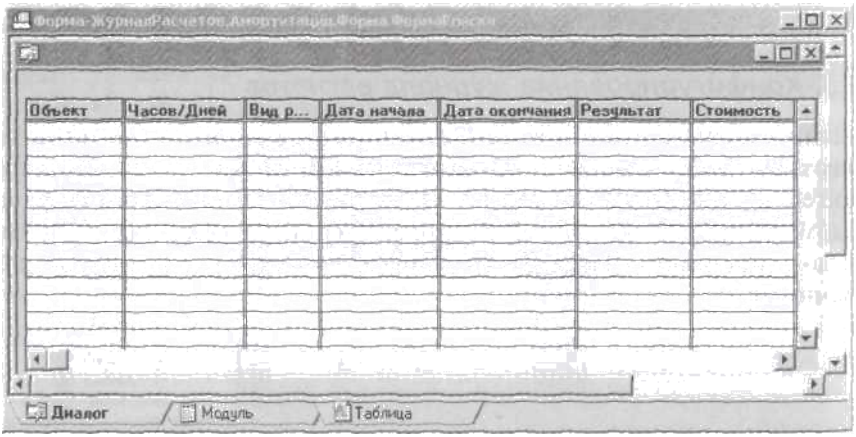
11.2.3. Конфигурирование журнала расчетов

Создадим журнал расчетов «Амортизация» с объектом расчетов типа справочник «Оборудование», с периодичностью расчетов 1 месяц, для которой характерна дата начала отсчета 1-е число месяца, размерность Результата зададим: длина -12, точность-2. В журнале создадим реквизиты типа число «Часов» с синонимом «Часов/Дней», длина 3, точность 0 и «Стоимость» , длина 12, точность 2.

Графы отбора позволяют отбирать записи журнала расчетов по значению реквизита элемента справочника-объекта. В качестве графы отбора выберем реквизит справочника «Оборудование» - «График».



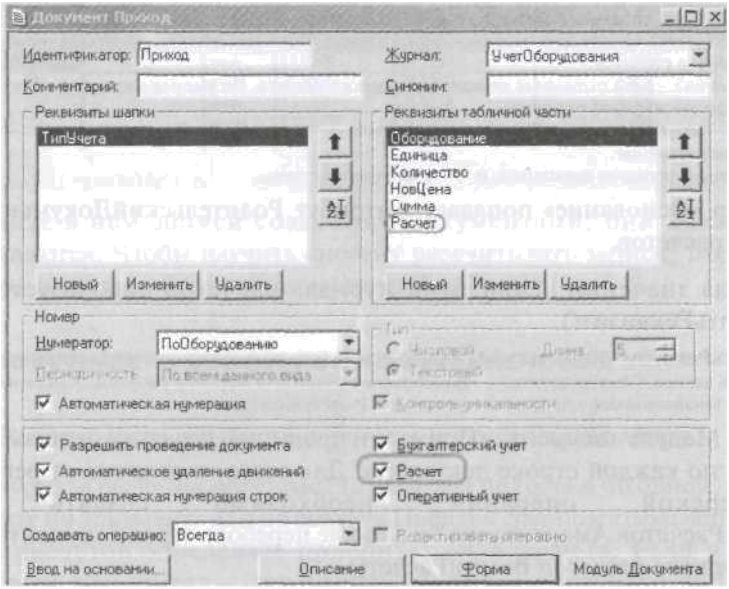
Отредактируем форму списка журнала.



11.3. Создание записей в Журнале расчетов

Записи журнала расчетов формируются документами расчета. Алгоритм создания записей журнала расчетов, как и другие движения в системе, описывается в модуле документа расчета.

Упражнение 34. Установите флаг расчета в свойствах документа «Приход» и создайте в нем реквизит табличной части «Расчет» типа «ВидРасчета», чтобы можно было откорректировать вид расчета перед вводом записи в журнал расчетов. Опишите в свойствах соответствующего поля диалога документа выражение для ввода из справочника «Оборудование» значения по умолчанию для реквизита «Расчет».



Метод *ВвестиРасчет()* позволяет ввести одним документом несколько записей в журнал расчетов по одному объекту.

Метод *ЗаписатьРасчетО* вводит документом только одну запись по объекту и виду расчета в одном периоде журнала расчетов. Если документом вводится запись по такому же объекту с таким же видом расчета, которая уже есть в данном периоде журнала расчетов, то существующая запись будет замещена новой записью.

Списки параметров для обоих методов одинаковы.

ВвестиРасчет(<Объект>,< ВидРасчета >,<ДатаНачала>,<ДатаОкончания>,<Результат>) позволяет ввести запись в журнал расчетов.
<Объект> - объект расчета - элемент справочника, заданного при конфигурировании журнала расчетов.
<ВидРасчета> - вид вводимого расчета - ссылка на агрегатный обьект ВидРасчета.
<ДатаНачала> - дата начала действия вводимого расчета. По умолчанию - дата начала текущего периода журнала расчетов.
<ДатаОкончания> - дата окончания действия вводимого расчета. По умолчанию - дата окончания текущего периода журнала расчетов.
<Результат> - результат расчета. По умолчанию - ноль.
Возвращаемое значение:
1 - если операция успешно выполнена, и 0 - в противном случае.

Для ввода записей расчета на основании существуют методы ВвестиРасчетНаОсновании() и ЗаписатьРасчетНаОсновании() с аналогичными различиями.

ВвестиРасчетНаОсновании(<Основание>,<Объект>,<ВидРасчета>,<ДатаНачала>,<ДатаОкончания>,<Результат>) позволяет ввести запись в журнал расчетов на основании произвольного документа.
<Основание> - документ, на основании которого вводится запись (или записи) в журнал расчетов.
<Объект> - объект расчета - элемент справочника, заданного при конфигурировании журнала расчетов.
<ВидРасчета> - вид вводимого расчета - ссылка на агрегатный объект ВидРасчета.
<ДатаНачала> - дата начала действия вводимого расчета. По умолчанию - дата начала текущего периода журнала расчетов.
<ДатаОкончания> - дата окончания действия вводимого расчета. По умолчанию - дата окончания текущего периода журнала расчетов.
<Результат> - результат расчета. По умолчанию - ноль.
Возвращаемое значение:
1 - если операция успешно выполнена, и 0 - в противном случае.

Параметр «Основание» попадает в атрибут РодительскийДокумент записи журнала расчетов.

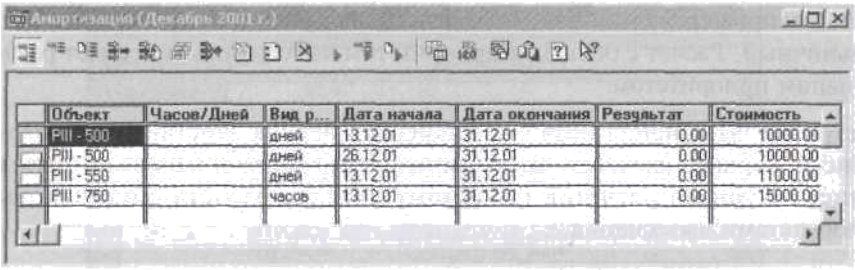
Для ввода значений реквизитов журнала расчетов используется метод УстановитьРеквизит().

УстановитьРеквизит(<ИмяРеквизита>,<Значение>) устанавливает значение реквизита журнала расчетов для записи.<ИмяРеквизита> - наименование реквизита журнала расчетов. <Значение> -значение, устанавливаемое для записи в реквизит <ИмяРеквизита>.

Откроем Модуль документа «Приход» и пропишем создание записей журнала расчетов по каждой строке документа. Для этого, в отличие от регистров и бухгалтерской операции, необходимо создать объект "ЖурналРасчетов.Амортизация" и в цикле перебора строк документа вызвать метод журнала расчетов ВвестиРасчет().

```
// ****************************************

Процедура ОбработкаПроведения()

ЖурнРасчетов=Создать0бъект("ЖурналРасчетов.Амортизация");
ВыбратьСтроки();

Пока ПолучитьСтроку() =1 Цикл
      // Создать запись журнала расчетов

      ЖурнРасчетов.УстановитьРеквизит("Стоимость",Сумма);

      ЖурнРасчетов.ВвестиРасчет(оборудование,Расчет,Датадок,,);

КонецЦикла;

КонецПроцедуры
```

Сохраните изменения, и откройте Предприятие. Перевведите оборудование в приходных документах, чтобы ввести вид расчета, перепроведите их и откройте журнал расчетов.

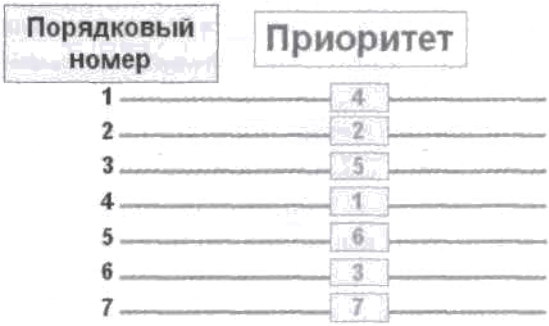


Вы увидите в нем записи созданные документами, они пока ещё не рассчитываются. Чтобы научить систему рассчитывать записи, рассмотрим подробнее свойства объектов метаданных ВидРасчета.

11.4. Особенности работы с видами расчетов

11.4.1. Очередность расчета записей журнала расчетов

Записи журнала расчетов могут рассчитываться не только по одной, но также и группами: по документу или по объекту Порядок группового расчета записей журнала расчетов определяет приоритет вида расчета. Если у записей одинаковый приоритет - они рассчитываются в порядке ввода, если приоритеты разные, то расчеты производятся в порядке возрастания приоритета.



Рекомендуется оставлять промежутки между значениями приоритетов на случай введения в конфигурацию новых видов расчетов.

11.4.2. Вытеснение записей журнала расчетов

Понятие вытесняющего расчета служит для исключения введения в одном и том же интервале дат двух или нескольких расчетов, которые исключают друг друга, например нельзя рассчитывать оклад, если в этом периоде введен больничный. Расчет с большим приоритетом вытеснения вытесняет расчет с меньшим приоритетом.

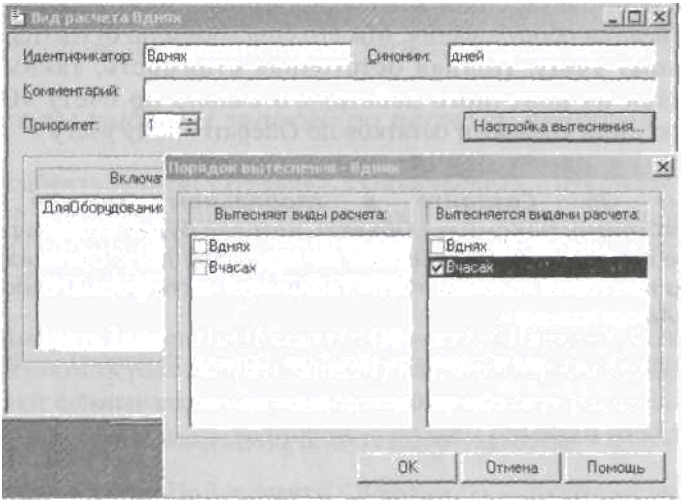
Настройка вытеснения позволяет изменить период действия вытесняемых записей журнала расчетов. В одном периоде действия не может быть двух записей журнала расчетов по одному объекту расчета с одинаковыми приоритетами вытеснения.

Разберем небольшой пример.

Сотруднику была начислена зарплата за 1-ый и 2-ой месяц. В 1-ом месяце он заболел. После болезни он вышел на работу и принес больничный лист.

Та часть вытесняющей записи по больничному листу, период действия которой более ранний, чем период регистрации, сторнирует запись по зарплате. Если период действия позже или совпадает с периодом регистрации вытесняющей записи, то период действия вытесненной записи по зарплате уменьшится на величину периода вытесняющей записи.

Откроем окно свойств вида расчетов «Вднях» и установим приоритет вытеснения так, записи с видом расчета «Вднях» вытеснялись записями с видом расчета «Вчасах».



11.4.3. Описание алгоритма расчета записи журнала расчетов в модуле вида расчета

Откроем модуль вида расчета «Вднях». В модуле вида расчета доступны атрибуты и реквизиты рассчитываемой записи журнала расчетов.

С помощью метода вида метаданных Календари - Дней(), определим количество дней отработанных оборудованием и запишем выражение для реквизита Журнала расчетов - «Часов».

Метод **Дней**(<ДатаНачала>,<ДатаОкончания>)позволяет получить количество "ненулевых" дней за период.
Возвращает: целое положительное число - количество дней.
<ДатаНачала> - дата начала периода, в котором определяется количество дней.
<ДатаОкончания> - дата окончания периода, в котором определяется количество дней. Если <ДатаНачала> больше, чем <ДатаОкончания>, будет выведено сообщение об ошибке.

Коэффициент амортизации оборудования задается у нас в процентах в час. Примем что в день у нас 10 рабочих часов для оборудования.

```
Процедура ПровестиРасчет()

    Часов=Объект.График.Дней (ПериодДействия. ДатаНачала,

        ПериодДействия.ДатаОкончания);

    Результат=Стоимость* Часов *Объект.КоэффАморт/10 ;

КонецПроцедуры
```

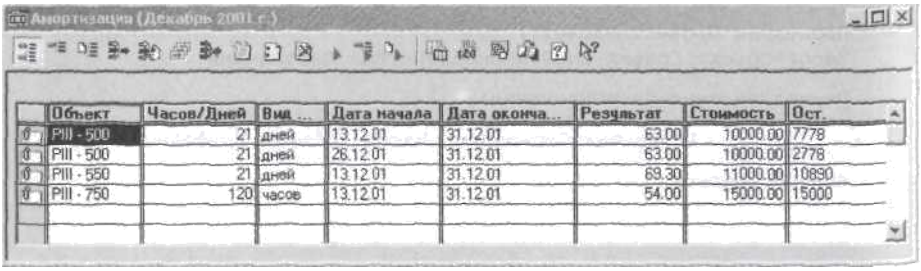
Упражнение 35. Создайте алгоритм расчета записи для вида расчета «Вчасах», используя метод календарей Часов(ДатаНачала.ДатаКонца). Рассчитайте записи в режиме Предприятия.

Определим остаточную стоимость оборудования, чтобы определить момент полного списания оборудования. Мы его списываем по бухгалтерскому и по оперативному учету. Полная остаточная стоимость, таким образом, складывается из конечного дебетового сальдо по счету «ОБ.ПР» по разделителю учета «Белое» и остатков по Оперативному учету.

Упражнение 36. Создайте в глобальном модуле функцию ОстаточнаяСтоимость(КонДата,Объект), которая будет возвращать полную остаточную стоимость оборудования на заданную дату. Организуйте вызов функции из текстовой колонки «Ост.» табличной части формы списка журнала расчетов «Амортизация».

```
Функция ОстаточнаяСтоимость(КонДата,Объект,Документ)    Экспорт
Рег=Создать0бъект("Регистр.Амортизация");
БИ=Создать0бъект("БухгалтерскиеИтоги");
БИ.ИспользоватьРазделительУчета(Перечисление.ТипыУчета.Бел);
Если КонДата<ПолучитьДатуТА() Тогда Рег.ВременныйРасчет();
      РассчитатьРегистрыНа(КонДата);
КонецЕсли;

БИ.Рассчитать(,КонДата,"ОБ.ПР",,Перечисление.ТипыУчета.Бел);
Возврат БИ.СКД("ОБ.ПР",,,Объект,Документ) +
      Рег.СводныйОстаток(Объект,Документ,"Стоимость");
КонецФункции //ОстаточнаяСтоимость()
```

В этот остаток не вошли суммы списания по себестоимости по оперативному и бухгалтерскому учету, там в Партию записано значение документа списания.



	Объект	Часов/Дней	Вид ...	Дата начала	Дата оконча...	Результат	Стоимость	Ост.
0	PIII - 500	21	дней	13.12.01	31.12.01	63.00	10000.00	7778
0	PIII - 500	21	дней	26.12.01	31.12.01	63.00	10000.00	2778
0	PIII - 550	21	дней	13.12.01	31.12.01	69.30	11000.00	10890
0	PIII - 750	120	часов	13.12.01	31.12.01	54.00	15000.00	15000

11.5. Методы выборки записей из журнала расчетов

Методы выборки записей из журнала расчетов делятся на две группы: методы осуществляющие выборку по периоду действия записи, вспомним, что запись может действовать в нескольких периодах, и методы осуществляющие выборку по периоду регистрации записи, период регистрации для записи всегда один.

11.5.1. Методы выборки записей по периоду действия

- Метод **ВыбратьЗаписи(<Начало>,<Окончание>)** - открывает выборку записей журнала расчетов. Выбираются все записи, период действия которых хоть на один день затрагивается тем периодом, который задан параметрами <Начало> и <Окончание>.

- Метод **ВыбратьЗаписиПоОбъекту(<Объект>,<Начало>,<Окончание>)** отличается от метода ВыбратьЗаписи() тем, что в первом случае в выборку попадают записи по одному конкретному объекту расчета, заданного параметром <Объект>, а во втором — по всем объектам расчета.

- Метод **ВыбратьЗаписиПоДокументу(<Документ>)** — открывает выборку всех
  записей, порожденных документом.

11.5.2. Методы выборки записей по периоду регистрации

- Методы **ВыбратьПериод(<Дата>), ВыбратьПериодПоОбъекту(<Объект>, <Дата>)** и **ВыбратьПоЗначению (<ГрафаОтбора>, <ЗначениеОтбора>, <НачПериод>, <КонПериод>)** отличаются от вышеописанных методов тем, что осуществляют выборку в другом временном разрезе. Выбираются те записи, которые введены в том расчетном периоде, в который попадает <Дата> или периоды отбора, но не записи, имеющие дату начала и дату окончания, лежащие в этом периоде. Эти методы выбирают записи по времени их появления (регистрации) в системе.

Методы работают только для переменных, созданных функцией СоздатьОбъект, и, как правило, применяются перед циклом, выполняющим перебор записей журнала расчетов при помощи метода **ПолучитьЗапись().**

Упражнение 37. В модуле формы документа «Амортизация» создадим процедуру РассчитатьКалендари() для автозаполнения календарей по дату документа и функцию ВыполнитьРасчет (объект) для расчета амортизации по записям журнала расчетов. Процедуру РассчитатьКалендари() выполнять при создании нового документа в

информационной базе. Процедуру ВыполнитьРасчет() вызывать для расчета результирующей суммы списания по

выбранному элементу справочника «Оборудование».

Метод **Автозаполнение**(<ДатаНачала>,<ДатаКонца>) выполняет автозаполнение календаря в заданном периоде. Возвращает число: 1 - получилось; 0 - не получилось <ДатаНачала> - дата начала периода автозаполнения. <ДатаКонца> - дата конца периода автозаполнения.

```
Функция ВыполнитьРасчет(объект)
Сум= 0;

ЖурнРасчетов=Создать0бъект("ЖурналРасчетов.Амортизация");

Если ЖурнРасчетов.ВыбратьЗаписиПоОбъекту(объект,
ЖурнРасчетов.НачалоПериодаПоДате(ДатаДок),
ЖурнРасчетов.КонецПериодаПоДате(ДатаДок)) =1 Тогда
      Пока ЖурнРасчетов.ПолучитьЗапись()=1 Цикл
            Если ЖурнРасчетов.Рассчитана=0 Тогда
                  ЖурнРасчетов.Рассчитать();
            КонецЕсли;
            Сум = Сум + ЖурнРасчетов.Результат;
      КонецЦикла;
Иначе
      Сообщить(" Записи по объекту "+объект+" отсутствуют !") ;
КонецЕсли;
Возврат Сум;
КонецФункции / / ВыполнитьРасчет
//_____________________________________
Процедура РассчитатьКалендари ()
Н= 1 ;
Пока Метаданные.Календарь(Н).Выбран()=1 Цикл
     Календарь=Создать0бъект(Метаданные.Календарь(Н).
                                        ПолныйИдентификатор());
     Если Календарь.Часов(НачМесяца(ДатаДок),КонМесяца(ДатаДок))=0
     Тогда
          Календарь.Автозаполнение(НачМесяца(ДатаДок),ДатаДок);
     КонецЕсли;
     Н=Н+1;
КонецЦикла;
```

```
КонецПроцедуры // РассчитатьКалендари
```

В форме диалога добавим в свойствах колонки табличной части «Оборудование» на закладке Дополнительные формулу

```
Результат= ВыполнитьРасчет (Оборудование).
```

Вызов процедуры РассчитатьКалендари () можно поместить в предопределенные процедуры ВводНового() и ВводНаОсновании().

```
Процедура ВводНового()
            РассчитатьКалендари();
            ТипСписания=Константа.ТипСписания.Получить(ДатаДок);
            УстановитьНовыйНомер("А");
КонецПроцедуры
```

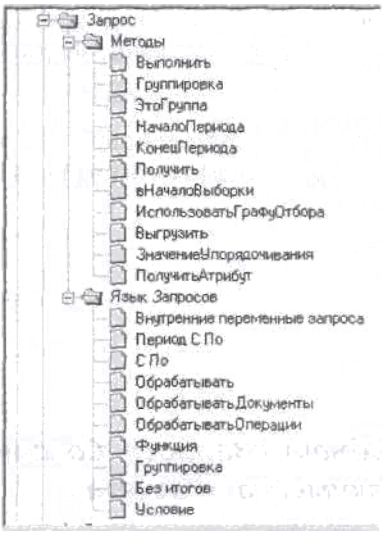
11.6. Служебный объект «Запрос». Создание регламентного документа «Расчета»

Создадим процедуру автоматического заполнения табличной части документа «Амортизация» по записям текущего периода журнала расчетов. Выборку записей можно осуществить, используя служебный объект «Запрос».

Преимущество использования этого объекта заключается в том, что при выполнении запроса к информационной базе производится комплексная выборка данных и формируется таблица выборки в оперативной памяти, после чего программа может не обращаться к таблицам информационной базы, и работать непосредственно с таблицей выборки, что существенно в многопользовательском режиме работы информационной базы.

Служебный объект «Запрос», имеет свой собственный «язык запросов» для описания создаваемой таблицы выборки. Для создания таблицы выборки текст на языке запросов передается в параметр метода Выполнить(ТекстЗапроса) объекта типа Запрос.

Метод **Выполнить**(<ТекстЗапроса>) выполняет запрос и создает таблицу выборки по запросу. Возвращает: 1 - если запрос выполнен, 0 • иначе. <ТекстЗапроса> - строка с текстом запроса на языке генерации запросов.



Описание внутренних переменных запроса — полей таблицы выборки.

ИмяПеременной = ОписаниеПеременной[, ОписаниеПеременной...];
ОписаниеПеременной - указывает на конкретный реквизит документа, справочника, измерение или ресурс регистра.
Можно указывать несколько вариантов ОписанияПеременной через запятую.
Дополнительные атрибуты внутренних переменных типа документ:
ТекущийДокумент- ссылка на текущий документ;
НомерДок - номер документа;
ДатаДок - дата документа;
НомерСтроки номер строки документа;
ВремяДок - время документа.
Дополнительные атрибуты внутренних переменных типа справочник:
ТекущийЭлемент ссылка на элемент справочника;
Код - код;
Наименование - наименование

Определение периода выборки.

[Период] С <Дата1>|<ВнешняяПеременная1> **[По** <Дата2>|Внешняя переменная2]; <Дата1>,<Дата2> - значение типа "Дата", "Документ" или позиция документа. <ВнешняяПеременная1>,<ВнешняяПеременная2> - внешняя переменная, задающая значение типа "Дата", "Документ" или позиция документа.
Если вторая часть оператора после ключевого слова По пропущена или значение второго параметра команды равно нулю, то интервал времени применяется от начального момента времени до ТА (или по РабочуюДату, если не установлена компонента "Оперативный учет"). Если в описании запроса команда Период С опущена, то интервал дат формирования запроса устанавливается в точку актуальности итогов. Для журнала расчетов конструкция Период С ... По... означает выборку в разрезе расчетных периодов журнала, а конструкция С ... По... означает выборку в разрезе времени действия записей журнала расчетов, определяемых реквизитами журнала "ДатаНачала" и "ДатаОкончания".

Установка фильтров по объектам.

Обрабатывать Все|ПомеченныеНаУдаление|НеПомеченныеНаУдаление;

ОбрабатыватьДокументы Проведенные|Непроведенные|Все;
ОбрабатыватьОперации Включенные|Выключенные|Все;

Вычисление функций и включение их результатов в таблицу выборки. Функции — это вычисляемые поля таблицы выборки. Списки вычисляемых функций различны для различных типов параметров функций.

Функция <Имяфункции> = <Типфункции>(<Параметр>|<УточненныйПараметр>)
[Когда (<Условие>)];
<ИмяФункции> - имя, присваиваемое функции;
<Типфункции> - ключевое слово одной из встроенных функций языка;
<Параметр> - имя внутренней переменной, параметр вызова функции;
<УточненныйПараметр> - конкретизация внутренней переменной, параметр вызова функции.
<Условие> - условие вычисления функции (необязательно).

Типы встроенных функций:

Сумма - сумма значений параметра;
Среднее - среднее значений параметра;
Минимум - минимальное значение параметра,
Максимум - максимальное значение параметра;
НачОст - начальный остаток значений параметра-ресурса Регистра остатков;
КонОст - конечный остаток значений параметра-ресурса Регистра остатков;
Приход - приход значений параметра-ресурса Регистра остатков;
Расход - расход значений параметра-ресурса Регистра остатков;
Счётчик - количество записей, вошедших в выборку, <Параметр> - не нужен;
СНД, СКД, СНК, СКК - сальдо начальное или конечное по дебету или кредиту;
ДО, КО, КорДо, КорКо - обороты по дебету или кредиту счета или корсчета;
В функциях; Сумма, Среднее, Максимум, Минимум в качестве аргумента возможно использование арифметического выражения в терминах встроенного языка.
К параметрам, указывающим на ресурсы Регистров, могут применяться только оговоренные функции.

Поля группировок задают порядок обхода базы данных, а также позволяют записать в таблицу выборки значения необходимых реквизитов переменных запроса, выбранных в качестве полей группировок. Выборка данных из таблицы выборки производится во вложенных циклах по полям группировок. Порядок вложенности циклов должен соответствовать порядку номеров группировок в тексте запроса, как и в бухгалтерском запросе циклы по субконто.

Группировка <ИмяГруппировки>|<ПредопредГруппировка> [Упорядочить по <КонкретизацГруппировки>, ...][Без Упорядочивания][Без Групп][Все [ВошедшиеВЗапрос]];
<ИмяГруппировки> - имя внутренней переменной, задающей группировку;
<ПредопредГруппировка> - ключевое слово одной из встроенных группировок;
<КонкретизацГруппировки> - конкретизация переменной <ИмяГруппировки>, задающая порядок групп.

Ключевые слова:

Упорядочить по - параметрами, следующими за данным ключевым словом, могут быть атрибуты и реквизиты внутренней переменной, задающей группировку. Эти параметры записываются в таблицу выборки и определяют упорядочивание строк в группировке.

Без Упорядочивания - необязательное добавочное ключевое слово, которое преследует цель уменьшения времени формирования запроса, при условии, что ни упорядочивание, ни значения упорядочивания при использовании данного запроса не нужны.

Без Групп - группы справочника не выводятся в запрос (для группировки по справочнику);

Все - в запрос выводятся все значения, и нулевые тоже (используется для группировок по справочникам и временных группировок).

ВошедшиеВЗапрос - уточняет предыдущее ключевое слово 'Все'. Использование данного слова подразумевает, что в каждую строку запроса будут включены значения данных (в том числе нулевые),

для которых есть ненулевое значение хотя бы в одной строке запроса.

Предопределенные группировки:
Документ - позволяет детализацию до документа;
СтрокаДокумента - позволяет детализацию до строки документа;
ПериодЖурнала - группировка по времени ввода записи журнала расчетов или по времени ее действия. **Группировки по дате:**
День;
Неделя;
Месяц;
Квартал;
Год.

Для увеличения скорости выполнения запроса, при условии, что итоговые записи при использовании запроса не нужны, можно не накапливать итоги по группировкам.

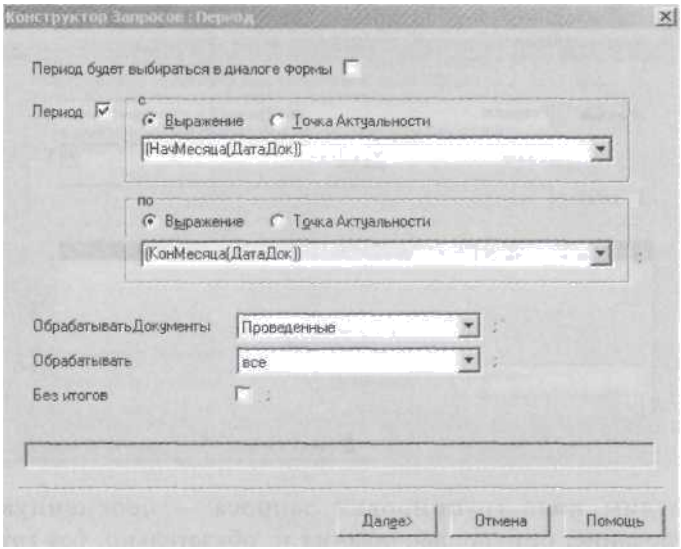
Метод **Без итогов.** В случае применения данного оператора в тексте запроса, при обходе результатов запроса применяется только один цикл обхода, используя метод объекта "Запрос" ГруппировкаО без параметра. Если в тексте запроса используется группировка по многоуровневому справочнику и не указано "Без Групп", то итоги по группам справочника будут накапливаться. Другими словами, если в запросе не нужны итоги по группам справочника, то в тексте запроса кроме использования оператора "Без Итогов" дополнительно следует в операторах "Группировка ..." использовать ключевое слово "Без Групп".

Можно назначить условие включения информации в запрос.

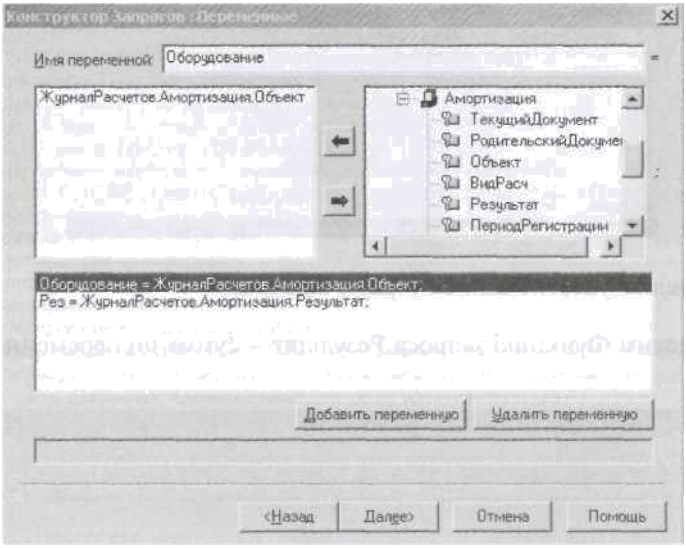
Условие(<ЛогическоеВыражение>);
В логическом выражении могут участвовать внутренние и внешние переменные. Внутренние переменные ставятся слева от логического оператора, внешние - справа.
Кроме обычных логических операторов (>, <, =, >=, <=, о. И, ИЛИ, НЕ) в операторе "Условие ..." языка запросов можно использовать дополнительный оператор: **логический оператор принадлежности - В**
Оператор языка запросов "Условие (А в Б);" говорит о том, что условие истинно, когда значение А является подмножеством значения Б. Следует особо отметить, что если значение Б пустое (объект не выбран), то условие является истинным, в отличие от оператора "=" (равно). Если на принадлежность проверяется значение типа элемент справочника, то проверка выполняется с учетом его возможного вхождения в группу справочника. Аналогично, проверка на принадлежность субсчета осуществляется с учетом его возможного вхождения в счет-группу. В качестве включающего подмножества логического оператора принадлежности (второй параметр после слова "в") может выступать как простое значение, так и список значений. В этом случае проверка выполняется с учетом вышеотмеченных особенностей для каждой строки списка значений.

Запрос можно сформировать с помощью конструктора. Войдем в модуль формы документа «Амортизация» и вызовем конструктор «Запрос » из меню «Конструкторы». Создадим запрос с именем «Заполнить».

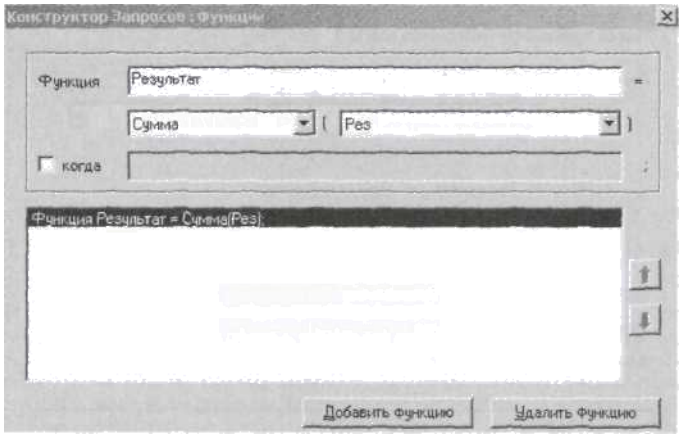
Определим период запроса.



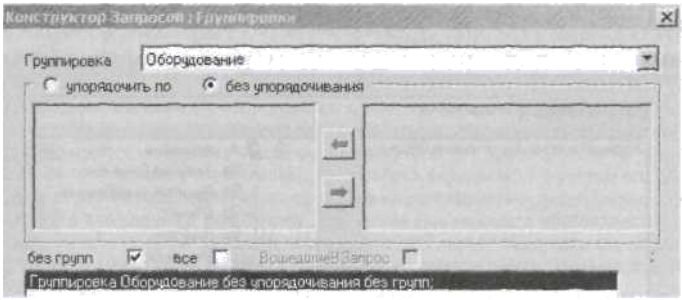
Определим переменные запроса: Оборудование и Рез - из журнала расчетов.



Определим Функцию запроса Результат — сумму по переменной запроса Рез.

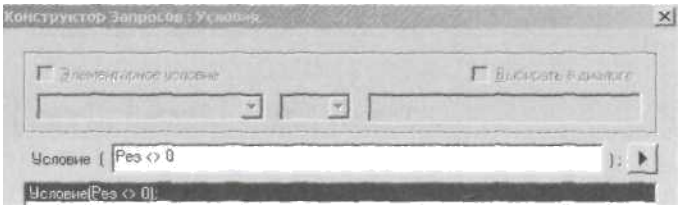


Определим поле группировки запроса — переменную запроса Оборудование, без упорядочивания и, обязательно, без групп, чтобы группы справочника в выборку не попадали, они в документе не нужны.

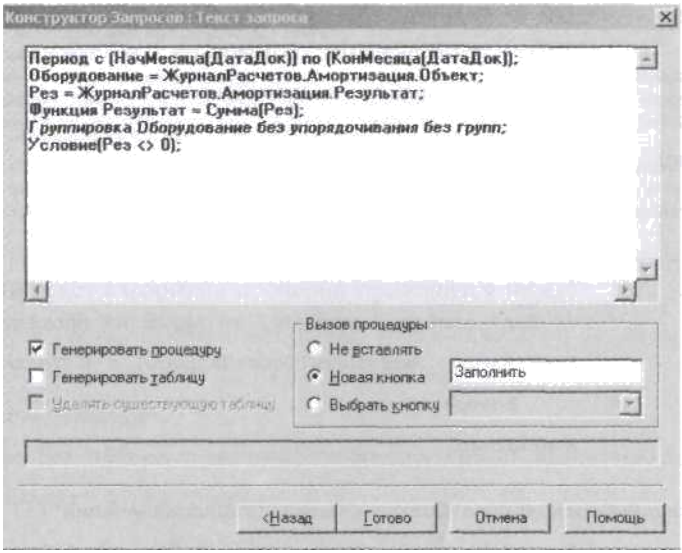


Определим условие записи строки выборки.

Определим Функцию запроса Результат — сумму по переменной запроса Рез.



Определим параметры конструктора запросов.



Отредактируем сформированную конструктором процедуру, используя следующие методы:

Метод **Выгрузить**(<ТаблЗнач>,<Флаг>,<Итоги>) позволяет выгрузить результаты запроса в таблицу значений. Возвращает число: 1 - если выгрузка произошла успешно, иначе - 0. <ТаблЗнач> - Таблица значений, куда выгружаются результаты запроса. <флаг> - необязательный параметр. Число или строка:
0- значения групп и функций (по умолчанию);
1- значения групп и функций, дополнительных переменных;
2- значения упорядочиваний групп и функций;
3- значения упорядочиваний групп и функций, дополнительных переменных;
Строка - " ТоварО), Товар(2), Товар, Склад, Приход, Расход", где ТоварО) - значение первого упорядочивания группировки "Товар". <Итоги> - необязательный параметр. Число:
0- итоги по группировкам не выводить;
1- итоги по группировкам выводить сверху (по умолчанию);
2- итоги по группировкам выводить снизу;
3- итоги по группировкам выводить сверху и снизу.
Метод **ЗагрузитьТабличнуюЧасть(<ТаблЗнач>)** позволяет загрузить многострочную часть документа из таблицы значений. <ТаблЗнач> - значение типа "Таблица значений", откуда загружается многострочная часть документа. Колонки совмещаются по идентификаторам.

```
Процедура Заполнить()

   Перем Запрос, ТекстЗапроса, ТаблицаВыборкиЗапроса;

   //Создание объекта типа Запрос

   Запрос = СоздатьОбъект("Запрос");

   ТекстЗапроса =

   "//{{ЗАПРОС(Заполнить)

   |Период с (НачМесяца(ДатаДок)) по (КонМесяца(ДатаДок));
```

```
    |Оборудование = ЖурналРасчетов.Амортизация.Объект;
    |Рез = ЖурналРасчетов.Амортизация.Результат;
    |Функция Результат = Сумма(Рез);
    |Группировка Оборудование без упорядочивания без групп;
    |Условие(Рез о 0); |" //}}ЗАПРОС

    // Если ошибка в запросе, то выход из процедуры
    Если Запрос.Выполнить(ТекстЗапроса) = 0 Тогда
        Возврат ;
    КонецЕсли;

    ТаблицаВыборкиЗапроса=СоздатьОбъект("ТаблицаЗначений");
    Запрос.Выгрузить(ТаблицаВыборкиЗапроса,0,0);
    ЗагрузитьТабличнуюЧасть(ТаблицаВыборкиЗапроса);
КонецПроцедуры
```

Процедуру Заполнить() можно вызвать из процедуры ВводНового().

Упражнение 38. (Необязательное) Создать реквизит документа формы документа «Амортизация», графу отбора журналов документов и алгоритм для проверки наличия регламентного документа в периоде расчета, которому принадлежит вводимый документ. Если в указанном периоде уже есть проведенный регламентный документ, то запретить ввод нового документа.

■Для проведенного регламентного документа разрешить только просмотр документа.

■Если регламентный документ уже есть, то при вводе нового выдать соответствующее предупреждение и открыть форму регламентного документа.

Метод **ВыбратьПоЗначению**(<Дата1>,<Дата2>,<ИмяОтбора>,<Знач>)
открывает выборку документов в интервале дат с заданным значением реквизита отбора. Возвращает:
1 - если действие выполнено и в выборке есть хотя бы один документ;
0 - если действие не выполнено или в выборке нет ни одного документа.
<Дата1> - дата, документ или позиция начала выборки документов. Если данный параметр опущен, то выборка начинается с самого первого существующего в системе документа.
<Дата2> - дата, документ или позиция конца выборки документов. Если данный параметр опущен, то выборка заканчивается самым последним существующим в системе документом.
<ИмяОтбора> - строка с названием Общего реквизита документов либо названием Графы отбора журналов;
<Знач> - значение отбора, по которому строится выборка документов.
Метод можно использовать только для объектов, созданных функцией СоздатьОбъект.

Метод **ОткрытьФормуМодально**(<Документ>,<КонтекстФормы>,<РежимПросмотра>) открывает визуальную форму существующего документа или элемента/группы справочника.
Возвращает: 1 - если действие выполнено, 0 - иначе.
<Документ> - выражение со значением типа 'документ' или 'справочник';
<Контекстформы> - имя переменной, куда можно задать значение любого типа для передачи в открываемую форму. Данное значение будет доступно в открытой форме как атрибут Форма.Параметр.
После исполнения данного метода система вернет в данную переменную контекст открытой формы (необязателен);
<РежимПросмотра> - необязательный параметр. Числовое выражение: 1 - открыть форму в режиме только просмотра; 0 - открыть форму в режиме редактирования; -1 (минус единица) - в этом случае используется вариант, предусмотренный текущим значением параметра "Режим открытия объектов", установленным пользователем интерактивно.
Пока форма открыта, тип значения параметра <КонтекстФормы> равен 100, когда закрыта - 0.

Примерный текст алгоритма проверки приводится ниже.

```
Функция ПроверитьРегламент(ДатаД =РабочаяДата())     Экспорт

    Док=Создать0бъект("Документ.Амортизация");

    ФлагРегламента=0;

    Если Док.ВыбратьПоЗначению(НачМесяца(ДатаД),КонМесяца(ДатаД),

    "Регламент",1) =1 Тогда

        Док.ПолучитьДокумент();

        Если Док.Проведен()=1 Тогда

            Предупреждение("В этом периоде существует регламентный

            документ!");

            ОткрытьФормуМодально(Док,,1);

            ФлагРегламента=1;

        КонецЕсли;

    КонецЕсли;

    Возврат ФлагРегламента;

КонецФункции //ПроверитьРегламент
```

Вызов этой функции Вам предлагается сделать самостоятельно.

Данную проверку целесообразно делать и при проведении документа «Амортизация».

Упражнение 39. (Необязательное) Создайте алгоритм, который покажет документы «Амортизация», которые надо перепровести, если изменился ТипУчета в проведенном документе «Приход». Создайте алгоритм перепроведения этих документов.

Примерный текст алгоритма в глобальном модуле учебного примера приводится ниже.

```
Перем ПерепровестиДокументы Экспорт;
```

```
Процедура Перепровести(Конт)   Экспорт
  Перем зн,поз,стр;
  Если Конт.Проведен()=1 Тогда
     СписокДокументов=Создать0бъект("СписокЗначений");
     0пер=Создать0бъект("Операция");
     Опер.ИспользоватьКорСубконто(ВидыСубконто.Партия,
     Конт.ТекущийДокумент());

     Опер.ВыбратьОперацииСПроводками(Конт.ПолучитьПозицию(),,"ОБ.СП",
     "ОБ.ПР", 2, );
     Пока Опер. ПолучитьПроводку 0=1 Цикл
        БДок= Опер.Документ.ТекущийДокумент();
        Если (БДок.Вид()="Амортизация") И
        (СписокДокументов.Принадлежит(БДок)= 0) Тогда

            СписокДокументов.ДобавитьЗначение(БДок,БДок.НомерДок);
        КонецЕсли;
     КонецЦикла;
     Рег=Создать0бъект("Регистр.Амортизация");
     Рег.УстановитьФильтр(,Конт.ТекудийДокумент());
     Рег.ВыбратьДвижения(Конт.ПолучитьПозицию(),,);
     Пока Рег.ПолучитьДвижение()=1 Цикл
          Док=Рег.ТекущийДокумент();
          Если (Док.Вид()="Амортизация") И
          (СписокДокументов.Принадлежит(Док)=0) Тогда

              СписокДокументов.ДобавитьЗначение(Док,Док.НомерДок);
          КонецЕсли;
     КонецЦикла;
     СписокДокументов.Сортировать();
     Если СписокДокументов.ОтметитьЗначения (ЗН,"Пометить
     документы",Поз)=1 Тогда
        Для i=1 по СписокДокументов.РазмерСписка() Цикл
          Если СписокДокументов.Пометка(i)=1 Тогда
```

```
                ЗначениеДокумента=СписокДокументов. ПолучитьЗначение(i,Стр);
                Если ПерепровестиДокументы. Принадлежит(ЗначениеДокумента)= 0
                Тогда

                    ПерепровестиДокументы.ДобавитьЗначение(ЗначениеДокумента,Стр);
                КонецЕсли;
            КонецЕсли;
          КонецЦикла;  //   Для
      КонецЕсли;
  КонецЕсли;
КонецПроцедуры  //Перепровести
//_____________________________________________
Процедура Перепроведение()
  Перем Стр;
  Если ПустоеЗначение(ПерепровестиДокументы)=0 Тогда
       Док=СоздатьОбъект("Документ.Амортизация");
       Для i=1 по ПерепровестиДокументы.РазмерСписка() Цикл
          Док.НайтиДокумент(ПерепровестиДокументы.
                                 ПолучитьЗначение(i,Стр));
          Если Док.Провести()=1 Тогда
               Сообщить("Документ "+Док+ " проведен.");
          Иначе
               Сообщить("Документ "+Док+ " НЕ ПРОВЕДЕН! !!!!!!!!");
          КонецЕсли;
       КонецЦикла;   //ПерепровестиДокументы.РазмерСписка()
  КонецЕсли;
КонецПроцедуры //Перепроведение
//_____________________________________________
...
...

ПерепровестиДокументы=СоздатьОбъект("СписокЗначений");
```

Упражнение 40. Постройте самостоятельно (Конструктором отчетов) Оборотные ведомости по бухгалтерским итогам, с помощью конструктора бухгалтерских запросов, и по регистру остатков, используя конструктор запросов. Покажите Начальные и конечные остатки (сальдо) и обороты по счету «ОБ.ПР» и, аналогично, по движениям Приход и Расход регистра остатков.

Вопросы для самоконтроля

Какое основное отличие компоненты Расчет от учетных компонент?

Можно ли включать один и тот же ВидРасчета в записи различных журналов расчета?

Можно ли вести в одном журнале расчета записи по разным объектам учета?

Атрибуты какого элемента данных непосредственно доступны в модуле расчета?

Можно ли в одном календаре вести индивидуальные графики всех объектов учета?

XII. Связь с другими базами данных

12.1. Загрузка и выгрузка через текстовый файл с помощью служебного объекта Текст

Текст модуля обработки выгрузки в текстовый файл справочника «Оборудование».

Метод **ВыбратьФайл**(<ТипДиалога>,<ИмяФайла>,<ИмяНачКаталога>, <ЗаголовокОкна>,<Фильтр>,<Расширение>,<Таймаут>) открывает окно диалога выбора/сохранения файла. Возвращает: 0 - если в окне диалога нажата кнопка 'Отмена', 1 - если нажата кнопка 'ОК'.
<ТипДиалога> : 0 - диалог типа <открыть>, 1 - диалог типа <сохранить>;
<Имяфайла> - переменная, содержащая на входе строку с именем файла, а на выходе - имя выбранного файла;
<ИмяНачКаталога> - переменная, содержащая на входе строку с начальным каталогом, а на выходе -имя выбранного каталога; <ЗаголовокОкна> - строка с заголовком окна;
<фильтр> - строка с фильтром отбора файлов (например: 'Все файлы (*.*) [*.*');
<Расширение> - строка с расширением файла по умолчанию;
<Таймаут> - время ожидания отклика пользователя в секундах (необязателен).

```
Процедура Сформировать()
    Перем ИмяФайла,ИмяКаталога;
    Спр=СоздатьОбъект("Справочник.Оборудование");
    ТекстВыгрузки=Создать0бъект("Текст");
    Спр.ВыбратьЭлементы();
    Пока Спр.ПолучитьЭлемент()=1 Цикл
       Если Спр.ЭтоГруппа()=1 Тогда
              ТекстВыгрузки.ДобавитьСтроку("Г"+
                    Спр.Наименование+"#"+Спр.ГрЦека+"#");
       Иначе
ТекстВыгрузки.ДобавитьСтроку(""+Спр.Уровень()+Спр.Наименование+
"#"+Спр.Цена.Получить(РабочаяДата())+"#"+Спр.КоэффАморт+"#");
       КонецЕсли;
    КонецЦикла;
    ФС.ВыбратьФайл(1,ИмяФайла,ИмяКаталога,"Файл выгрузки",
                          "Файлы выгрузки(*.txt) | *.txt","*.txt") ;
    ТекстВыгрузки.Записать(ИмяКаталога+ИмяФайла);
КонецПроцедуры
Процедура ПриОткрытии()
```

```
            Сформировать();
            // Чтобы интерактивная форма не открывалась
    СтатусВозврата(0);
    Возврат;
    КонецПроцедуры
```

Текст модуля обработки загрузки из текстового файла. Справочник «Загрузка» должен иметь реквизиты «ГрЦена», «Цена», «КоэффАморт» такие же, как в справочнике «Оборудование».

Обратите внимание на алгоритм загрузки элемента справочника в определенную группу (используется переменная модуля «Группа») и на алгоритм последовательного «обрезания» строк текстового файла.

Метод **Лев**(<Строка>,<Число>) возвращает строку, содержащую первые (самые левые) символы текстовой строки.
<Строка> - строка, содержащая извлекаемые символы;
<Число> - количество символов, которое должна вернуть функция.
Метод **Прав**(<Строка>,<Число>) возвращает строку, содержащую последние (самые правые) символы текстовой строки.
<Строка> - строка, содержащая извлекаемые символы;
<Число> - количество символов, которое должна вернуть функция.
Метод **Сред**(<Строка>,<Число1>,<Число2>) возвращает подстроку исходной строки, заданную номером позиции и числом символов. <Строка> - строка, содержащая извлекаемые символы;
<Число1> - определяет позицию первого символа, извлекаемого из строки (начиная с 1);
<Число2> - количество символов, которое должна вернуть функция (если опущен, то до конца строки).
Метод **НайТи**(<Строка1>,<Строка2>) возвращает позицию первого вхождения в строку поиска заданной подстроки.
<Строка1> - строка в которой ищем (место поиска);
<Строка2> - строка которую ищем (шаблон поиска). Если не находит - возвращает число 0. Первая позиция имеет индекс 1.
Метод **СтрДлина**(<Строка>) возвращает длину строки.
<Строка> - строковое выражение.
Метод **ИспользоватьДату**(<Дата>,<УстСразу>) позволяет установить дату выборки периодических реквизитов справочника. Возвращает текущее значение используемой даты (на момент до исполнения метода).
<Дата> - значение типа дата.
<УстСразу> - необязательный параметр. Число: если 1, то дата, переданная в качестве параметра, будет установлена уже в текущей выборке; если 0 - то дата, переданная в качестве параметра, будет установлена при следующем открытии выборки. Значение по умолчанию - 0.
Если к объекту применен метод ИспользоватьДату, то нельзя применять к этому же объекту метод Получить. Метод нельзя использовать через две точки.

```
    Перем Группа;
    //_____________________________________________________________

    Функция Проверка(Пар)
        Пров=Создать0бъект("Справочник.Загрузка");
        Возврат Пров.НайтиПоНаименованию(пар,0,1);
    КонецФункции //Проверка
```

//

```
Функция Обрез(С,Ф)
    С=Прав (С,СтрДлина (С)-Ф) ;
    Ф=Найти(С,"#");
    Возврат Лев(С,Ф-1) ;
КонецФункции //Обрез
Процедура Сформировать()
    Перем ИмяФайла,ИмяКаталога;
    спр=СоздатьОбъект("Справочник.Загрузка");
    // Назначить дату записи периодических реквизитов
    Спр.ИспользоватьДату(РабочаяДата());
    ТекстЗагрузки=Создать0бъект("Текст");
    ФС.ВыбратьФайл(0,ИмяФайла,ИмяКаталога,"Файл выгрузки",
           "Файлы выгрузки(*.txt) | *.txt","*.txt");
    ТекстЗагрузки.Открыть(ИмяКаталога+ИмяФайла);
    Для н=1 по ТекстЗагрузки.КоличествоСтрок() Цикл
           Стр= ТекстЗагрузки.ПолучитьСтроку(Н);
           Если Лев(Стр,1)="Г" Тогда
                  Спр.НоваяГруппа();
                  Фл=Найти(Стр,"#");
                  Спр.Наименование=Сред(Стр,2,Фл-2);
                  Стр=Прав(Стр,СтрДлина(Стр)-Фл);
                  Фл=Найти(Стр,"#");
                  Спр.ГрЦена=0брез(Стр,Фл);
                  Если Проверка(Спр.Наименование)=0 Тогда
                         Спр.Записать();
                  КонецЕсли;
                  Группа=Спр.ТекущийЭлемент();
           Иначе
                  Если Число(Лев(Стр,1))>1 Тогда
                         Спр. ИспользоватьРодителя (Группа);
                  Иначе
                         Спр.ИспользоватьРодителя
              (ПолучитьПустоеЗначение("Справочник.Загрузка"));
```

```
                    КонецЕсли;
                    Спр.Новый();
                    Фл =Найти(Стр,"#");
                    Спр.Наименование=Сред(Стр,2,Фл-2);
                    Стр=Прав(Стр,СтрДлина(Стр)-Фл);
                    Фл=Найти(Стр,"#");
                    Спр.Цена=Число(Обрез(Стр,Фл));
                    Спр.КоэффАморт=Число(Обрез(Стр.Фл));
                    Если Проверка(Спр.Наименование)=0 Тогда
                       Спр.Записать();
                    КонецЕсли;
              КонецЕсли;
      КонецЦикла;
КонецПроцедуры
Процедура ПриОткрытии()
    Сформировать();
    СтатусВозврата(0);
    Возврат;
КонецПроцедуры
```

12.2. Загрузка и выгрузка через файл формата dBase с помощью служебного объекта Xbase

Текст модуля обработки выгрузки в файл справочника «Оборудование».

Метод **АвтоСохранение**(<Режим>)позволяет установить режим автоматического сохранения изменений в базе. Возвращает текущее числовое значение режима автоматического сохранения изменений в базе (на момент до исполнения метода) <Режим> - выражение: 1 - задает режим автоматического сохранения изменений в базе, 0 - снимает (по умолчанию - 0)

```
//**********************************************
Процедура Сформировать()
    Перем Имяф,ИмяК;
    ДБ=Создать0бъект("XBase");
    ДБ.ДобавитьПоле("Code" , 2 ,10 ,);  //"", 10 ,
    ДБ.ДобавитьПоле("Descr",2 , 25 , ) ;
    ДБ.ДобавитьПоле("Acc",1,10,2) ;
    ДБ.ДобавитьПоле("Gr" , 2 , 2 5 , ) ;
    ДБ.ДобавитьПоле("GGr",1,1,0) ;
```

```
ФС.ВыбратьФайл(1,Имяф,ИмяК,
            "Сохраните файл","Файл БД (*.dbf) | *.dbf","*.dbf");
ДБ.СоздатьФайл(Имяф,ИмяК);
ДБ.СоздатьИндексныйФайл(Лев(Имяф,СтрДлина(Имяф)-3)+"cdx");
Об=Создать0бъект("Справочник.Оборудование") ;
ДБ.АвтоСохранение(1) ;
Об.ИспользоватьДату(ТекущаяДата());
Об.ВыбратьЭлементы();
Пока Об.ПолучитьЭлемент()=1 Цикл
    ДБ.Добавить();
    ДБ.Code=06.Код;
    Если Об.ЭтоГруппа()=1 Тогда
           ДБ.Acc=0;
           ДБ.GGr=i;
           ДБ.Descr=06.Наименование+"V";
    Иначе
        ДБ.Descr=Об.Наименование;
        ДБ.Acc=Об.Цена;
        ДБ.GGr=0;
        Если Об.Уровень()>1 Тогда
           ДБ.Gr=Об.Родитель. Наименование+''V" ;
        КонецЕсли;
    КонецЕсли;
КонецЦикла;
Дб.Записать();
//!!!!! Записать последнюю запись
Дб.ЗакрытьФайл();
КонецПроцедуры
Процедура ПриОткрытии()
    Сформировать();
    СтатусВозврата(0);
    Возврат;
КонецПроцедуры
```

Текст модуля обработки загрузки справочника «Оборудование» из файла.

Добавляет записи в существующий справочник «Оборудование».

Метод **Первая()** позволяет перейти на первую запись. Возвращает:1 - если действие выполнено; 0 - если действие не выполнено.

Метод **Следующая()** позволяет перейти на следующую запись. Возвращает: 1 - если получена следующая запись; 0 - иначе.

```
Перем ДБ,Об;
Процедура ЗаписьЭлемента()
          Если ДБ.GGR=1 Тогда
               Об.НоваяГруппа();
               Об.Наименование=ДБ.DESCR;
          ИначеЕсли Дб.GR<>ПолучитьПустоеЗначение() Тогда
               Об.НайтиПоНаименованию(Дб.GR,0,1);
               Об.ИспользоватьРодителя(Об.ТекущийЭлемент());
               Об.Новый ();
          Об.Наименование=ДБ.DESCR;
          Об.Цена= ДБ.ACC; КонецЕсли;
          Об.Записать();
КонецПроцедуры //ЗаписьЭлемента
//_____________________________________________________________

Процедура Загрузить()
          Перем Имяф,ИмяК;
          ДБ=Создать0бъект("XBase");
          ФС.ВыбратьФайл(0,Имяф,ИмяК,
"Сохраните файл","Файл БД (V*.dbf) | *.dbf");
          Дб.ОткрытьФайл(Имяф,Лев(Имяф,СтрДлина(Имяф)-
3) +"cdx" , 0) ;
          Если Дб.Открыта()=0 Тогда
               Предупреждение("База не открыта!");
               Возврат;
          КонецЕсли;
          0б=Создать0бъект("Справочник.Оборудование");
          Об.ИспользоватьДату(ТекущаяДата());
          Если Дб.Первая()=1 Тогда
               ЗаписьЭлемента();
          Иначе
```

```
                    Возврат;
                КонецЕсли;
                Пока Дб.Следующая()=1 Цикл
                     ЗаписьЭлемента();
                КонецЦикла;
                Дб.ЗакрытьФайл();
                ОткрытьФорму("Справочник.Оборудование.ФормаСписка",);
КонецПроцедуры Процедура ПриОткрытии()
                Загрузить();
                СтатусВозврата(0);
                Возврат;
КонецПроцедуры
```

Приложение 1. Метод «ОткрытьПодбор» и предопределенная процедура модуля формы ОбработкаПодбора()

Рассмотрим другой метод заполнения форм объектов на примере заполнения табличной части документа «Приход».

Если Вы хотите при открытии формы списка выбирать сразу несколько значений, а не открывать её по каждой строке документа, можно использовать метод «ОткрытьПодбор», который открывает форму списка для множественного подбора и передает выбранное значение предопределенной процедуре ОбработкаПодбора().

Метод
ОткрытьПодбор(<Объект>,<ИмяФормы>,<Контекстформы>,<ФлагМножВыбора>,<ТекЗнач>)
открывает форму для подбора значений.
<Объект> - строка с именем объекта агрегатного типа для подбора. Можно указывать: "Справочник.XXXXX" или "Документ.XXXXX" или "Журнал.XXXXX" или "Журнал.Подчиненные" или "ЖурналОпераций.XXXXX", "ПланСчетов.XXXXX" (если XXXXX не задан, то открывается подбор из любого(всех) плана счетов) или "Отчет.XXXXXX' или, "Обработка.XXXXXX", где XXXXX - имя вида соответствующего объекта, как он задан в конфигураторе.
<Имяформы> - строка с именем Формы подбора;
<Контекстформы> - необязательный параметр. Имя переменной, куда можно задать значение любого типа для передачи в открываемую форму. Данное значение будет доступно в открытой форме как атрибут Форма.Параметр. После исполнения данного метода система вернет в данную переменную контекст формы подбора. С помощью значения этого контекста можно затем произвольно манипулировать формой подбора, пока она открыта. Пока форма открыта, тип значения данного параметра равен 100 (см. ТипЗначения), если закрыта - 0.
<ФлагМножВыбора> - число: 1 - выбор нескольких значений; 0 - выбор одного значения, после чего окно закрывается;
<ТекЗнэч> - необязательный параметр. В случае выбора из списка, здесь можно передать значение, на которое следует изначально установить курсор при открытии формы подбора.
Доступ к методу возможен только в контексте Модуля формы.

Предопределенная процедура обработки подбора значения
ОбработкаПодбора(<Элемент>,<КонтФормы>)
<Элемент> - элемент справочника подбора или документ, передаваемый для обработки.
<КонтФормы> - контекст той формы, из которой шел подбор. Процедура может располагаться только в программном модуле формы.

```
//Пример использования метода «ОткрытьПодбор» в форме документа
«Приход».
Перем НомСтроки,СписокРасчетов;
Перем СписокРасчетов,  НомСтр;
// В шапке модуля
//_____________________________________________
Процедура Подбор()
   СписокРасчетов=Создать0бъект("СписокЗначений");
   Для н=1 по  ГруппаРасчетов. ДляОборудования.Количество() Цикл
```

```
расч= ГруппаРасчетов. ДляОборудования.ПолучитьРасчет(н);
```

```
            СписокРасчетов.ДобавитьЗначение(расч, расч.Наименование);
  КонецЦикла;
  ОткрытьПодбор("Справочник.Оборудование", " ДляПодбора " , , 1, );
КонецПроцедуры //Подбор
//--------
Процедура ОбработкаПодбора(ВыбранноеЗначение)
Перем ЗначениеРасчета, Позиция;
Если ВыбранноеЗначение.Выбран()=1 Тогда
  Если ВыбранноеЗначение.Вид()="Оборудование" Тогда
     НоваяСтрока();
     НомСтр=НомерСтроки; Оборудование=ВыбранноеЗначение;
     Контф_мы=ВыбранноеЗначение;
     ОткрытьПодбор("Справочник.Единицы","",Контф_мы,0);
   Иначе
      ПолучитьСтрокуПоНомеру(НомСтр;
      Единица=ВыбранноеЗначение;
      цена_=Оборудование.Цена.Получить(ДатаДок);
      Кол_во=1;
      Если ВвестиЧисло(Кол_во,"Введите количество",6,2)=1 Тогда
           Количество=Кол_во;
           Если ВвестиЧисло(цена_,"Введите цену",10,2)=1 Тогда
              НовЦена=цена_;
           КонецЕсли;
      КонецЕсли;
      Сумма=Кол_во*цена_*Единица. Коэффициент;
      Если СписокРасчетов.ВыбратьЗначение(ЗначениеРасчета,
                                  "Выберите расчет".Позиция,,0)=1 Тогда
           Расчет= ЗначениеРасчета;
      КонецЕсли;
  КонецЕсли;
КонецЕсли;
КонецПроцедуры
```

Процедуру Подбор() надо поместить в соответствующую кнопку, аналогично кнопке «Печать».В форме списка справочника Единицы надо определить Владельца при её программном открытии методом ОткрытьПодбор("Справочник.Единицы", «», Контф_мы, 0). Значение Владельца передается через параметр Контф_мы.

Для этого опишем в модуле формы списка справочника Единицы предопределенную процедуру ПриОткрытии(). Значение параметра формы в открываемой форме списка равно значению «Контекста формы» (Контфмы), ТипЗначения()=11 — это справочник.

```
Процедура ПриОткрытии()
Если  (ТипЗначения(Форма.Параметр)=11)   Тогда
  //Для метода ОткрытьПодбор из документа
  // Передаём значение типа справочник
ИспользоватьВладельца(Форма.Параметр,0);
КонецЕсли;
КонецПроцедуры //ПриОткрытии
```

Приложение 2. Совокупность основных «*.dbf» файлов, реализующих хранение

Файлы	Назначение
1SUSERS	Системный файл: контроль соединений и обновлений информации пользователями.
1SSYSTEM	Системный файл; содержит общие параметры информационной базы (точку актуальности, рассчётнный период бухгалтерских итогов, периодичность оперативных итогов и т.д.).
1SCONST	Значения констант и периодических реквизитов справочников и бухгалтерских счетов.
1SJOURN	Содержит заголовки всех документов (внутренний идентификатор, номер, дату, время, общие реквизиты, по которым установлен отбор)
1SCRDOC	Содержит вхождения документов в графы отбора, списки подчиненных документов, вхождения документов в общие журналы, для которых определен состав документов.
1SDNLOCK	Содержит временный список номеров документов, которые в данный момент вводятся, для автоматической нумерации документов с учетом вводимых.
ISUIDCTL	Используется для дополнительного контроля уникальности внутренней идентификации объектов (документов, справочников, бухгалтерских счетов).
ISBLOB	Содержит значения реквизитов справочников, документов, счетов имеющих тип «Строка неограниченной длины». Также содержит описания шаблонов типовых операций.
SC*	Содержит данные справочника конкретного вида. Каждый справочник хранится в отдельном файле
DT*	Содержит данные реквизитов табличной части документа конкретного вида. Создается при наличии реквизитов реквизитов табличной.
DH*	Содержит данные реквизитов шапки, общих реквизитов без признака «Отбор» документа конкретного вида и итоговые суммы по колонкам тех реквизитов табличной части, для которых установлен признак "Итог по колонке". Создается при наличии реквизитов.
RA*	Содержит движения регистра конкретного вида.
RG*	Содержит итоги регистра конкретного вида (остатки для регистров остатков, обороты для оборотных регистров).
1SACCS	Содержит список бухгалтерских счетов всех планов счетов
1SOPER	Содержит данные бухгалтерских операций (сумму, содержание, доп. реквизиты). Содержит одну строку на каждый документ, по которому создана операция.
1SENTRY	Содержит бухгалтерские проводки
1SBKTTLC	Содержит рассчитанные бухгалтерские итоги оборотов между синтетическими счетами.
1SBKTTL	Содержит рассчитанные бухгалтерские итоги остатков и оборотов

	по синтетическим счетам и объектам аналитики.
1SCORENT	Содержит список корректных проводок.
1SACCSEL	Содержит вхождения проводок в отборы по бухгалтерским счетам.
1SSRSFT	Содержит список вхождений проводок в отборы по субконто
1STOPER	Содержит список типовых операций.
CJ*	Содержит данные журнала расчетов конкретного вида.
CJPROP	Содержит свойства журналов расчетов (расчетный период, глубина просмотра и т.п.)
CL*	Содержит данные календаря конкретного вида.
1SUPDTS	Системный файл компоненты «Управление распределенными ИБ». Содержит таблицу регистрации изменений.
1SDWNLD	Системный файл компоненты «Управление распределенными ИБ». Содержит таблицу регистрации выгрузок.
1SDBSET	Системный файл компоненты «Управление распределенными ИБ».

Приложение 3. Некоторые ошибки, выдаваемые системой

Возвращаемые коды ошибок:

Код	Причина ошибки
-10	Ошибка закрытия файла
-20	Ошибка создания файла
-30	Ошибка определения длины файла
-40	Ошибка установки длины файла
-50	Ошибка при попытке заблокировать файл
-56	Ошибка захвата файла. В течение 60 сек. система не смогла захватить файл. Машина, на которой запущена система 1С:Предприятие - не база, перегружена другими задачами.
-60	Ошибка при открытии файла
-70	Ошибка чтения файла
-80	Ошибка удаления файла
-90	Ошибка переименования файла
-100	Ошибка позиционирования в файле
-110	Ошибка снятия блокировки с файла
-120	Ошибка записи в файл
-200	Файл не является базой данных DBF-формата
-210	Неопознанное имя поля
-220	Неопознанный тип поля
-230	Запись слишком длинная
-300	Индексный файл не содержит информации о за-писи
-310	Ссылки в индексном файле или разрушены, или указывают на несуществующие элементы базы данных
-920	Не достаточно оперативной памяти

ДЛЯ ЗАМЕТОК

Отпечатано в ИПО -Лев Толстой.» Зак.304

1С:ПРЕДПРИЯТИЕ

система программ

Практическое пособие рассчитано на пользователей, знакомых с интерфейсом «1С:Предприятие» и имеющих навыки программирования. Основное назначение - помочь начинающим разработчикам в освоении методов конфигурирования системы «1С:Предприятие». Пособие используется в качестве методического материала при прохождении курса по настройке системы «1С:Предприятие», а также может быть рекомендовано для самостоятельного изучения программирования на платформе V7. Материалы, изложенные в пособии, охватывают все виды объектов системы «1С:Предприятие». Предложенные для рассмотрения примеры раскрывают основные свойства и методы объектов системы, а также приемы организации учета хозяйственной деятельности средствами системы «1С:Предприятие».

ИКС ТЕХНОЛОГИИ