ПРИДНЕСТРОВСКИЙ ГОСУДАРСТВЕННЫЙ
УНИВЕРСИТЕТ им. Т. Г. ШЕВЧЕНКО

Рыбницкий филиал
кафедра прикладной информатики
кафедра физики, математики и информатики

Visual FoxPro 9.0 SP1

Часть II

Лабораторный практикум

Рыбница,
2010

УДК 681.3.06
ББК 32.973.2-018
С 44

Visual FoxPro 9.0 SP1 (Часть II): лабораторный практикум / Л.К. Скодорова, А.А. Ляху – Рыбница, 2010. – 179 с.

Данное учебное пособие предназначено для студентов III курса специальности ПИВЭ (351400) и ПОВТ и АС (220400). Лабораторный практикум представляет собой подбор теоретического и практического материала с целью обучения студентов основным приемам разработки реляционной базы данных. Методичка содержит шесть лабораторных работ. Цикл предложенных лабораторных работ разработан согласно рабочей программе по дисциплине «Базы данных» в соответствии с государственным образовательным стандартом. В пособии приведен сквозной пример по разделам выполнения всего практикума.

Авторы: Л.К. Скодорова, доцент кафедры прикладной информатики,
А.А. Ляху, ст. преп. кафедры физики, математики и информатики

Рецензенты: И.А. Павлинов, доцент кафедры прикладной информатики
А.Б. Глазов, ст. преп., кафедры физики, математики и информатики

Рекомендовано НМС ПГУ им. Т.Г. Шевченко

СОДЕРЖАНИЕ

Введение

Учебное пособие «Visual FoxPro 9.0 SP1 (Часть II)» является продолжение работы по изучению системы управления базами данных. Visual FoxPro – это объектно-ориентированный, визуально программируемый язык, управляемый по событиям, которые в полной мере соответствует новым требованиям, предъявляемы к современным средствам проектирования и реализации программного обеспечения.

В Visual FoxPro реализованы все атрибуты реляционных систем управления базами данных. В базе данных поддерживается целостность данных с помощью первичных ключей и связей между таблицами. Для обработки событий добавления, удаления или изменения записей таблиц самой базы данных можно использовать триггеры и хранимые процедуры.

Вторая часть содержит шесть лабораторных работ. Каждая работа содержит дополнительный материал практического характера, раскрывающего некоторые особенности работы с Visual FoxPro 9.0 SP1, отсутствующие в большинстве русскоязычных изданиях. Темы лабораторных работ подобраны из расчета последовательности выполняемых действий при разработке реальных приложений и являются логическим продолжением первой части.

Лабораторная работа № 7

Тема: Создание отчетов

Цель: Научиться создавать отчеты с помощью конструктора отчетов. Размещать в отчете различные объекты и производить настройку их.

Общие понятия

Под отчетом понимается формированное представление данных, выводимое на экран, принтер или в файл.

Прежде чем приступить к созданию отчетов вы должны ответить на следующие вопросы:

- С какой целью создается настоящий отчет, и чем он будет вам полезен?
- Какая информация, и из каких таблиц должна быть предоставлена в отчете?
- Какого вида вы предполагаете создать отчет (табличный, в свободной форме или наклейки)?
- Предполагается ли группировка данных?

Четкие ответы на поставленные вопросы облегчат вашу работу при создании отчета.

При знакомстве со средствами Visual FoxPro для создания экранных форм вы могли убедиться в их гибкости и мощности. Аналогичные средства Visual FoxPro предоставляются в ваше распоряжение для создания отчетов. При создании отчетов вы

можете воспользоваться стандартными средствами, ускоряющими процесс создания отчета, или разработать для отчета специальный формат, с помощью *конструктора отчетов*. Конструктор отчетов позволяет создавать отчеты как в табличном виде, так и в свободной форме.

Табличный отчет представляет собой напечатанную таблицу, в которой данные упорядочены по столбцам и строкам. Каждый из столбцов отчета содержит поле исходной таблицы или вычисляемое поле, а строка представляет собой запись. Табличный отчет позволяет напечатать данные из таблиц в наиболее простом и естественном виде. Однако табулированное представление данных в отчете имеет свои недостатки. На практике в ряде случаев (почтовые этикетки, чеки, письма и т.д.) поля исходной таблицы должны располагаться в специально отведенных для них местах отчета. Очевидно, что табличный отчет не пригоден для этих целей.

Отчеты в свободной форме позволяют устранить ограничения, свойственные табличным отчетам. При получении отчета в свободной форме вы можете воспользоваться стандартным форматом, автоматически создаваемым Visual FoxPro для каждой таблицы. В этом формате поля исходной таблицы расположены вертикально. Однако с помощью конструктора отчетов вы можете разработать специальный формат отчета, где поля исходной таблицы будут расположены в требуемых местах отчета.

В Visual FoxPro для создания отчетов используются:

- **Мастер отчетов (Report Wizard)**, позволяющий достаточно быстро создать отчет, выбрав параметры сортировки и группировки данных, стиль отображения данных и их расположение;
- **Стандартный отчет (Quick Report)**, позволяющий создать стандартный отчет, в котором поля отчета располагаются автоматически по внутреннему алгоритму Visual FoxPro;
- **Конструктор отчета**, в котором вы самостоятельно разрабатываете собственные отчеты.

Окно конструктора отчетов

Любой отчет состоит из пояснительного текста, полей отчета и рамок. Текст носит произвольный характер. Рамки служат для улучшения восприятия информации. Поля отчета могут непосредственно соответствовать полям исходной таблицы или являться результатом вычисления над ними.

При создании и модификации отчетов конструктор отчетов позволяет вам удалять, добавлять, перемещать области вместе с расположенными в них объектами. Вы можете установить цвет и управлять параметрами отображения любых элементов и областей отчета.

Открыть окно конструктора отчетов при создании нового отчета можно одним из следующих способов:

- Выполните команду **File | New**. В появившемся окне диалога «New» установите опцию **Report** и нажмите кнопку **New File**;
- Нажмите кнопку **New** в окне проекта, выбрав предварительно группу «Reports», и в открывшемся диалоговом окне **New Report** выберите кнопку **New Report;**
- Нажмите кнопку **New** на стандартной панели инструментов. В открывшемся окне диалога «New» установите опцию **Report** и нажмите кнопку **New File**.

Для работы в конструкторе отчетов используются панели инструментов Report **Designer** (конструктор отчетов) и **Report Controls** (элементы управления отчетов) (табл.1), а также команды пункта **Report**.

На рис. 1 представлено окно конструктора отчетов с панелями инструментов «Report Designer» и «Report Controls». Обратите внимание на то, что в строке основного меню появился пункт **Report**.

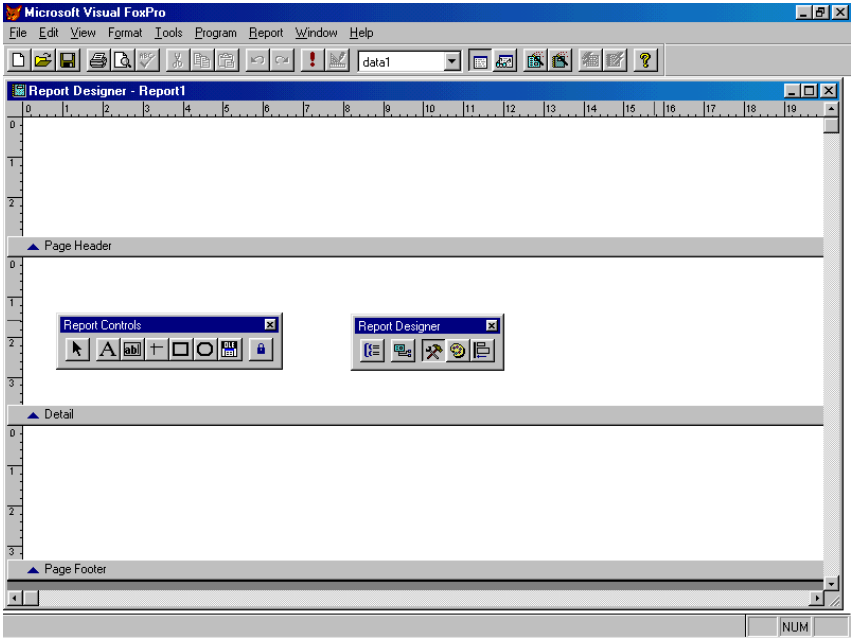

Рис. 1. Окно конструктора отчета

.

Таблица 1

Кнопки панели инструментов «Report Controls»

Кнопка	Наименование	Назначение
	Select Objects	Является указателем выбора объектов отчета
	Label	Размещает текст
	Field	Размещает поля
	Line	Рисует линии
	Rectangle	Рисует прямоугольники
	Rounded Rectangle	Рисует прямоугольники со скругленными краями
	Picture/OLE Bound Control	Помещает в отчет рисунок
	Button Lock	Закрепляет выбор кнопки

Вся рабочая область конструктора отчетов по умолчанию разбивается на три полосы, ограничиваемые разделительными строками. Наименование полосы отображается на разделительной строке, находящейся непосредственно под этой полосой. При использовании в отчете группирования данных, добавлении в него титульной страницы и итоговых данных появляются дополнительные полосы. Каждая полоса может содержать элементы управления отчетов, такие как, текст, табличные поля, вычисляемые поля, линии, прямоугольники и рисунки.

Типы полос, возможных в отчете (табл. 2).

Основное назначение полосы – определять, когда и где будут печататься размещенные в полосе объекты.

Таблица 2.

Типы полос отчета

Наименование	**Описание**
Title (Титульный лист)	Информация, появляющаяся перед основным отчетом и называется титульной. Титулом может быть имя отчета, сопроводительное письмо или любая информация, которую необходимо поместить на первой странице отчета.
Page Header (Верхний колонтитул страницы)	Информация, которая печатается в начале каждой страницы (верхний колонтитул). Обычно в этой полосе содержится название отчета, текущая дата, номер страницы и т.д.
Group Header (Верхний колонтитул группы)	Информация, используемая при группировке. При группировке данных группа может иметь верхние полосы, печатаемые до нее. Они помогают идентифицировать информацию, содержащуюся на каждом уровне группировки.

Detail (Детали)	Данные полей из таблицы или результат вычислений над полями из таблицы.
Group Footer (Нижний колонтитул группы)	Итоговая информация по группе.
Page Footer (Нижний колонтитул страницы)	Информация, содержащая название отчета, дату, номер страницы и итоговые значения по данным текущей страницы.
Summary (Итоги)	Информация, появляющаяся один раз после основного отчета и содержащая итоговые значения или заключительный текст, подводящий итог содержимого отчета.

Вы можете управлять высотой полосы для добавления в нее текста, полей или просто, чтобы добавить в свой отчет пробелы. Чтобы изменить высоту полосы, выберите нужную полосу (вы увидите двустороннюю стрелку) и опустите ее границу вниз, если вам нужно больше места. В противном случае поднимите границу полосы вверх. Можно также дважды нажать мышью на полосе и ввести высоту в появившемся окне диалога.

Создание стандартного отчета

Процесс создания отчета может включать в себя все или часть из приведенных ниже процедур:

- Определение окружения.
- Размещение объектов в отчете: текста, полей, линий, прямоугольников и рисунков.
- Группированные данные.

- Сохранение отчета.
- Просмотр созданного отчета в окне предварительного просмотра.
- Печать отчетов.

Ускорить размещение данных в отчете можно с помощью команды **Quick Report** (Быстрый отчет) из меню **Report**. Отчет, получаемый в результате выполнения этой команды, называется быстрым или стандартным отчетом. Это средство конструктора отчетов, которое автоматически помещает выбранные поля и надписи к ним в окно конструктора отчета. После этого можно модифицировать полученный отчет, изменив текст надписей, порядок расположения полей, добавить в отчет группирование данных, заголовок и т. д.

Для создания отчета с использованием команды **Quick Report** необходимо выполнить следующие действия:

1. Откройте базу данных проекта.
2. Откройте любым удобным для вас способом окно конструктора отчета.
3. Выберите в меню **Report** команду **Quick Report.**
4. В открывшемся диалоговом окне **Open** содержится список всех таблиц открытой в проекте базы данных (рис. 2). Выберите таблицу, для которой создается стандартный отчет, и нажмите кнопку **ОК.**
5. После выбора таблицы открывается окно **Quick Report,** в котором предлагаются команды расположения полей в отчете

– В столбец или строку. Выберете один из предложенных вариантов:

- при нажатии левой кнопки поля будут размещены в полосе **Detail** (Детали) слева на право по всей страницы;
- при нажатии правой кнопки поля будут размещаться в полосе **Detail** (Детали) друг над другом.

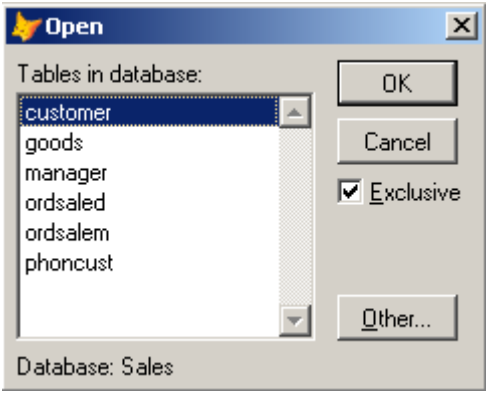

Рис. 2. Окно выбора источника данных

Диалоговое окно **Quick Report** содержит флажки, описание которых приводится в табл. 3.

Таблица 3.

Флажки диалогового окна **Quick Report**

Флажок	**Назначение**
Titles (Заголовок)	При установке флажка в отчет помещаются поля и надписи к ним

Add alias (Добавить псевдоним)	Флажок определяет, указывать ли псевдоним таблицы в именах полей в окне конструктора отчета
Add table to data environment (Добавить таблицу в среду окружения)	При установке флажка используемая в отчете таблица помещается в среду окружения

6. Если необходимо разместить в отчете все поля исходной таблицы, то этот шаг пропустите и сразу нажмите **ОК**, чтобы закрыть диалоговое окно **Quick Report.** Для выбора полей, размещаемых в отчете, нажмите Fields (Поля). Откроется диалоговое окно **Fields Picker** (Выбор поля) (рис. 3). Выберите поля, которые необходимо поместить в отчет, используя для этого кнопку **Move** (Переместить). Если в отчет необходимо переместить все поля, воспользуйтесь кнопкой **All** (Все).

7. Если необходимо разместить в отчете все поля за исключением нескольких, сначала выберите все поля, а затем удалите лишнее, воспользовавшись кнопкой **Remove** (Удалить).

8. Завершив выбор полей, нажмите кнопку **ОК** для закрытия диалогового окна **Field Picker**. Нажмите также кнопку **ОК** в окне **Quick Report** (Быстрый отчет). Теперь отчет содержит все необходимые поля. Кроме того, в полосе Page Footer (Нижний колонтитул) расположено поле с функцией DATE () и поле с системной переменной _PAGENO, указывающие дату и текущий номер страницы отчета соответственно.

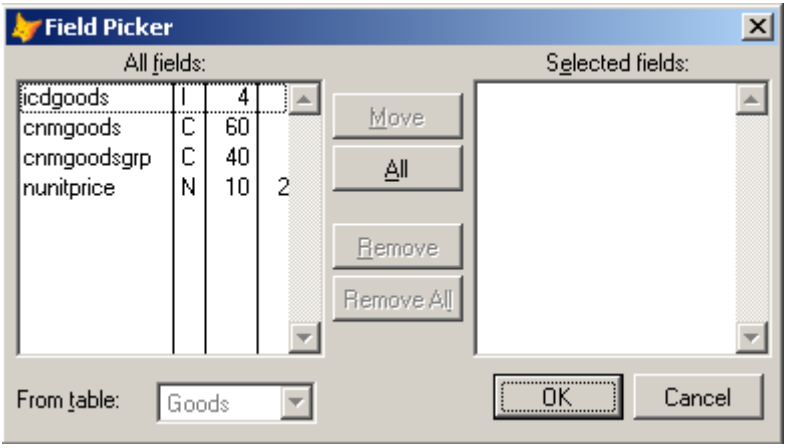

Рис. 3. Диалоговое окно **Field Picker**

Фрагмент отчета, сформированного с помощью команды **Quick Report** представлен на рис.4.

Если текст и наименование полей в отчете отображаются некорректно, необходимо изменить шрифт этих объектов. Для этого выполните следующие действия:

1. Выберите все объекты отчетов, воспользовавшись командой **Select All** (Выбрать все) из меню **Edit** (Правка).
2. Откройте диалоговое окно **Шрифт**. Для этого в меню Format выберите команду **Font** (Шрифт).
3. Используя список **Шрифт** диалогового окна **Шрифт**, установите необходимый шрифт.
4. Задайте начертание, стиль, цвет и размер символа выделенных объектов.

5. Завершив установку параметров, нажмите кнопку **ОК** для закрытия диалогового окна.
6. Щелкните мышью в любом месте отчета вне выделенной области.

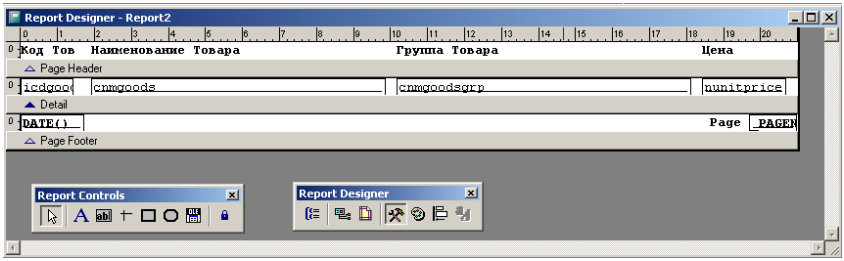

Рис. 4. Фрагмент отчета, созданного с помощью команды **Quick Report**

Установка среды окружения отчета

Для отчета, созданного мастером или с помощью команды **Quick Report** , среда окружения отчета уже определена. Разработчику, создающему отчет с помощью конструктора отчетов, среду необходимо сформировать, выполнив для этого следующие действия:

- разместить в окружении используемые в отчете таблицы;
- установить для таблиц необходимые индексы;
- установите отношения между таблицами.

Вся эта информация, относящаяся к среде окружения, хранится в файле описания отчета. Для формирования среды

окружения отчета используется окно **Data Environment** (Среда окружения).

Для открытия окна «Data Environment», в котором осуществляется настройка среды окружения (рис. 5), вы можете воспользоваться одним из следующих способов:

- Выполните команду **View | Data Environment** основного меню;
- Выберите команду **Data Environment** контекстного меню.

В среде окружения необходимо разместить все таблицы, используемые в отчете. Для добавления таблицы в окно используется команда **Add** контекстного меню или **Data Environment | Add** основного меню. Таблицы, связанные между собой в базе данных, переносятся в окно диалога «Data Environment» с сохранением связей, установленных между ними. В этом окне можно также устанавливать связи между таблицами или изменить существующие.

Как и в формах, при создании отчетов можно использовать адаптер курсора. Для этого в контекстном меню расположены две команды. **Add CursorAdapter** (Добавить адаптер курсора) добавляет в среду окружения отчета адаптер курсора. Команда **Builder** (Построитель) запускает построитель, осуществляющий настройку адаптера курсора.

После размещения таблиц в среде окружения отчета, необходимо упорядочить данные, находящиеся в таблицах. Для этого выполните следующее действия:

1. Выделите таблицу, в которой хотите упорядочить данные.
2. Откройте окно свойств таблицы. Для этого установите на нее курсор, нажмите правую кнопку мыши и выберите в появившемся контекстном меню команду **Properties** (Свойства).
3. Выделите свойство Order (Порядок) (рис. 6).
4. В поле изменения свойства нажмите кнопку раскрытия списка. Из списка индексов таблицы выберите тот, по которому необходимо упорядочить данные в отчете.

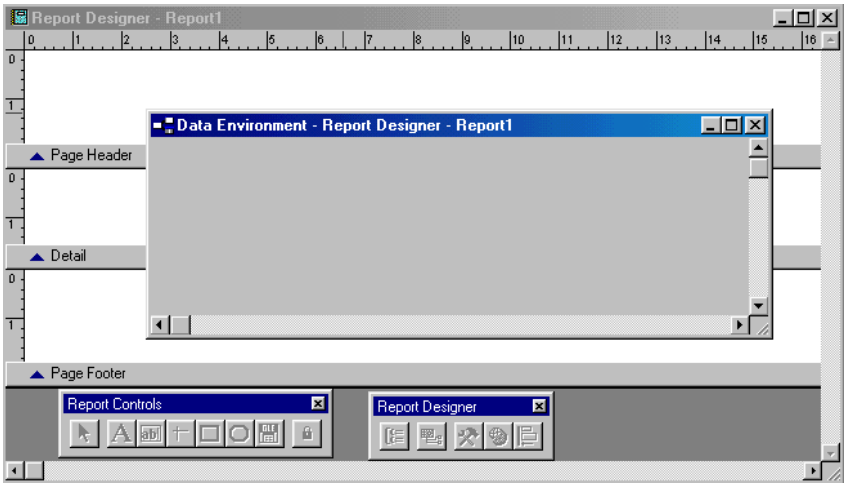

Рис. 5. Окно диалога «Data Environment»

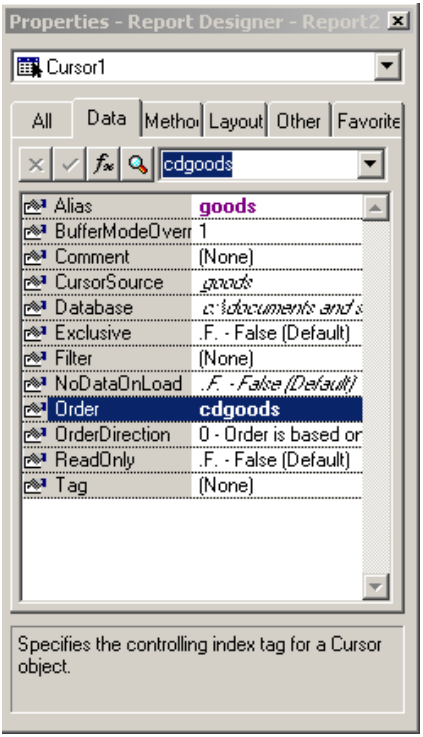

Рис. 6. Свойство **Order** используется для упорядочения записей в таблице

Если в отчете используется несколько связанных таблиц, необходимо убедиться, что установлены так, как требуется для создания правильного отчета. Для этого выполните следующие действия:

1. Выделите линию, соединяющею таблицы. При этом в окне свойств **Properties** будут отображаться свойства, характеризующие установленную связь.

2. Проверте, какая из таблиц является родительской, а какая дочерней по отношению к ней. Для этого просмотрите свойства ChildAlias (Дочерняя таблица) и ParentAlias (Родительская таблица).
3. Просмотрите выражение, по которому связаны таблицы. Для этого воспользуйтесь свойством RelationalExpr (Выражение отношения).

После размещения в окне Data Environment (Среда окружения) все используемых в отчете таблиц, закройте его, после чего Visual FoxPro сохранит созданную среду окружения.

Свойство объектов отчета

Любой отчет состоит из объектов: текста, полей отчета, разделительных линий и рамок, графических изображений. Для настройки их свойств используются команды меню, а также диалоговое окно, которое можно открыть одним из следующих способов:

- дважды щелкните кнопкой мыши на объекте;
- выберите из контекстного меню объекта команду **Properties.**

Окно свойств содержит несколько вкладок, количество которых различно для разных объектов (рис.7).

Вкладка General (Общие) окна свойств, предназначена для настройки общих параметров объекта.

Расположение объекта в полосе отчета задается переключателем **Object position** (Положение объекта), который содержит опции, приведенные в табл. 4.

Таблица 4.

Опции переключателя **Object position**

Опция	Назначение
Float (Плавающие)	Позиция объекта в отчете может измеряться при изменении размеров окружающих его объектов
Fix relative to top of band (Постоянное относительно верхнее полосы)	Объект сохраняет постоянную позицию относительно верхней границы полосы
Fix relative to bottom of band (Постоянное относительно нижней полосы)	Объект сохраняет постоянную позицию относительно нижней границы полосы

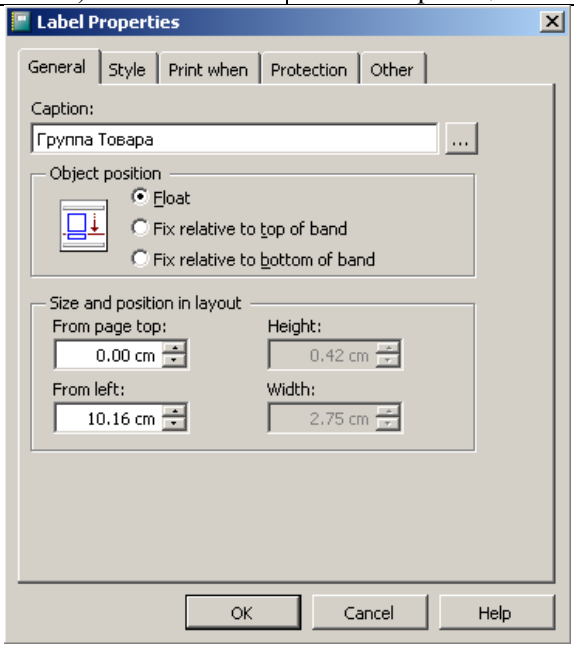

Рис. 7. Диалоговое окно **Label Properties**

Область **Size and position in layout** (Размер и положение в шаблоне) содержит поля, определяющие расположение объектов относительно страницы отчета. Их назначение в табл. 5.

Таблица 5.

Поля области **Size and position in layout**

Поле	Назначение
From page top (От заголовка страницы)	Расстояние от заголовка страницы до объекта
From left (От левой стороны страницы)	Расстояние от левой стороны страницы до объекта
Height (Высота)	Высота объекта
Width (Ширина)	Ширина объекта

Вкладка Style (Стиль) диалогового окна свойств (рис.8) предназначена для настройки оформления объекта. Для текстовых объектов отчета (надпись, поле) на вкладке **Style** расположена область **Font** (Шрифт), предназначенная для настройки оформления объекта (Табл. 6).

Таблица 6.

Флажки области **Font**

Флажок	Назначение
Use font script (Использовать набор символов)	Указывает на использование набора символов, заданного при настройке шрифтов в диалоговом окне Шрифт
Strikethrough (Зачеркнутый)	Текст перечеркнут сплошной линией
Underline (Подчеркнутый)	Текст подчеркнут сплошной линией

Область **Color** (Цвет) содержит флажки, приведенные в табл. 7.

Таблица 7.

Флажки область **Color**

Флажок	Назначение
Use default foreground (pen) color (Использовать для цвета текста значение по умолчанию)	Указывает, какой цвет использовать для текста, значение устанавливается по умолчанию или задается с помощью диалогового окна **Цвет**
Use default background (fill) color (Использовать для цвета фона значение по умолчанию)	Указывает, какой цвет использовать для фона, значение устанавливается по умолчанию или задается с помощью диалогового окна **Цвет**

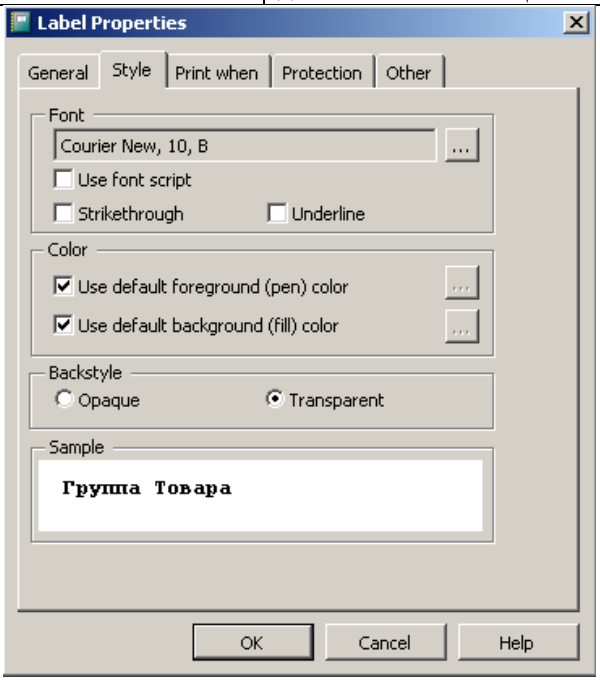

Рис. 8. Вкладка **Style** диалогового окна **Label Properties**

Вкладка **Print when** окна свойств используется для задания условий печати объектов отчета (рис. 9). С помощью параметров этой вкладки вы можете удалить из отчета пустые строки, определить условия печати значений полей при переходе на следующею страницу или при изменении выражения группы и т. п.

Для подавления печати повторяющихся значений объекта используются значение **No** переключателя **Print repeated values** (Печатать повторяющиеся значения). При установленном значении **Yes** печатаются все значения объекта.

Область **Also print** (Печатать) содержит флажки, описание которых приводится в табл. 8.

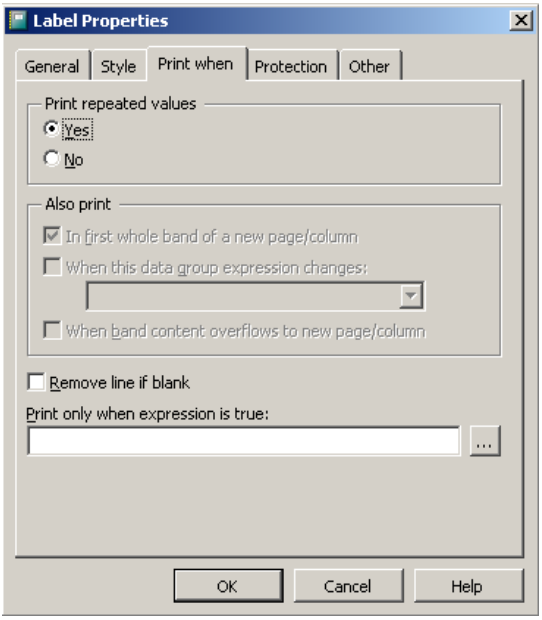

Рис. 9. Вкладка **Print when** диалогового окна **Label Properties**

Таблица 8.

Флажки области **Also print**

Флажок	Назначение
In first whole band of new page/column (На первой целой полосе новой страницы/колонки)	Объект печатается в первой полосе новой страницы или колонки
When this data group expression changes (При изменении значения выражения группы)	Объект печатается при изменении значения выражения группы, выбранной в списке групп
When band content to new page/column (При переходе на новую страницу/колонну)	Объект печатается при переходе полосы Detail на новую страницу/колонку

При установке флажка **Remove line if blank** (Удалять пустые строки) пустые строки удаляются из отчета.

В поле **Print only when expression is true** (Печатать, если истинно), используя построитель выражения, можно задать выражение, вычисляемое перед печатью данного объекта. Если значение выражения ложно, то значение объекта печататься не будет.

Вкладка **Protection** (Защита) (рис. 10) устанавливает защиту от изменения объектов отчета с помощью конструктора отчетов. Она содержит флажки, описание которых приведено в табл. 9.

Таблица 9.

Флажки вкладки **Protection**

Флажок	Назначение
Object cannot be moved or resized (Объект не может быть	Запрещает перемещение и изменение размера объекта

изменен или перемещен в размерах)	
Object cannot be edited (Объект не может редактироваться)	Запрещает редактирование объекта
Object cannot be deleted (Объект не может быть удален)	Запрещает удаление объекта
Object cannot be selected (Объект не может быть выбран)	Объект не может быть выбран для изменения
Object is not visible in Designer (Объект не виден в конструкторе)	Объект не видим в конструкторе

Для того чтобы свойства по защите вступили в силу, необходимо открыть отчет с ключом *Protection.* Например, MODIFY REPORT MyReport PROTECTED.

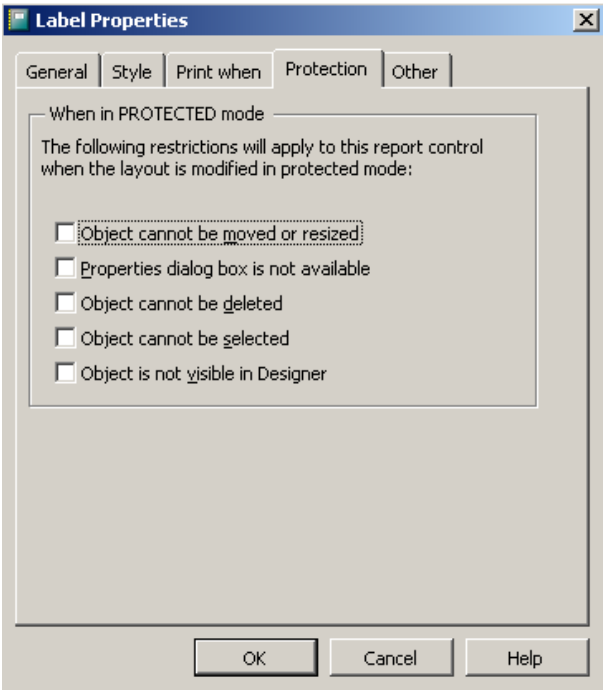

Рис. 10. Вкладка **Protection** диалогового окна **Label Properties**

Вкладка Other (Другие) (рис. 11) диалогового окна свойств объекта отчета содержит дополнительные возможности для объектов отчета.

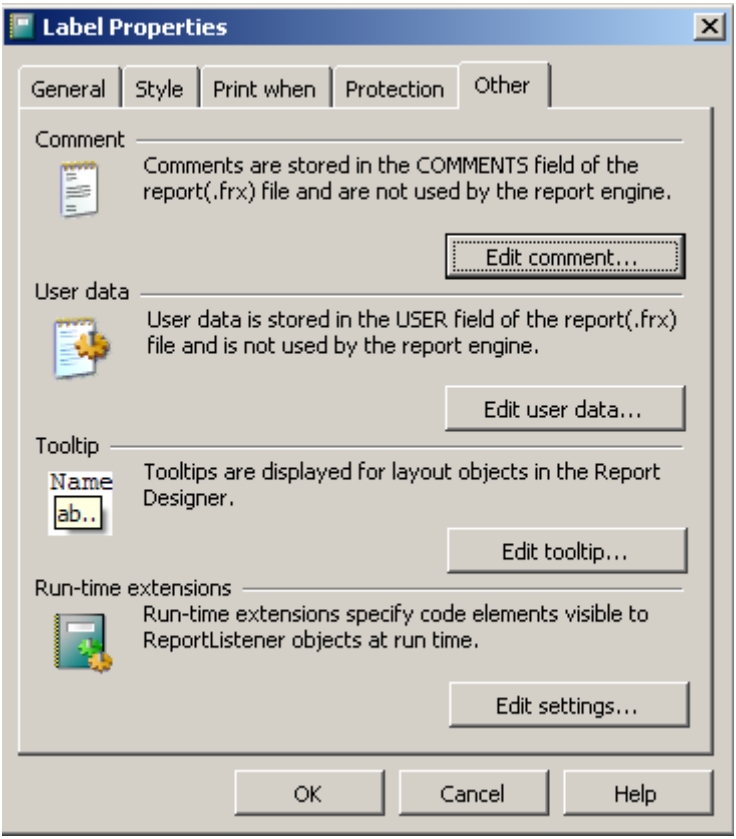

Рис. 11. Вкладка **Other** диалогового окна **Label Properties**

Вкладка содержит кнопки, назначение которых описано в табл. 10.

Таблица 10.

Кнопки вкладки **Other**

Кнопки	Назначение
Edit comment (Редактирование комментария)	Открывает диалоговое окно комментария к объекту отчета
Edit user data (Редактирование пользовательских данных)	Открывает диалоговое окно, в котором вы можете добавлять или редактировать пользовательские данные
Edit tooltip (Редактирование всплывающих подсказок)	Открывает диалоговое окно для ввода всплывающей подсказки, которая будет появляться при выборе объекта в конструкторе отчетов
Edit settings (Редактирование настроек)	Определяются настройки, которые могут использоваться внешним кодом в процессе формирования отчета

Размещение объектов в отчете

Текстовая информация. Размещаемый в отчете текст является объектом, который можно выделять, перемещать, сохранять во временном буфере Windows, копировать из буфера или удалять.

Для ввода или редактирования текста в отчете выполните следующие действия:

1. Нажмите кнопку **Label** (Метка) на панели инструментов Report Controls (Элементы управления отчета).

2. Щелкните мышью в том месте окна конструктора отчета, где необходимо разместить или исправить текст.
3. Внесите необходимые добавления или изменения. Текст может состоять из нескольких строк. Для переноса части текста на новую строку используйте клавишу «**Enter**».
4. Нажмите кнопку **Select Objects** (Выбор объектов) на панели инструментов **Report Controls** (Элементы управления отчета).
5. Используя диалоговое окно свойств объекта Label Properties (Свойства метки), настройте его параметры.

Для размещения в отчете текста можно изменять параметры шрифта и цвет, используя вкладку **Style** (Стиль) окна свойств объекта **Label Properties** (Свойства метки) (рис.12) или команду **Font** (Шрифт) меню **Format**.

Которое открывается при выполнении команды Шрифт, используемый в отчете текста по умолчанию, можно задавать в поле **Default font** (Шрифт по умолчанию) на вкладке **Reports** диалогового окна **Options** (Параметры), **Options** (Параметры) из меню Tools (Сервис).

Размещение полей

Для размещения в отчете поля, которое может быть полем таблицы или вычисляемым полем, выполните следующие действия:

1. Нажмите кнопку **Field** (Поле) на панели инструментов **Report Controls.**
2. Щелкните мышью в месте предполагаемого размещения поля в окне конструктора отчета.
3. Открывается диалоговое окно **Field Properties (**Свойства полей**)** (рис.12). Используя объекты интерфейса вкладок, настройте параметры поля. Окно свойств позволяет:
    - определить источник данных или задать выражение, результат вычисления которого будет выводиться в данное поле;
    - задать формат отображения данных в поле;
    - указать условие печати;
    - установить положение поля в отчете.
4. Завершив настройку параметров, закройте окно свойств, нажав кнопку **ОК**.

Для размещения поля в отчете можно также использовать метод перенести-и-оставить. Откройте окно **Data Environment** (Среда окружения) отчета, установите курсор на название размещаемого поля, нажмите кнопку мыши и, удерживая ее в нажатом состоянии, переместите курсор в место предполагаемого

размещения в отчете, после чего отпустите кнопку мыши. Если в конструкторе таблицы для поля задано свойство **Caption** (Надпись), поле разместится в отчете вместе с надписью.

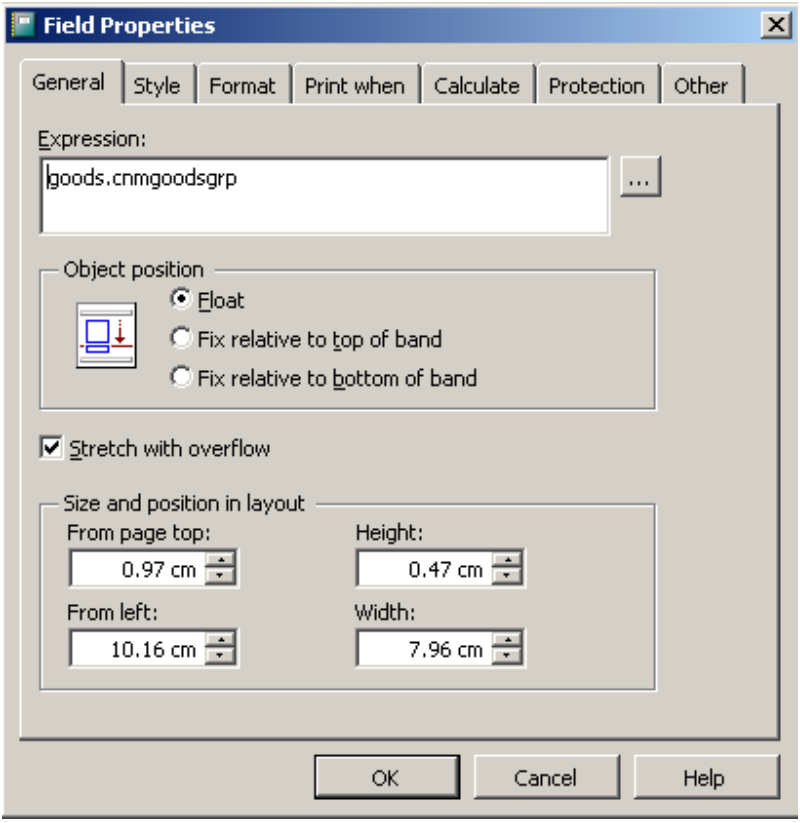

Рис. 12. Диалоговое окно **Field Properties**

Формирование выражения поля

Для определения выражения, результат выражения которого будет в размещенное в отчете поле, предназначено поле

Expression (Выражение) вкладка General (Общие) окна свойств Field Properties.

Выражение в это поле может быть введено вручную или задано с помощью построителя, открываемого нажатием расположенной справа от поля кнопки. При этом на экране откроется диалоговое окно **Expression Builder** (Построитель выражения) (рис.13), где в поле **Expression** (Выражение) необходимо задать требуемое выражение.

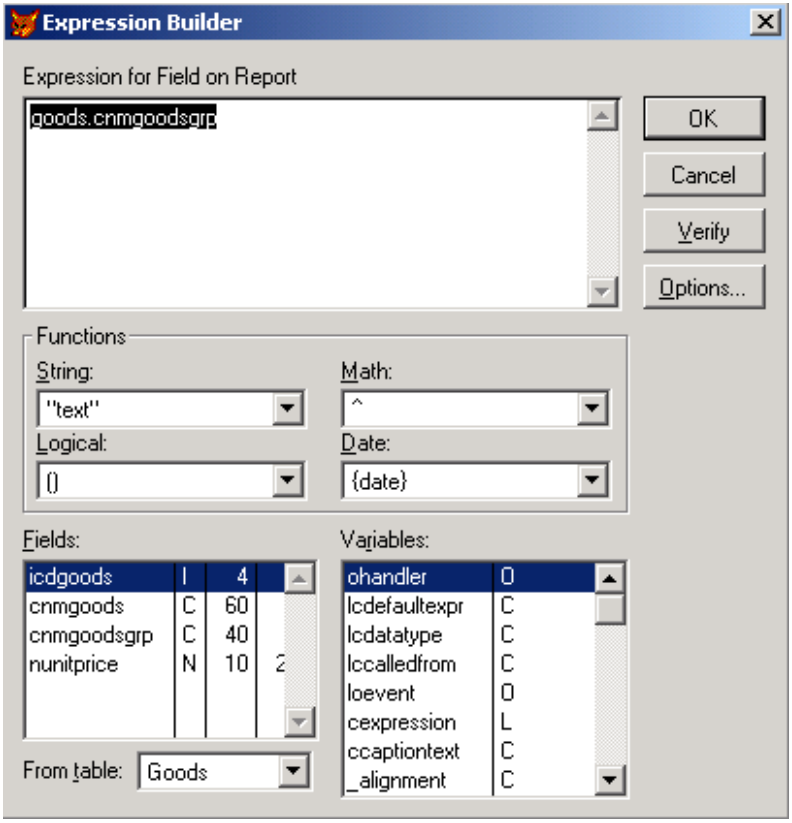

Рис. 13. Определение вычисляемого поля

Список Fields (Поля) диалогового окна **Expression Builder** содержит поля помещенных в окружение отчета таблиц и список **Variables** (Переменные) – системные переменные Visual FoxPro. В области **Functions** (Функция) размещены строковые, логические, математические функции, а также функции даты и времени. Используя значения из этих списков, сформируйте необходимое выражение для создаваемого поля.

Для формирования выражения нет необходимости вводить информацию в поле **Expression** вручную. Достаточно выбрать требуемое значение из любого списка и щелчком мыши перенести его в это поле.

При создании вычисляемых полей сформируйте выражение и проверти его правильность с помощью кнопки **Verify** (Проверить). Например, вы можете отображать в отчете вместо трех полей cLastName, cFirstName, cSecondName одно вычисляемое поле, которое содержит фамилию и инициалы покупателя. Выражение для этого поля имеет следующий вид:

ALLTRIM(Customer.cLastName)+”
”+SUBSTR(Customer.cFirstName,1,1)
+”.”+SUBSTR(Customer.cSecondName,1,1)+”.”

Например, вы можете отобразить в отчете вместо двух полей, содержащих город и адрес клиента, одно вычисляемое поле, которое содержит полный адрес. Выражение для этого поля имеет следующий вид: **ALLTRIM (Customer.cCity) +”, ”+ALLTRIM(Customer.cAddress)**

Завершив формирования выражения, нажмите кнопку **ОК** для закрытия для закрытия диалогового окна **Expression Builder.**

Задание формата данных

Для задания формата отображения поля при печати предназначена вкладка **Format** диалогового окна **Field Properties** (рис.14). Она позволяет:

- преобразовать весь символьный вывод и прописные буквы;
- выравнивать информацию;
- показывать в числах пробелы и десятичные запятые;
- переводить дату из американского формата в европейский и многое другое.

Перечень параметров настройки поля определяется выбором соответствующей опции для формирования полей символьного, числового типов, а также полей дат. Список этих параметров представлен в табл. 11 – 13.

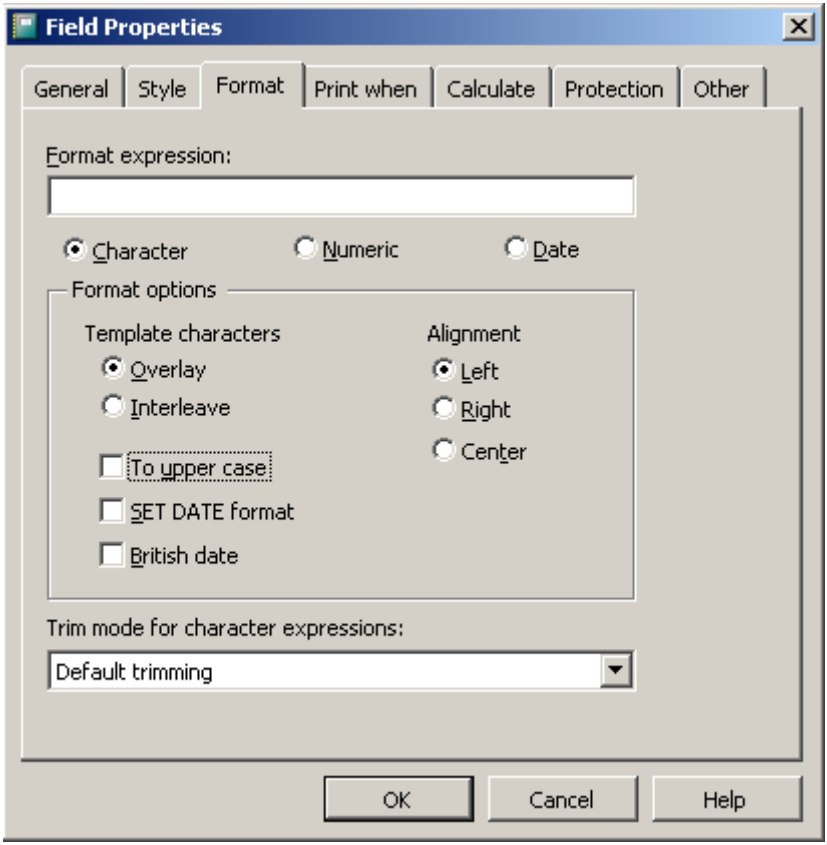

Рис. 14. Вкладка **Format** диалогового окна **Field Properties**

Таблица 11.

Параметры настройки символьных полей

Параметры	Назначение
Переключатель **Template characters** (Символьный шаблон)	Определяет, как символы шаблона влияют на взаимодействие символа шаблона с символами текстового поля. Переключатель содержит две опции: **Overlay** (Заменить) и **Interleave** (Вставлять).

	Например, если текстовое поле содержит значение АБВ1234, то при установке переключателя **Overlay** будет отображаться АБВ – 234 (символ 1 будет заменен дефисом): а при выборе **Interleave –** АБВ – 1234
Переключатель **Justification** (Выравнивание)	Переключатель содержит опции **Left, Right, Center**, указывающие на выравнивание данных по левому, правому краю поля и по центру
To upper case (Верхний регистр)	Символы преобразуются в прописные
Ignore input mask (Игнорировать маску ввода)	Отображает на экране, но не сохраняет в заданной формате в таблице
SET DATE format (В виде даты)	Данные отображаются в виде даты в формате установленном командой **SET DATE**
British date (Европейская дата)	Данные отображаются в виде даты в европейском формате

Таблица 12.

Параметры настройки числовых полей

Параметры	**Назначение**
Left justify (Сдвинуть в лево)	Число выравнивается по левому полю
Blank if zero (Пусто, если нуль)	Нуль не печатается
(Negative) (Отрицательное)	Отрицательные числа заключаются в круглые скобки
SET DATE format (В виде даты)	Данные отображаются в виде даты в формате, установленном командой SET DATE

British date (Европейская дата)	Данные отображаются в виде даты в европейском формате
CR if positive (CR, если положительно)	Если число положительное, поле его становиться CR (кредит)
DB if negative (DB, если отрицательно)	Если число отрицательное, поле его становиться DB (дебит)
Leading zeros (Ведущие нули)	Печатаются все ведущие нули
Currency (Денежная единица)	Данные отображаются в формате денежной единицы, который задается командой SET CURRENCY
Scientific (Экспоненциально)	Отображает число в экспоненциальном формате

Таблица 13.

Параметры настройки полей типа дата

Параметры	Назначения
SET DATE format (В виде даты)	Дата отображается в формате, установленном командой SET DATE
British date (Европейская дата)	Дата отображается в европейском формате
Blank if empty (Пусто, если дата не задана)	Ничего не отображается, если дата пустая

Размещение итогового поля

В колонтитулах, полосах группы, в итоговой части отчета, а также в полосе **Detail** можно размещать поля, содержащие статистические значения полей отчета.

Итоговые поля, размещаемые в полосе **Detail,** предназначены для вывода значений нарастающих итогов.

Для определения поля в качестве итогового выполните следующие действия:

1. Откройте диалоговое окно **Field Properties**.
2. Используя поле **Expression** вкладки **General**, задайте выражение для поля.
3. Перейти на вкладку **Calculate** (Вычислить).
4. Используя список **Calculate type,** выберите математическую операцию над значением выражения, определенного для данного поля (табл. 14).
5. Используя список **Reset based on** (Сброс значений на основе), укажите момент обнуления итогового поля.
6. Установите требуемые параметры, закройте окно свойств, нажав кнопку **ОК.**

Таблица 14.

Опции списка **Calculate type**

Опция	**Назначение**
None (Нет)	Над полем не производится вычислений
Count (Сосчитать)	Вычисляется количество значений поля (сами значения поля не используются)
Sum (Сумма)	Вычисляется итоговая сумма значений поля
Average (Сред. Знач.)	Вычисляется среднее арифметическое значений поля
Lowest (Наименьшее)	Отображается наименьшее значение поля
Highest (Наибольшее)	Отображается наибольшее значение поля

Standard deviation (Квадратный корень из дисперсии)	Возвращается квадратный корень из дисперсии
Variance (Отклонение от среднего)	Возвращается статистическая величина отклонения отдельных значений поля от среднего в группе

Создание простого отчета

Последовательность действий по созданию отчета для таблицы *Customer*, содержащей список клиентов:

1. Откройте проект *Sales.*
2. Откройте базу данных проекта.
3. Выберите группу **Reports** вкладки **Documents** и нажмите кнопку **New**.
4. В открывшемся диалоговом окне **New Report** выберите опцию **New Report**.
5. Для задания среды окружения отчета откройте диалоговое окно **Date Environment**, выбрав команду **Date Environment** в меню **View** или из контекстного меню.
6. Для добавления таблицы в окружение отчета в меню **Date Environment** выберите команду **Add**.
7. В открывшемся диалоговом окне **Add Table or View**, выберите таблицу *Customer* и нажмите кнопку **Add.** Закройте окно **Add Table or View** с помощью кнопки Close. В окне **Date Environment** будет отображена выбранная таблица.

8. Откройте окно свойств таблицы. Для этого установите курсор на ее названии, нажмите правую кнопку мыши и выберите из контекстного меню команду **Properties**.
9. Выделите свойство *Order* (Порядок). Для упорядочивания данных в отчете по кодам клиентов в поле изменения свойств нажмите кнопку раскрытия списка и из списка индексов таблицы выберите *icdCustomer*.
10. Закройте окно **Date Environment.**
11. Для размещения полей таблицы в отчете воспользуйтесь командой **Quick Report** в меню **Report**.
12. Выберите вариант размещения полей по столбцам и нажмите кнопку **Fields**.
13. В диалогом окне **Field Picker** (Выбор поля) выберите поля, используя кнопку **Move** (Перенести). Нажмите кнопку **ОК**.
14. Возвратившись в диалоговое окно **Quick Report,** нажмите кнопку ОК для завершения процедуры размещения полей в отчете.
15. Используя кнопку **Label** (Метка) панели инструментов **Report Controls** (Элементы управления отчета)**,** в случае необходимости скорректируйте заголовки полей.

16. Для того чтобы придать отчету законченный вид, добавьте область заголовка отчета, выбрав в меню **Report** команду **Optional Bands** (Дополнительные полосы).
17. На вкладке **Optional Bands** (Дополнительные полосы) диалогового окна **Report Properties** (Свойства отчета) установите флажок **Report has title band** (Отчет содержит титульную полосу) и нажмите **ОК**. в отчете появится полоса **Title**. Разместите в ней текст заголовка отчета с помощью кнопки **Label** панели инструментов **Report Controls** (Элементы управления отчета).
18. Просмотрите внешний вид отчета, воспользовавшись командой контекстного меню **Preview** (Просмотр) (рис. 15).
19. Сохраните отчет.

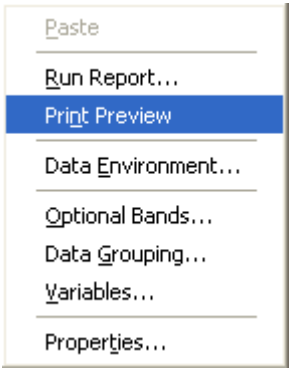

Рис. 15 Контекстное меню с пунктом предпросмотра отчета

Создание отчета в свободной форме

Рассмотрим создание отчета в свободной форме, содержащего рассылку новых прайс–листов потенциальным клиентам.

1. Откройте проект Sales. Откройте базу данных.
2. Перейдите на вкладку **Documents**, выберите группу **Reports** и нажмите кнопку **New**. В открывшемся окне диалога **New Report** выберите опцию **New Report**. На экране откроется окно конструктора отчетов.
3. Откройте среду окружения отчета **Data Environment**, выполнив команду **View | Environment** или выбрав опцию **Data Environment** всплывающего меню. Для добавления таблицы в окружение выполните команду **Data Environment | Add**. В открывшемся окне диалога **Add Table or View** выберите таблицу *Customer* и нажмите кнопку **OK**. В окне диалога **Data Environment** появилась выбранная таблица. Закройте окно **Data Environment**.
4. Разместите в поле *Detail* поля с наименованием предприятия, индекса и страны.
5. Добавьте в полосе *Detail* отчета вычисляемое поле для отображения имени и отчества представителя покупателя, содержащее следующее выражение поля:

ALLTRIM(Customer.cFirstName)+' '+
ALLTRIM(Customer.cSecondName)

Затем добавьте вычисляемое поле для вывода адреса, которое содержит следующее выражение поля:

ALLTRIM(Customer.cCity)+' '+
ALLTRIM(Customer.cAddres)

6. Разместите текст и поля, как показано на рис. 16.
7. Просмотрите внешний вид отчета с помощью команды всплывающего меню **Preview**. Ваш экран будет иметь вид, представленный на рис. 17.
8. Сохраните отчет.

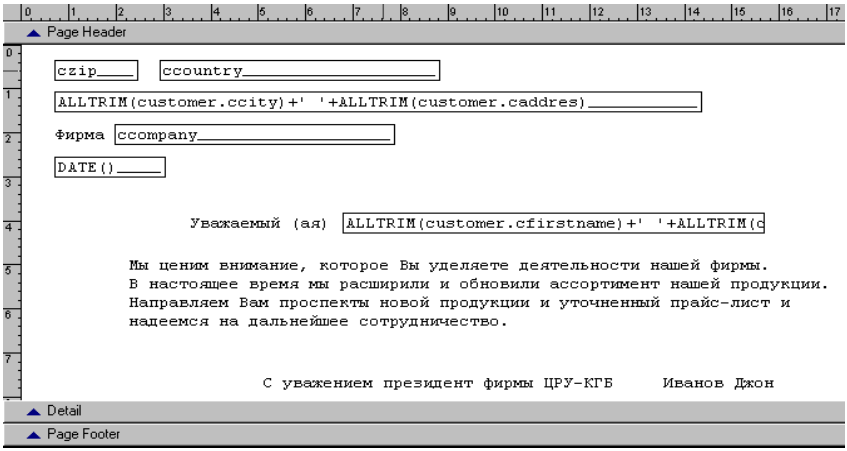

Рис. 16. Отчет в свободной форме

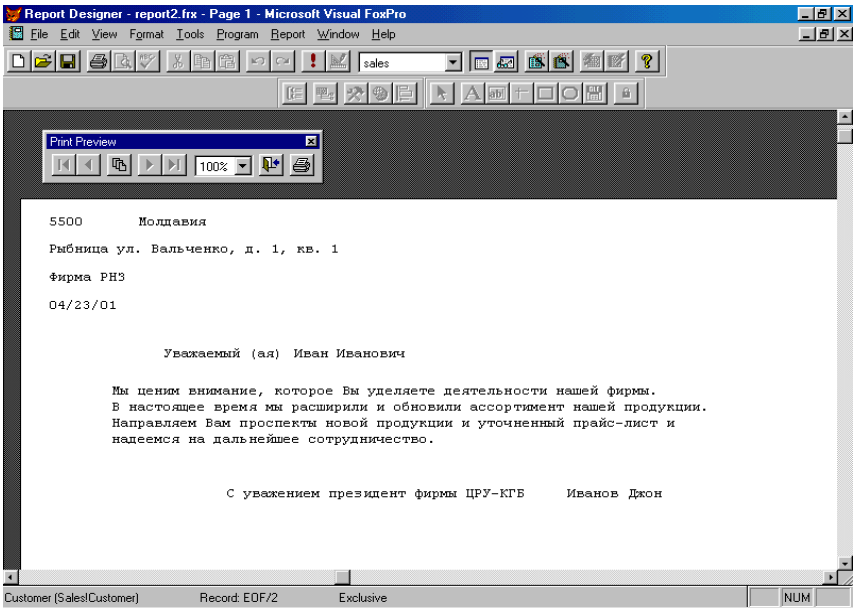

Рис. 17. Просмотр отчета в свободной форме

Примечания

1. Для просмотра отчета можно использовать команду REPORT FORM <имя отчета>, например: *REPORT FORM report1*

   Отчет будет выдан на текущее устройство вывода. По умолчанию – это принтер.

   Добавление служебного слова PREVIEW после имени отчета служит для открытия окна предпросмотра, которое выдается на экран, например:

   REPORT FORM report1 PREVIEW

Дальнейшую печать можно осуществить используя соответствующий управляющий элемент на панели **Print Preview**.

2. Если пути поиска файлов в проекте не заданы, рекомендуется указывать полный путь к файлу отчета. Задать путь поиска можно в разделе **Tools | Options | File Locations | Search Path.**
3. Для более гибкой конфигурации экранного отображения информации в отчетах можно использовать вкладку PRINT WHEN в свойствах элементов отчета (надписях, полях вывода и т.д.) (рис. 9). Значение поля при этом будет выдаваться на экран только в случае, если указанное условия в окне ввода *Print only when expression is true*, будет принимать логическое значение ИСТИНА. Например, задание условия ***nunitprice>100*** приведет к тому, что на экране будут отображаться значения цен товаров только больше 100, в остальных случаях отобразится пустое место. На отображение других элементов указанное условие не повлияет.
4. Для организации изменения цвета отображения значения поля на экране в зависимости от какого–либо условия можно использовать следующую технологическую последовательность:
   – создать два одинаковых поля вывода;
   – указать для них желаемые цвета отображения;

- указать непересекающиеся условия печати для каждого поля (непересекающиеся означает, что не должно быть значений, для которых оба логических выражений дают истину. Например: выражения A>=10 и A<10 являются непересекающимися, а A>=10 и A=<10 – пересекаются в точке 10);
- расположить оба поля в одном и том же месте на экране конструктора отчетов. Поля должны визуально накладываться одно на другое.

В этом случае, при предварительном просмотре, на экране будут отображаться надписи разных цветов, в зависимости от значения указанного логического выражения.

5. Экранный вид отчета рассчитывается системой в момент его запуска. Для отображения данных, которые введены в таблицу после запуска отчета, его необходимо пересчитать, т.е. закрыть и открыть отчет заново.
6. В отличие от экранных форм, таблицы, указанные в окружении данных отчета, автоматически системой не открываются. Если указанная таблица не открыта, система выдаст предложение выбрать источник данных для отчета.

Задание

1. Создайте отчет для справочника товара с помощью команды **Quick Report**. Отчет должен содержать только названия товаров и их цены.

2. Создайте отчет по заказчикам при помощи конструктора. В отчете должны отражаться данные по фамилиям и инициалам заказчиков, а так же названиям фирм. Названия фирм должны отображаться синим цветом.
3. В виде отчета получите данные о номерах заказов и кодах товаров, которые были заказаны.
4. Добавьте новый товар при помощи экранной формы в заказ с наименьшим номером. Отметьте, в каком месте появился введенный заказ.
5. Установите сортировку в источнике данных предыдущего отчета по номеру заказа. Отметьте разницу между текущим отчетом и отчетом, полученным в п.4.
6. Создайте отчет для справочника товаров. Все цены меньше 1000 должны отображаться синим цветом, а от 1000 и выше – красным.
7. Добавьте в предыдущий отчет титульный лист, на котором должна отображаться текущая системная дата.

Лабораторная работа № 8

Тема: Создание табличного отчета

Цель: Научиться создавать табличные отчеты с помощью конструктора отчетов

Для улучшения внешнего вида отчета и повышения читабельности можно использовать линии и прямоугольники. Для проведения вертикальной или горизонтальной линии в отчете выполните следующие действия:

1. Нажмите кнопку **Line** на панели **Report Controls** конструктора отчетов.
2. Установите указатель в начальную точку линии.
3. Нажмите кнопку мыши и, удерживая ее, проведите линию необходимой длинны.
4. Используя опции команды **Pen** (Перо) из меню **Format**, установите атрибуты линии (толщину т тип).
5. Для задания цвета размещенной в отчете линии щелкните мышью на понравившемся цвете цветовой палитры панели **Color Palette** (Цветовая палитра).

Для размещения в отчете прямоугольника и прямоугольника со скругленными углами используются кнопки **Rectangle** (Прямоугольники) и **Rounded Rectangle**

(Прямоугольник со скругленными углами) на панели инструментов **Report Controls** конструктора отчетов. Нажмите необходимую кнопку на панели инструментов, установите указатель в какой-либо угол прямоугольника и перемещайте курсор в противоположный угол до получения прямоугольника нужного размера. Затем установите необходимые атрибуты прямоугольника, используя для этого следующие опции команды **Pen** (Перо) из меню **Format** (табл. 1).

Таблица 1.

Опции команды **Pen** из меню **Format**

Опция	**Атрибут объектов**
Hairline	Контур шириной в один пиксел
1 Point (1 пункт)	Контур шириной в один пункт
2 Point (2 пункт)	Контур шириной в два пункта
4Point (4 пункт)	Контур шириной в четыре пункта
6 Point (6 пункт)	Контур шириной в шесть пунктов
None ()	Выделенный объект не имеет контура
Dotted	Пунктирный контур
Dashed	Контур в виде коротких штрихов
Dash-dot	Контур в виде чередующихся штрихов и точек
Dash-dot-dot	Контур в виде чередующихся штрихов и пар точек

Для настройки параметров линий и прямоугольников можно использовать не только команды основного меню, но и окно свойств объекта. С помощью списка **Style** (Стиль) вкладки **Style** (рис.1) задается тип контура. Список Weight (Толщина) позволяет указать толщину контура, а список Curvature

(Кривизна) – закругленность. Вид получаемого объекта можно просмотреть в области Sample (Образец).

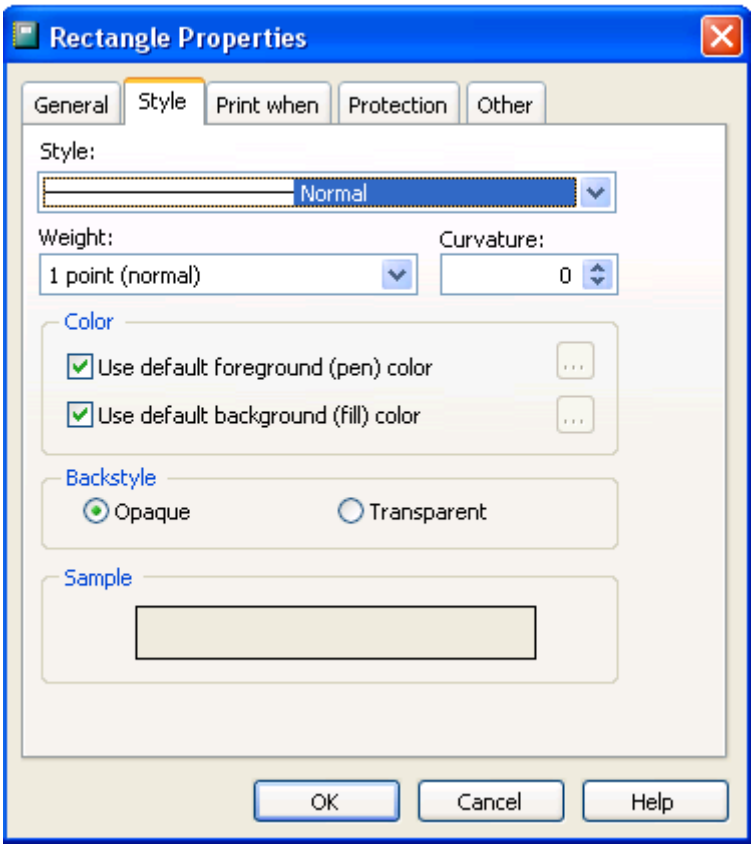

Рис. 1. Вкладка **Style** диалогового окна настройки прямоугольника

Панель инструментов Color Palette

Для изменения цвета объекта можно использовать область **Color** (Цвет) окна свойств объекта или панель инструментов **Color Palette** (Цветная палитра). Для изменения в отчете цвета объекта выполните одно из следующих действий:

1. Разместите на экране панель инструментов **Color Palette** (Цветная палитра) выполнив одно из следующих действий:

- нажав кнопку **Color Palette Toolbar** (Панель инструментов **Цветная палитра**) на панели инструментов **Report Designer** (Конструктор отчета);
- выбрав в меню **View** (Вид) команду **Color Palette Toolbar** (Панель инструментов **Цветная палитра**).

Панель инструментов **Color Palette** (Цветная палитра) содержит шестнадцать кнопок с заданными цветами и три дополнительные кнопки (табл. 2).

Таблица 2.

Кнопки панели инструментов **Color Palette**

Кнопка	**Назначение**
	Задает цвет объекта
	Задает цвет фона
	Открывает диалоговое окно **Цвет** для задания цветов, отсутствующих на панели

2. Выберите в отчете объект, цвет которого вы хотите изменить.

3. Щелкните мышью на панели инструментов **Color Palette** кнопку **Foreground Color** (Цвет объекта) или **Background Color** (Цвет фона) в зависимости от того, что вы хотите поменять, – цвет объекта или его фона.

4. Щелкните мышью на панели инструментов любой понравившейся цвет.

При создании отчетов можно использовать более широкую цветовую гамму. Для этого необходимо воспользоваться кнопкой **Other Colors** (Другие цвета) панели инструментов **Color Palette.** Возможно создание пользовательского цвета в области **Дополнительные цвета**.

Размещение в отчете рисунка

В отчеты можно включать растровые рисунки, которые улучшают внешний вид отчета. В письмах, рассылаемых клиентам, можно, например, поместить фирменный знак или эмблему фирмы.

Для размещения в отчете рисунка используется кнопка **Picture/OLE Bound Control** (Изображение/ OLE-объекта) панели инструментов **Report Controls** (Элементы управления отчета) конструктора отчетов. Нажмите данную кнопку, а затем установите курсор в один из углов области, в которой должен находиться рисунок, и переместите курсор в противоположный угол до образования рамки необходимого размера. При этом открывается диалоговое окно **Picture/OLE Bound Properties** (Свойство изображения/ OLE-объекта) (рис. 2), в котором вы определяете источник данных рисунка и его параметры. Источником данных может быть файл, поле таблицы типа **General**, выражение или переменная. Для указания типа источника данных используются опции области **Control source**

type (Тип источника данных), а источника данных – поле **Control source** (Источник данных).

При размещении в отчете графического изображения на вкладке **General** (Общие) диалогового окна **Picture/OLE Bound Properties** (Свойство изображения/ OLE-объекта) установите опцию **Image file name** (Имя файла изображения). Затем нажмите кнопку выбора файла, расположенную с правой стороны поля **Control source** (Источник данных).

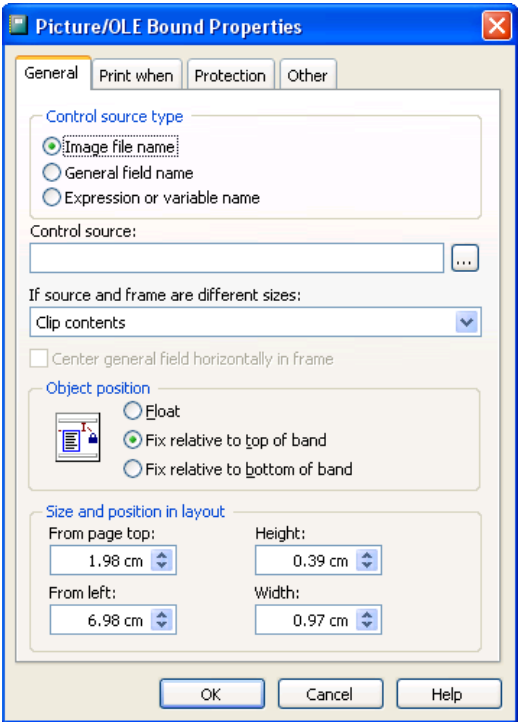

Рис.2. Диалоговое окно **Picture/OLE Bound Properties**

В открывшемся диалоговом окне **Open Picture** (Открыть изображение) выберите требуемый графический файл. Если вы знаете полное имя файла, то можете ввести его непосредственно в

поле ввода **Control source** (Источник данных) без использования диалогового окна **Open Picture** (Открыть изображение).

В том случае, если вы хотите печатать в отчете изображения, размещенные в поле таблицы (в Visual EoxPro для хранения графических изображений используются поля типа General), на вкладке **General** диалогового окна **Picture/OLE Bound Properties** установите опцию **General field name** (Имя поля типа General). Затем нажмите кнопку выбора, расположенную с правой стороны поле **Control source** (Источник данных) и в открывшемся диалоговом окне **Expression Builder** (Построитель выражения) выберите необходимое поле таблицы.

Если размер выделенной для размещения изображения области и размер самого изображения не совпадают, воспользуйтесь опциями списка **If source and frame are different sizes** (Если разный размер) (табл.3).

Таблица 3.

Опции списка **If source and frame are different sizes**

Опции	Режим отображения
Clip contents (Обрезать содержимое)	Объект фиксируется в левой верхней части рамки, сохраняя первоначальный размер
Scale contents, Retain shape (Масштабировать, сохраняя форму)	Объект полностью заполняет отведенное ему поле, сохраняя относительные пропорции растрового изображения
Scale contents, fill the frame (Масштабировать, заполняя рамку)	Объект полностью заполняет отведенное ему поле, в случае необходимости подвергаясь вертикальному или горизонтальному искажению

Создание табличного отчета

Создадим в конструкторе отчетов табличный отчет, содержащий список заказов, использующий таблицы Ordsalem, Ordsaled, Customer, Goods.

1. Откройте проект Sales.
2. Откройте новое окно в конструкторе отчетов.
3. Откройте окно **Data Environment** и добавьте в него поочередно таблицы Ordsalem, Ordsaled, Customer, Goods. Связи, установленные между таблицами при создании базы данных, переносятся вместе с таблицами. Упорядочив данные, закройте окно **Data Environment**. При создании связей между таблицами для данного отчета необходимо учитывать следующее:
    a. таблица Ordsalem является родительской по отношению к таблицам Ordsaled, Customer;
    b. связь между таблицами Ordsalem и Customer осуществляется по коду клиента;
    c. связь между таблицами Ordsalem и Ordsaled осуществляется по коду заказа;
    d. связь между таблицами Goods и Ordsaled осуществляется по коду товара;
    e. данные в таблице Ordsalem должны быть упорядочены по коду заказа, в таблице Customer – по коду клиента, в таблице Ordsaled – по коду заказа, в таблице Goods – по коду товара.
4. Разместив в отчете следующие поля:

- *cCompany* с наименованием компаний из таблицы Customer;
- *dOrderDate*, содержащее дату проданного товара из таблицы Ordsalem;
- *nOrderQuant* с количеством проданного товара из таблицы Ordsaled;
- *nUnitprice* с ценой товара из таблицы Goods.

5. Создайте вычисляемое поле, которое содержит выражение:
   Ordsaled.nOrderQuant* Goods.nUnitPrice
6. Добавьте в полосе **Page Header** заголовки для созданных полей.
7. Добавьте заголовок и итоговую часть отчета, выполнив команду **Report | Optional Bands (**Дополнительные полосы**)** и установив на одноименной вкладке открывшегося диалога окна **Report Properties** флажок **Report has title band** (Отчет содержит титульную полосу).
8. В заголовке отчета разместите текст «Список заказов», а также рисунок, представляющий собой эмблему фирмы. В итоговой части отчета создайте итоговое поле, которое вычислит стоимость всех заказов, просуммировав стоимость проданного товара.
9. Отчет в окне конструктора отчетов будет иметь вид, представленный на рис. 3. Просмотрите внешний вид отчета в окне предварительного просмотра

(File | Print Preview) (рис. 4).

10. Сохраните отчет.

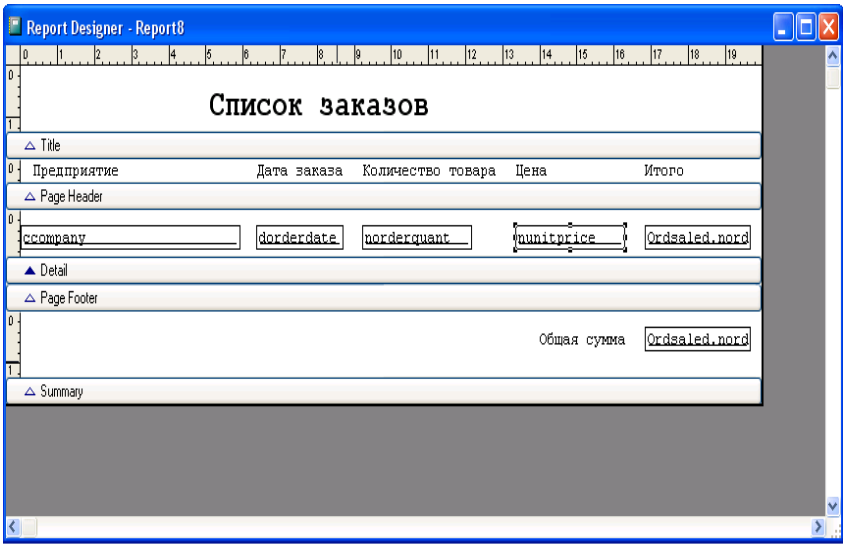

Рис. 3. Создание отчета в окне конструктора отчетов

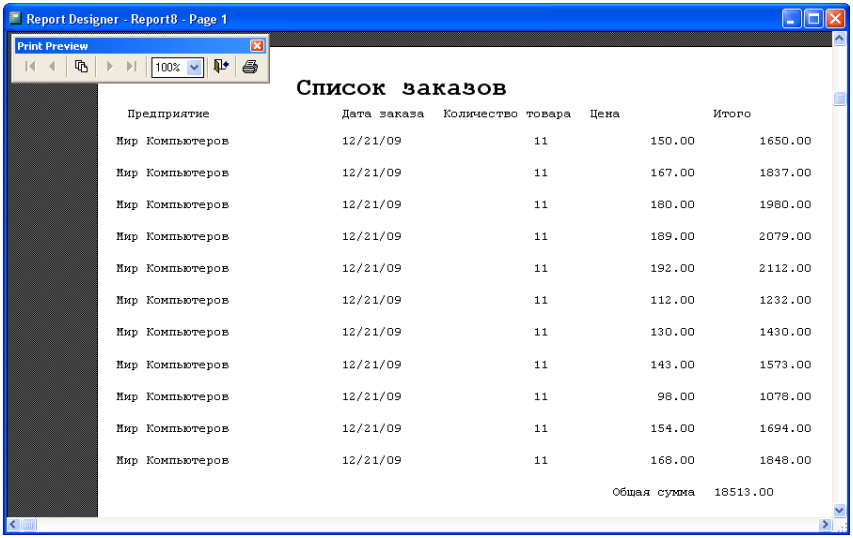

Рис. 4. Просмотр табличного отчета

Обратите внимание, что на самом деле, данные, полученные в отчете, не соответствуют действительности. Это происходит из–за того, что таблица Ordsalem является родительской одновременно по отношению к двум таблицам сразу (Customer и Ordsaled), а Ordsaled – дочерней так же для двух таблиц (Goods и Ordsalem). СУБД не имея возможности определить по какому источнику следует ориентироваться, использует текущий, что никак не соответствует логике связей в базе данных. Соответственно, вычисленные для отчета данные будут ложными.

Рассмотрим простейший способ разрешения этой проблемы. Он заключается в том, что построителю отчета следует указать только один источник данных. Объединение данных из нескольких таблиц в один источник возможно при использовании представлений данных.

Создадим в конструкторе отчетов табличный отчет, содержащий список продаж товаров, использующий данные из таблицы Ordsalem, Ordsaled, Goods, Customer.

1. Откройте проект Sales.
2. Создайте представление данных View1, указав в качестве источников для него таблицы Ordsalem и Customer. Убедитесь в правильности установленной связи!
3. Выберите из таблицы Customer поля icdCustomer и cCompany, а из таблицы Ordsalem – icdOrder и dOrderDate.

4. Сохраните представление данных View1. (В качестве дополнительной меры самоконтроля рекомендуется созданное представление просмотреть и убедиться в том, что данные считаются правильно).
5. Создайте представление данных View2, указав в качестве источников для него созданное представление View1 и таблицу Ordsaled. Убедитесь в правильности установленной связи! (Не забывайте, что представление данных View1 является главным по отношению к таблице Ordsaled).
6. Выберите все поля, относящиеся к View1 и добавьте поля icdGoods и nOrderQuant из таблицы Ordsaled.
7. Сохраните представление данных View2.
8. Создайте представление данных View3, указав в качестве источников для него созданное представление View2 и таблицу Goods.
9. Выберите все поля, относящиеся к View2, и добавьте поля cnmGoods и nUnitPrice из таблицы Goods.
10. Сохраните представление данных View3.
11. Откройте новое окно в конструкторе отчетов.
12. Откройте окно **Data Environment** и добавьте в него созданное представление данных View3. Закройте окно **Data Environment**.
13. Добавьте в полосе «Detail» отчета поля cCompany, dOrderDate, nOrderQuant, nUnitPrice из View3.

14. Создайте вычисляемое поле, которое содержит выражение:

    View3.nOrderQuant*View3.nUnitPrice

15. Добавьте в полосе **Page Header** заголовки для созданных полей.
16. Добавьте заголовок и итоговую часть отчета, выполнив команду **Report | Title/Summary**. В заголовке отчета разместите текст «Список продаж», а также рисунок, представляющий собой эмблему фирмы. В итоговой части отчета создайте итоговое поле, которое суммирует значение выражения:

    View3.nOrderQuant*View3.nUnitPrice

Полученный макет отчета ничем не будет отличаться от макета представленного на рис. 3.

17. Просмотрите внешний вид отчета с помощью команды **File | Print Preview** (рис. 5).
18. Сохраните отчет.

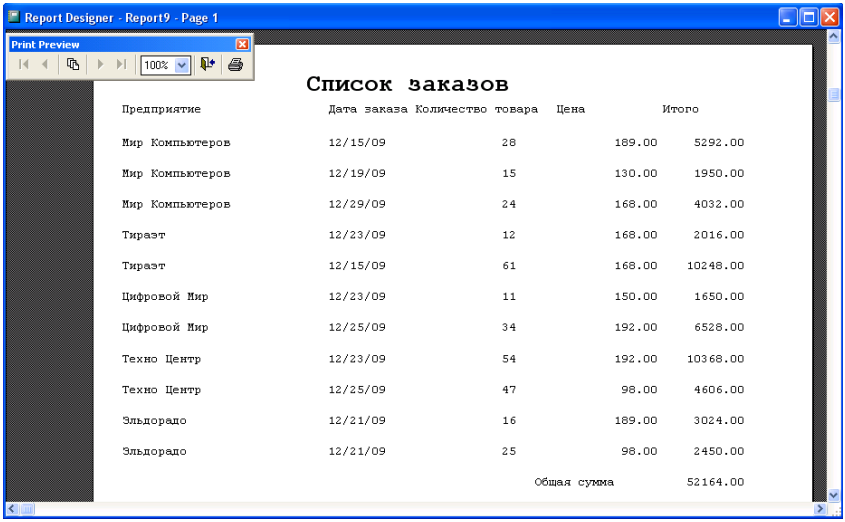

Рис. 5. Просмотр табличного отчета, созданного на базе представления данных

Примечание: В случаях, когда для получение отчета необходимы данные более чем из двух связанных таблиц, рекомендуется использовать технологию пошагового построения источника данных для отчета, т.е. *Таблица1+Таблица2=Представление1, Представление1+Таблица3=Представление2* и т.д. Результирующее представление следует использовать в качестве источника данных для отчета. Этот подход предпочтительнее еще тем, что разработчик имеет возможность отследить, на каком шаге вычислений система начинает считать неправильно.

Группировка данных в отчете

Индексирование таблицы или сортировка позволяют распечатывать записи в требуемом порядке. Однако этих средств далеко не всегда оказывается достаточно. На практике достаточно часто возникает необходимость объединять записи в группы. Для этой цели используется команда меню **Report | Data Grouping** или команда **Grouping** всплывающего меню, которые позволяют создавать до 20 уровней вложенности групп и выполнять над ними следующие операции:

- напечатать текст, идентифицирующие конкретные группы
- напечатать каждую группу с новой страницы
- осуществлять сброс нумерации страниц при печати групп с новой страницы

В результате выполнения команды **Report | Data Grouping** открывается окно диалога «Data Grouping» (рис. 6), которое содержит весь список созданных групп. Можно редактировать или удалять группы, также добавлять новые.

Следует помнить, что при указании группировки по нескольким полям, система вычислит данные для отчета последовательно, в указанном порядке. Группировка будет проведена сначала по значениям первого указанного поля, затем ***для каждой полученной группы*** будет произведена группировка по значениям второго указанного поля и т.д.

При указании группировки по нескольким полям следует руководствоваться следующим правилом: *Вначале указывается поле, значение которого наиболее масштабно для отчета*. Например, если необходимо получить детальную информацию по каждому заказу, произведенному покупателями, то группировка должна осуществляться сначала по покупателям, а затем по заказам.

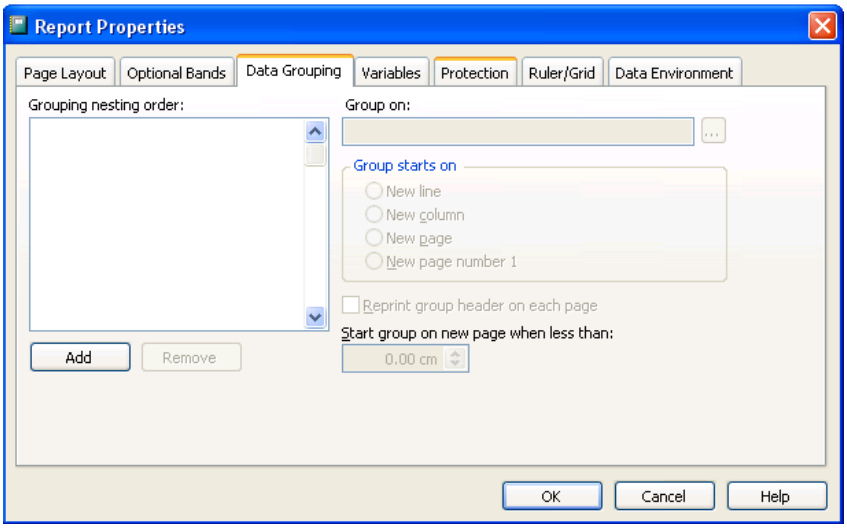

Рис. 6. Вкладка «Data Grouping» диалогового окна **Report Properties**

На вкладке **Data Grouping** (Группировка данных)диалогового окна **Report Properties** можно установить следующие параметры группировки данных (табл. 4).

Таблица 4.

Параметры группировки вкладки **Data Grouping**

Флажок	**Назначение**
New line (Новая строка)	При каждом изменении группы происходит формирование новой строки
New column (Новая колонка)	При каждом изменении группы происходит формирование новой колонки
New page (Новая страница)	Начинает новую страницу при каждом изменении группы
New page number to 1 (Новая страница с номером 1)	Начинает новую страницу при каждом изменении группы, а нумерацию страницы – с 1
Reprint group header on each page (Верхняя полоса группы для каждой страницы)	Размещает верхнюю полосу группы после верхнего колонтитула страницы, если группа занимает несколько страниц
Start group on new page when less than (Печатать группу с новой страницы, если)	Если под заголовком группы остается расстояние меньше указанного в данном поле, то информация группы будет перенесена на новую страницу

После установки необходимых флажков нажмите кнопку **OK**. Для удаления полосы группы воспользуйтесь кнопкой **Remove**.

С помощью кнопки ADD вкладки **Data Grouping** можно добавить в отчет новую группу. При ее нажатии открывается окно построителя выражения, позволяющее сформировать выражение для группировки.

Добавление группировки данных

Модифицируем отчет, созданный в предыдущем примере (отчет на базе представления данных). Добавьте в него группировку по покупателям и разместите в полосах группы наименование покупателя и итоговую сумму по каждому покупателю.

1. Откройте созданный отчет.
2. Создайте группу по полю icdCustomer. Для этого выполните команду **Report | Data Grouping**. В открывшемся окне диалога «Data Grouping» нажмите кнопку вызова построителя выражения группировки данных. В окне диалога «Expression Builder» сформируйте выражение группировки, выбрав из списка всех полей открытых таблиц поле icdCustomer.
3. Расширьте полосу «Group Header» и перенесите в нее поле с наименованием предприятия, а также заголовок поля.
4. Расширьте полосу «Group Footer» и создайте в нем итоговое поле, которое суммирует значение выражения: View3.nOrderQuant*View3.nUnitPrice
5. Просмотрите внешний вид отчета с помощью команды **File | Print Preview**.
6. Отметьте разницу между полученным и предыдущим результатом.
7. Сохраните отчет (рис. 7).

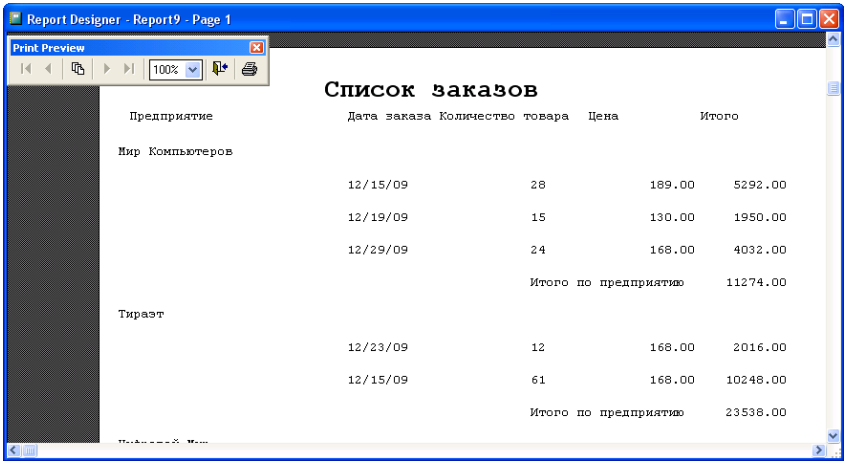

Рис. 7. Просмотр отчета с группировкой данных

Использование в отчете переменных

В отчете можно использовать переменные из программы, доступные в момент вызова отчета, а также переменные, определенные в конструкторе отчетов и используемые для хранения результатов вычислений, выполняемых во время печати отчета. Переменные отчета определяются на вкладке **Variable** (Переменные) диалогового окна **Report Properties (**Параметры отчета) (рис.7), для открытия которого используется команда **Variables** из меню **Report.** В открывшемся окне можно добавлять в отчет новые переменные и удалять существующие.

В случае, если в системе уже определена глобальная переменная с именем, совпадающим с именем переменной созданной в отчете, ее значение будет использовано при построении отчета. Это позволяет разработчикам передавать

необходимые данные построителю отчета при его вызове из исполняемого приложения.

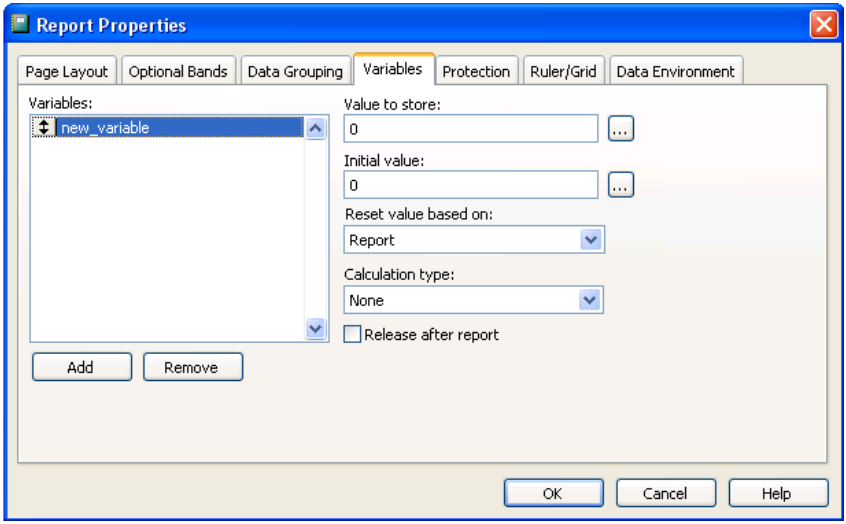

Рис. 7.Вкладка **Variables** диалогового окна **Report Properties**

Вкладка **Variables** диалогового окна **Report Properties** содержит поля, описанные в табл. 5.

Таблица 5.

Поле вкладке **Variables**

Наименование поля	**Назначение**
Variables (Переменные)	Содержит наименование переменной, которое может содержать только буквы, цифры и символ подчеркивания и не может начинаться с цифры
Value to store (Хранимое значение)	Значение переменной
Initial value (Начальное значение)	Начальное значение переменной
Reset value based on (Сброс на	Список содержит три значения,

основе)	указывающие момент сброса переменной в начальное значение: в конце отчета, в конце страницы или в конце группы
Calculation type (Тип вычисления)	Опции, расположенные в списке, позволяют задать выражения, выполняемые над переменной вычисления (табл. 6)
Release after report (Освободить после отчета)	При установке флажка после завершения печати отчета переменная очищается из памяти

Для формирования значений переменных, задаваемых в полях **Value to store** и **Initial value,** можно использовать диалоговое окно Expression Builder, открываемое при нажатии расположенных с правой стороны поля кнопок.

Таблица 6.

Назначение опций списка **Calculation type**

Опции	Назначение
Non (Нет)	Над переменной вычисления не производятся
Count (Количество)	Вычисляется количество появлений переменной в группе, на странице, в колонке или отчете (значение переменной не используется)
Sum (Сумма)	Вычисляется итоговая сумма значений переменной
Average (Среднее)	Вычисляется среднее арифметическое значений переменной в группе, на странице, в колонке или отчете

Lowest (Минимальное)	Отображается наименьшее значение переменной в группе, на странице, в колонке или отчете
Highest (Максимальное)	Отображается наибольшее значение переменной в группе, на странице, в колонке или отчете
Standard deviation (Стандартное отклонение)	Возвращается квадратный корень из дисперсии значений переменной в группе, на странице, в колонке или отчете
Variance (Дисперсия)	Возвращается статистическая величина отклонения отдельных значений переменной от среднего в группе, на странице, в колонке или отчете

При использовании переменных в отчете необходимо иметь в виду следующее:

- переменные в отчете могут использоваться в качестве полей или в выражениях, определяющих поля;
- при запуске отчета переменной присваивается начальное значение, а затем в процессе формирования отчета ее значение изменяется в соответствии с выбранным выражением. При заданных для переменной условиях она принимает свое начальное значение;
- для определения начального или вычисляемого значения переменной могут использоваться другие переменные, но значения этих переменных должны

вычисляться до момента их использования переменной.

Разметка страницы отчета

Для разметки страницы отчета используется вкладка **Page Layout** (Разметка страницы) диалогового окна **Report Properties** (Свойства отчета) (рис.8), которое открывается при выборе меню **File | Page Setup.** Объекты интерфейса этой вкладки позволяют определить:

- количество колонок в отчете;
- порядок вывода записей;
- ширину левого поля отчета;
- ширину колонок и расстояние между ними;
- единицу измерения координат отчета и выбор режима печати.

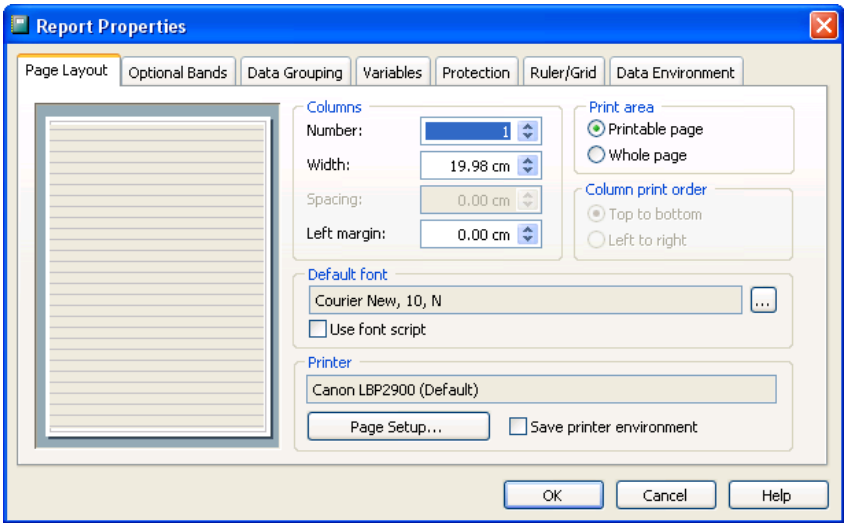

Рис.8. Вкладка **Page Layout** диалогового окна **Report Properties**

В области «Columns» устанавливаются значения, определяющие количество колонок на странице и их размеры:

Поле	Назначение
Number (Число)	Определяет число колонок на странице
Width (Ширина)	Определяет ширину колонок в сантиметрах или дюймах
Spacing (Расстояние)	Определяет расстояние между колонками
Left margin	Задает ширину левого поля отчета

Переключатель **Print area** содержит следующие опции:

Опция	Назначение
Printable page (Печатаемая страница)	Устанавливает режим печати с полями страницы, определяемыми в соответствии с требованиями текущего драйвера печати
Whole page (Страница в целом)	Устанавливается режим печати с минимальными полями

Для задания порядка вывода записей в многоколоночных отчетах используется переключатель **Column print order** (Порядок печати колонок).

Примечание: Для системы нет никакой разницы в использовании для построения отчета таблицы, представления данных или результата выборки оператора SELECT–SQL. В случае использования таблицы или представления, имена полей указываются явно и процедура предпросмотра не вызывает ошибки. При использовании результата оператора выборки, следует выполнить следующую последовательность действий:

- сохранить результат в курсор;
- в конструкторе отчетов явно указать имя курсора и поле из него в качестве источника данных для выводимых значений. (Символом разделителем служит точка *Cursor1.Field1*);
- запустить отчет.

обычно этот способ используют в программном коде кнопок управления на экранных формах, при этом сам код может выглядеть следующим образом:

```
*производим выборку в курсор temp1
SELECT * FROM Goods INTO CURSOR temp1
*запускаем отчет на предпросмотр
*в полях отчета ОБЯЗАТЕЛЬНО должны быть явные
ссылки на выбранные поля в курсоре
*в нашем случае это temp1.icdgoods или
temp1.cnmgoods
REPORT FORM Report1 PREVIEW
*закрываем временный курсор temp1, чтоб не
засорять память
SELECT temp1
use
```

Задания

1. Создайте отчет для справочника товара. Данные должны быть сгруппированы по группе товара.
2. Создайте отчет по произведенным заказам, относящимся к каждому покупателю.
3. Создайте отчет по каждому заказу с выводом общей суммы заказа и перечислением всех товаров, входящих в заказ.
4. Отметьте разницу между использованием для просмотра данных форм и отчетов.
5. Создайте отчет по клиентам и приобретенным ими товарами, проведите группировку в отчете по клиентам и разместив в отчете итоговую сумму приобретенных товаров.
6. Подготовьте данные для печати адресов, наклеиваемых на конверты для отправки клиентам. Образец отчета приведен ниже. В качестве источника следует использовать данные из таблицы Customer.

Куда: 63009 Россия Новосибирск, ул. Ильича, д. 34 АО Новости Кому: Смельченко П. И.	Куда: 173024 Россия Москва, ул. Свободы, д. 16 Банк Программ Кому: Ивлеву М.И.

Обратный адрес: 103617 Россия Москва, ул. Солнечная, 12 ООО «МИР ПК»	Обратный адрес: 103617 Россия Москва, ул. Солнечная, 12 ООО «МИР ПК»
Куда: 63009 Россия Новосибирск, ул. Ильича, д. 34 АО Новости Кому: Смельченко П. И.	Куда: 173024 Россия Москва, ул. Свободы, д. 16 Банк Программ Кому: Ивлеву М.И.
Обратный адрес: 103617 Россия Москва, ул. Солнечная, 12 ООО «МИР ПК»	Обратный адрес: 103617 Россия Москва, ул. Солнечная, 12 ООО «МИР ПК»

Лабораторная работа № 9

Тема: Система меню приложения

Цель: Научиться создавать меню приложения

Хорошо написанное законченное приложение может использоваться пользователем любой квалификации. Обычно законченное приложение имеет свое собственное меню, которое заменяет основное меню Visual FoxPro и содержит команды, предназначенные для выполнения конкретных задач.

Перед тем, как приступить непосредственно к созданию приложения, можно создать все требуемые объекты (базу данных, входящие в нее таблицы, экранные формы, отчеты, запросы). Затем отдельные объекты могут быть объединены посредством меню.

Планирование приложения

Первым этапом создания любого приложения является определение требований, предъявляемых к законченному приложению. Прежде всего, необходимо определить состав информации, которая будет содержаться в создаваемой базе данных.

После определения состава данных, которые будут храниться в базе данных, создаются все входящие в нее таблицы. На этом этапе определяются отношения между таблицами,

структура каждой таблицы и совпадающие поля для связывания таблиц.

Состав информации невозможно определить без учета конкретных задач, для решения которых предназначается создаваемое приложение. Поэтому одновременно с составом информации необходимо определить те средства, которые получит в свое распоряжение пользователь при работе с приложением.

Приложение должно содержать эффективную справочную систему, которая не требует специального руководства. В среде Windows предпочтительнее всего создать справочную систему в принятом в Windows стандарте, так как в этом случае пользователь сможет получить любую информацию в знакомой ему среде.

После того как спроектированы таблицы, входящие в состав базы данных, и определены основные требования, предъявляемые к приложению, можно приступить к созданию структуры меню приложения или, как часто его называют, дерева меню. Дерево меню можно организовать на основе функций, выполняемых приложением (например **Продажа, Поступление товара**) или же на основе таблиц, используемых в приложении (**Покупатель, Товар**). Оба варианта широко используются на практике, выбор одного из них или использование альтернативного принципа построения дерева меню определяется конкретными требованиями.

Прежде чем начинать описывать структуру меню с помощью конструктора меню, следует нарисовать его эскиз на бумаге, посоветоваться с пользователями приложения по поводу его структуры. Пока меню нарисовано только на бумаге, можно легко его изменить, добавить новые пункты и удалить лишнее. По тщательно разработанному эскизу меню достаточно легко определить набор экранных форм и отчетов, которые потом следует создать, и подключить к конкретному пункту меню.

Создание строки меню.

Строкой меню называется горизонтальное меню, расположенное в верхней части экрана. Примером строки меню является основное меню Visual FoxPro. Можно создать собственную строку меню, которая будет замещать основное меню Visual FoxPro или добавляться к нему. Для создания строки меню необходимо выполнить следующие действия:

1. Открыть окно конструктора меню.
2. Описать вид меню, текст, пункты меню и его атрибуты.
3. Определить действия, выполняемые при выборе пункта меню.
4. Сгенерировать меню, используя команду **Generate** (Генерация) из меню **Menu**. При этом создается программа, которую вы в результате и запускаете на выполнение.

На каждом шаге создания меню приложения можно просматривать его с помощью кнопки **Preview** конструктора.

Окно конструктора меню

Для создания нового меню в конструкторе меню можно воспользоваться следующими способами:

- Выполнить команду меню **File | New**. В открывшемся окне диалога «New» установить опцию **Menu** и нажать кнопку **New File**.
- В окне проекта перейти на вкладку «**Other**» и выбрать группу «**Menus**». Затем нажать кнопку **New** окна проекта.
- Находясь в группе «Menus» окна проекта, нажать кнопку **New** на стандартной панели инструментов Visual FoxPro. В открывшемся окне диалога «**New**» установить опцию **Menu** и нажать кнопку **New File**.

Независимо от используемого способа на экране откроется окно диалога «**New Menu**», предлагающее два варианта создаваемого меню (рис. 1):

- Menu – создание меню в виде строки;
- Shortcut – создание всплывающего меню, в котором основные пункты меню расположены вертикально.

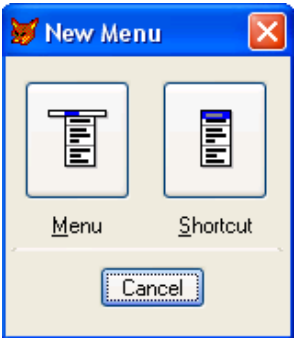

Рис. 1 Диалоговое окно **New Menu**

При выборе любого из вариантов появляется окно конструктора меню, а в основном меню Visual FoxPro добавляется новый пункт Menu (рис. 2). Создание каждого из видов меню совершенно идентично, хотя каждый вид описывается разными командами Visual FoxPro. Различия эти можно заметить, только просмотрев тексты файлов .MPR, в которых хранятся тексты команд определения меню. Эти файлы следует редактировать только в крайнем случае, при необходимости дополнительной настройки создаваемого меню.

Описание меню состоит из двух частей. В первой части описывается вид меню, текст и типы пунктом меню, а также их экранные атрибуты. Во второй части описываются действия, выполняемые при выборе пунктов меню. Если описать только первую часть, то меню будет отображаться на экране, но при выборе пунктов меню никакие действия выполняться не будут.

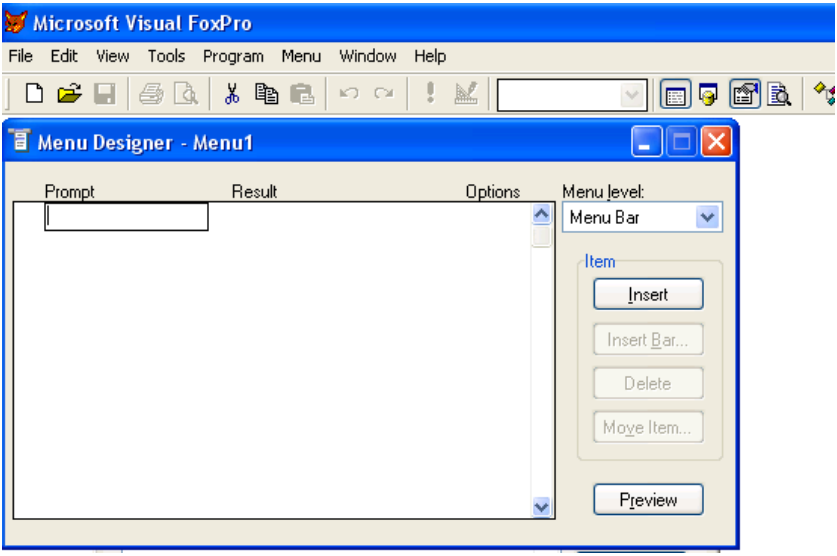

Рис. 2 Окно конструктора меню

В поле **Prompt** (Приглашает) можно ввести наименования пунктов меню. Раскрывающийся список **Result** (Результат) используется для указания типа пункта меню. Кнопка **Options** (Опции) открывает диалоговое окно **Prompt Options** (Опции элемента меню), в котором можно определить дополнительные параметры данного элемента меню («горячие» клавиши, сообщение, отображаемое в строке состояния при выборе пункта меню, и.т.д.). В списке Menu Level (Уровень меню) указывается уровень текущего меню.

Определение текстов пунктов строки меню

Для определения текстов пунктов строки меню необходимо нажать кнопку **Insert** и напечатать текст в поле **Prompt**. Для

определения типа пункта меню следует выбрать требуемый элемент из списка **Result.**

Возможен выбор следующих типов пунктов меню:

Тип	**Описание**
Command (Команда)	При выборе пункта меню будет выполняться связанная с ним команда
Pad Name (Наименование строки)	При выборе пункта меню никаких действий выполняться не будет. Как правило, используется в качестве дополнительного пояснения к меню.
Submenu (Подменю)	При выборе пункта меню раскрывается связанное с данным пунктом ниспадающее меню.
Procedure (Процедура)	При выборе пункта меню вызывается процедура, определенная для данного пункта меню.

Ввод команды, выполняемой при выборе пункта меню, осуществляется в поле, которое расположено правее описания типа пункта меню. При выборе типов **Procedure** и **Submenu** в окне конструктора правее описания типа пункта меню появляется кнопка **Create**, при нажатии на которую можно перейти в окно создания процедуры или начать создание ниспадающего меню для выбранного пункта меню.

Сохранение, генерация и запуск меню.

Для сохранения меню необходимо выполнить команду меню **File | Save as**. В открывшемся окне диалога «Save as» в поле **Папка:** требуется указать каталог, в который сохраняется файл, а в поле **Save Menu** ввести имя сохраняемого меню. Для сохранения служит кнопка **Сохранить.**

Кнопка **Preview** окна конструктора меню позволяет просмотреть внешний вид создаваемого меню, но не позволяет его активизировать. Для использования меню в приложениях его необходимо предварительно сгенерировать. Для этого служит команда **Menu | Generate**. При этом открывается окно диалога «Generate Menu» (рис. 3). В поле ввода окна диалога необходимо ввести наименование файла, который будет создан в результате генерации. Используя кнопку **…** можно выбрать путь и имя файла в стандартном окне Windows.

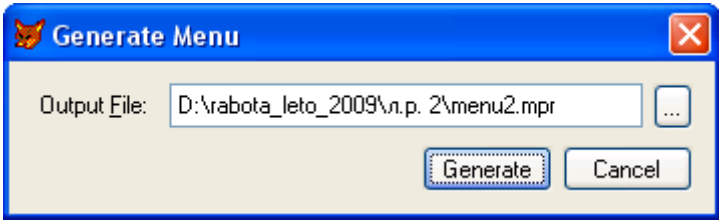

Рис. 3 Окно диалога «Generate Menu»

Для запуска генерации описания служит кнопка **Generate.**

После завершения генерации можно запустить программу меню. Для этого необходимо использовать команду меню **Program | Do**. В открывшемся окне диалога «Do» в случае необходимости следует выбрать каталог, содержащий файл меню, выбрать файл с расширением .MPR и нажать кнопку **Do**. На экране появится созданное меню, которое заменит основное меню Visual FoxPro. Для восстановления на экране системного меню в командном окне следует ввести команду **SET SYSMENU TO DEFAULT**.

Определение оперативных клавиш для пунктов меню.

Выбор пункта меню осуществляется при помощи клавиш–стрелок или мыши. Дополнительно можно определить символ из имени пункта меню, активизирующий этот пункт при нажатии комбинации *Alt* и указанного символа, называемого *оперативной клавишей*.

Для предоставления пользователю возможности ускоренного выбора пункта меню необходимо включить в его имя перед активизирующим символом «\<». Обычно в качестве оперативной клавиши используется первый символ имени пункта, однако допустимо использование любого символа из текста пункта. Указанные символы отображаются в строке меню с подчеркиванием (рис. 4).

«Горячие» клавиши для пунктов меню назначаются Visual FoxPro по умолчанию. Для их создания используются первые буквы элементов строки меню. Если два элемента вашего меню начинаются с одинаковой буквы, то обоим элементам строки меню в качестве «горячей» клавиши назначается одинаковый символ. В этом случае необходимо переопределить «горячею» клавишу для одного из элементов строки меню.

Для облегчения назначения «горячих» клавиш можно использовать искусственный прием: перед именем пунктов меню разместить цифры и использовать их в качестве «горячих» клавиш.

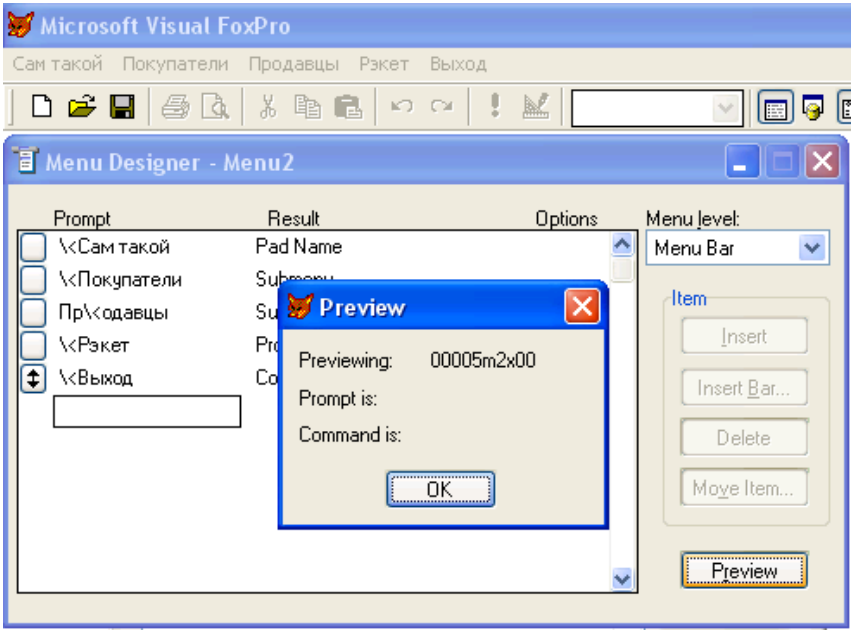

Рис. 4 Просмотр строки меню с назначенными оперативными клавишами

Задание параметров пункта меню в диалоговом окне Prompt Options

При перемещении по пунктам меню в конструкторе меню в столбце **Option** появляется кнопка **Options**. При нажатии на данную кнопку на экране открывается окно диалога «**Prompt Options**» для данного пункта меню (рис. 5).

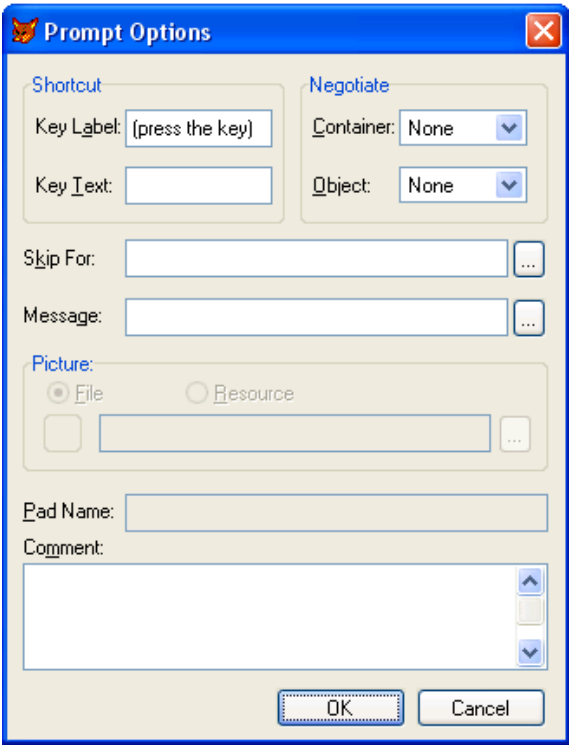

Рис. 5 Диалоговое окно **Prompt Options**

В окне диалога содержится область «**Shortcut**», предназначенная для задания клавиш ускоренного действия. В данном окне диалога расположены также следующие поля ввода:

Поле ввода	Назначение
Skip For (Пропустить для)	Блокирует команду меню
Message (Сообщение)	Задает сообщение для пункта меню в строке состояния
Pad Name (Имя пункта меню)	Задает имя пункта меню
Comment (Комментарий)	Задает комментарий к пункту меню

Опция **Negotiate** (Соглашение) предназначена для определения места расположения заголовков меню при редактировании по месту OLE объектов.

Область **Negotiate** (Соглашение) содержит два раскрывающихся списка, имеющих следующее назначение:

- **Container** (Контейнер) – определяет расположение меню редактирования по месту OLE объектов;
- **Object** (Объект) – задает расположение меню при выполнении приложения типа Active Document в Web-браузера.

Область **Picture** (Изображение) позволяет разместить слева от текста пункта меню графическое изображение. Расположенные в ней объекты интерфейса доступны при определении подпунктов меню.

Определение клавиш ускоренного действия

Оперативные клавиши позволяют использовать для ускоренного выбора пункта меню только символы текста пункта меню. В Visual FoxPro существует возможность определения для каждого пункта меню *клавиши ускоренного действия*. В качестве такой клавиши могут использоваться символы, функциональные клавиши, а также комбинации клавиш. При нажатии на эту клавишу активизируется связанный с ней пункт меню. Отличие между оперативными клавишами и клавишами ускоренного действия состоит в том, что с помощью клавиш ускоренного

действия команда выбирается без предварительного отображения меню команд.

Для назначения клавиши ускоренного действия для пункта меню следует нажать кнопку **Options** данного пункта меню. На экране откроется окно диалога «**Prompt Options**» (рис. 5). В области «**Shortcut**» поле **Key Label** этого окна диалога содержит подсказку «**press the key**». Находясь в этом поле необходимо нажать клавишу или комбинацию клавиш, используемую в качестве клавиши ускоренного действия, и в поле появится метка, соответствующая нажатию клавиши. Последовательность действий по определению клавиши быстрого вызова пункта меню:

1. В конструкторе меню установите курсор на пункте меню **Справка**.
2. Нажмите кнопку **Options.**
3. В открывшемся диалоговом окне **Prompt Options** (Опции элемента меню) в области **Shortcut** (Всплывающее меню) поле **Key Label** (Метка) содержит подсказку **press the key** (Нажмите клавишу). Установите курсор в поле **Key Label** и нажмите клавишу или комбинацию клавиш, используемую в качестве клавиши быстрого вызова, и в поле появиться метка, соответствующая нажатым клавишам. В поле ввода **Key Text** можно ввести краткое пояснение к определяемой клавише. Для пункта меню **Справка** нажмите клавишу «**F1**».

В поле **Key Text** (Пояснение) по умолчанию будет введено **F1**. оставьте это значение.

4. Нажмите кнопку **ОК** для закрытия диалогового окна.

Определение сообщения для пункта меню

Для каждого пункта меню можно определить сообщение, которое будет отображаться в строке состояния при установке курсора на этот пункт меню. Для определения сообщения используется поле ввода **Message** окна диалога «**Prompt Options**». Допускается использовать в качестве сообщения не только строку текста, но и произвольное символьное выражение. Для этого используется кнопка вызова с правой стороны поля и в окне диалога «**Expression Builder**» создается выражение.

Определим сообщение для пункта меню **Справка**. Для этого выполните следующие действия:

1. Нажмите кнопку **Options** для пункта меню **Справка.**
2. В открывшемся диалоговом окне **Prompt Options** нажмите кнопку вызова построителя выражения для поля **Message.**
3. В поле **Message** построителя выражения введите строку текста «**Вызов справочной системы**». Введенную строку текста не забудьте поместить в кавычки.

4. Нажмите кнопку ОК для закрытия диалогового окна **Expression Builder.** Вы окажитесь в окне **Prompt Options.** Введенное вами выражение размещено в поле **Message.**

Блокирование команд меню

Иногда возникает необходимость сделать какой-либо пункт меню недоступным для пользователя. В этих случаях можно определить для пункта меню условие блокировки. Для определения условия блокировки используется поле ввода **Skip For** в окне диалога «**Prompt Options**». Для этого следует нажать кнопку вызова конструктора выражений с правой стороны поля и в окне диалога «**Expression Builder**» создать выражение.

При определении условия блокировки можно вводить любое допустимое логическое выражение. Например, можно использовать условие блокировки для ограничения доступа определенных пользователей к отдельным пунктам меню.

Visual FoxPro делает пункт меню недоступным, когда значение выражения, заданное в условии блокировки, является истинным. Допускается использовать в условиях блокировки глобальные переменные, определенные при запуске приложения.

В режиме просмотра вы не увидите, что пункт меню заблокирован. Чтобы убедиться в правильно созданной блокировке, вам необходимо сгенерировать созданное в конструкторе меню и полученную в результате программу запустить на выполнение.

Определение имени пункта меню

По умолчанию при генерации программы Visual FoxPro в качестве имен пунктов меню создает уникальные имена. Для повышения читабельности программы рекомендуется определить эти имена явным образом, используя поле ввода **Pad Name** окна диалога «**Prompt Options**». Для предотвращения некорректной работы программы в дальнейшем следует задавать имена пунктов меню английскими буквами, но максимально приближенными к выполняемому действию.

Определение действий для пунктов меню

При выборе пункта меню выполняется действие, определенное для этого пункта меню. Результат выполнения действия определяется типом пункта меню:

Тип пункта меню	**Действие**
Submenu	Раскрывает связанное с данным пунктом меню ниспадающее подменю
Procedure	Выполняется процедура, которая определяется в конструкторе меню
Command	Выполняется команда, указанная в поле, расположенном правее типа пункта меню

Для определения команды необходимо ввести в поле, расположенное правее типа пункта меню, команду Visual FoxPro, которая будет выполняться при выборе пункта меню. Наиболее часто эти команды содержат вызов экранных форм, отчетов, а также пользовательских процедур (рис. 6).

Если при выборе пункта меню выполняется некоторая последовательность команд, можно выбрать тип пункта меню **Procedure** и определить эти команды непосредственно в

конструкторе меню. При выборе типа **Procedure** справа от списка типов появляется кнопка **Create**. При нажатии на эту кнопку на экране открывается окно редактирования процедуры. В нем нужно определить фрагмент кода, связанный с элементом строки меню. Не следует использовать команду PROCEDURE во фрагменте кода, поскольку Visual FoxPro генерирует эту команду автоматически. Во время генерации меню Visual FoxPro создает уникальное имя для каждой процедуры и включает это имя и местонахождение процедуры в код, связанный с соответствующим элементом строки меню. После определения текста процедуры, наименование кнопки **Create** изменится на наименование **Edit** (рис. 6).

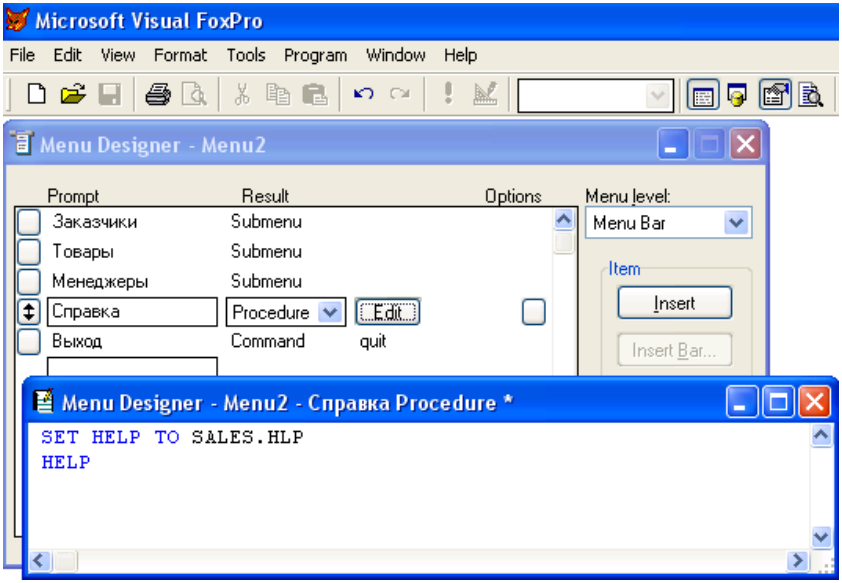

Рис. 6 Определение действий для пункта меню с открытым окном редактирования процедуры

Для создания подменю следует нажать кнопку **Create** выбранного пункта меню типа **Submenu**. На экране появится пустое окно конструктора меню, поле **Menu Level** которого содержит метку пункта строки меню (рис. 7). Пункты подменю определяются аналогично пунктам строки меню. Так как для редактирования основного меню и всех подменю применяется один конструктор меню, то для перехода между уровнями меню следует использовать поле **Menu Level**, а не закрывать окно конструктора меню.

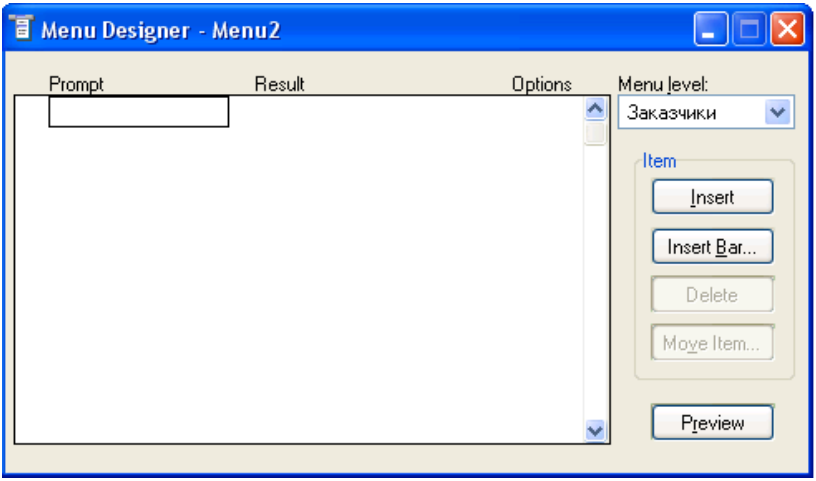

Рис. 7 Окно диалога создания подменю

Элементы создаваемого подменю можно разделить линиями, что позволяет улучшить внешний вид меню, объединить в группы схожие по смыслу команды. Разделительная линия представляет собой пункт меню, в котором в поле ввода **Prompt** вводятся символы «\-».

Обработка меню в проекте, созданном мастером

При автоматическом создании проекта мастер сам делает черновой вариант главного меню. Остается только отредактировать его в соответствии с нуждами конкретной задачи. Однако следует помнить о том, что настройка и корректная обработка главного меню в приложении производится либо в главной программе проекта, которая также создается автоматически, либо в редакторе класса, на основе которого создается приложение.

Для задания в качестве главного меню приложения следует:

1. Перейти на вкладку **Classes** в менеджере проекта, открыть библиотеку классов **SALES_APP** и выбрать класс создаваемого приложения **APP_APPLICATION** (рис. 8).

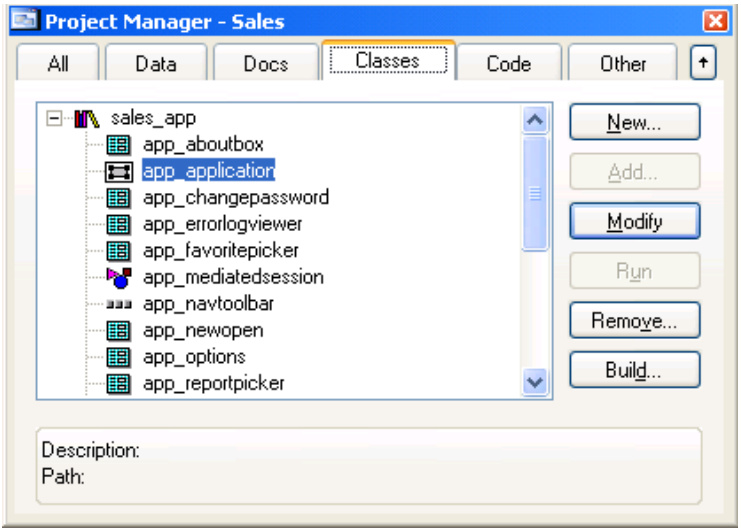

Рис. 8 Окно менеджера проекта с выбранным классом приложения

2. Для начала редактирования класса следует нажать кнопку **Modify**, после чего откроется окно конструктора классов (рис. 9).

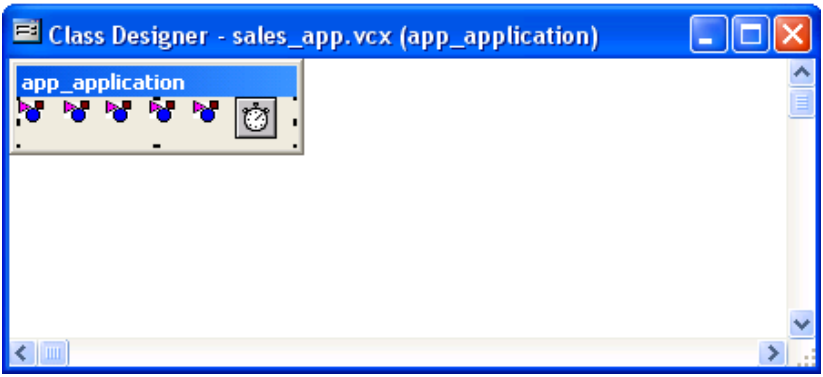

Рис. 9 Окно конструктора класса

3. Щелчком правой кнопки мыши на объекте класса следует вызвать контекстное меню, в котором необходимо выбрать пункт **Properties…**, после чего станет доступным окно свойств объектов (в случае, если оно ранее не было вызвано на экран).
4. В открывшемся окне следует найти свойство **cstartupmenu**, которое по умолчанию содержит в себе имя главного меню, автоматически сгенерированного мастером построения проекта.
5. Затем необходимо изменить это свойство, присвоив ему значение имени созданного пользовательского главного меню (просто написать новое имя меню в строке ввода окна и нажать Enter) (рис. 10).

6. В дальнейшем, при генерации исполняемого приложения, система автоматически подставит в исполняемый модуль меню с именем указанным разработчиком.

Важно: В качестве главного меню приложения нельзя указывать меню типа **Shortcut**.

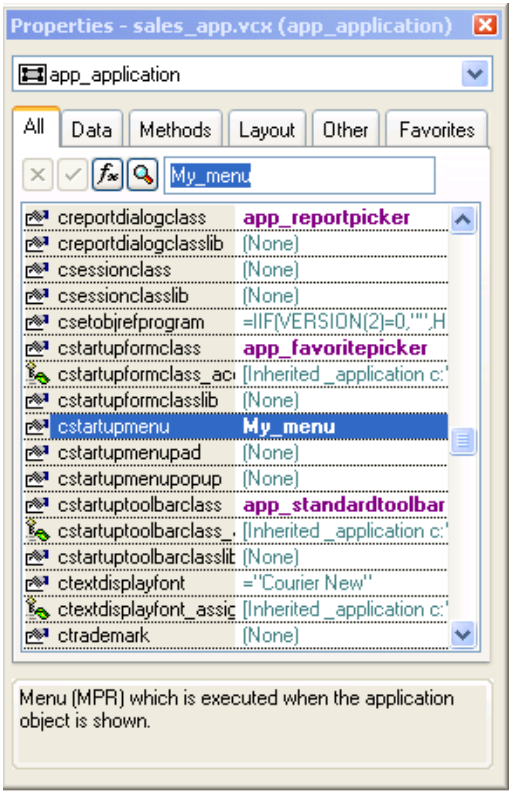

Рис. 10 Окно свойств объекта с выбранным свойством задания имени главного меню приложения

Альтернативным способом указания главного меню проекта является явное определение его имени в стартовом программном файле. Для этого необходимо перейти во вкладку **Code** менеджера проекта, выбрать стартовый файл и нажать кнопку **Modify**, после чего откроется окно редактирования программного кода. По умолчанию имя стартового файла мастер построения проекта формирует как ***<Имя проекта>_APP***, в нашем случае это файл **SALES_APP**. В окне редактирования программного кода необходимо найти строку, отвечающую за создание переменной, содержащей ссылку на объект типа приложения. Эта переменная всегда носит имя **APP_GLOBAL**, а строка создания выглядит следующим образом:

```
APP_GLOBAL = NEWOBJECT(APP_CLASSNAME, APP_CLASSLIB)
```

Функция *NEWOBJECT()* создает новый объект на основе ранее определенного класса. Значения переменных *APP_CLASSNAME* и *APP_CLASSLIB* определяются в заголовочном файле и изменять их не следует. Переменная *APP_GLOBAL* является глобальной, что означает ее видимость во всех функциях и процедурах, определенных в проекте.

Для указания главного меню проекта, после строки создания переменной проекта следует добавить оператор присваивания имени меню, свойству **cstartupmenu**. Итоговый программный код может выглядеть следующим образом:

```
APP_GLOBAL = NEWOBJECT(APP_CLASSNAME, APP_CLASSLIB)
```

```
*указываем имя разработанного главного меню проекта
app_global.cstartupmenu="My_menu"
```

Примечания:

1. Регистр символов при указании имени переменной не важен.
2. Использовать переменную APP_GLOBAL можно в любом месте программного кода приложения (в других процедурах, функция, программном коде экранных форм и т.д.).
3. Программный код в стартовом файле в основном отвечает за проверку системы и организацию работы окна приложения. Ничего удалять или модифицировать в нем не следует (лучше только добавлять новые команды, если разработчик понимает их действие).

Задание графических изображений пунктам меню

Visual FoxPro позволяет справа от команд пользовательского меню расположить графическое изображение, аналогичное тому, которое вы будите использовать для кнопки панели инструментов, выполняющей эту же команду. Например, добавим графическое изображение в пункт меню **О программе**, вызываемый из меню **Справка**. Для этого выполните следующие действия:

1. В конструкторе установите курсор на пункт меню **Справка**.
2. Из списка **Result** выберите значение **Submenu**.
3. Нажмите кнопку **Create** пункта меню **Справка**. На экране появляется пустое окно конструктора меню.
4. В поле **Prompt** введите наименование пункта меню **О программе**.
5. Нажмите для этой строки кнопку **Options**. Открывается диалоговое окно **Prompt Options.**
6. Область **Picture** содержит две опции, позволяющие указать источник графического изображения:
    - **File** – из графического файла;
    - **Resource** – из списка графических изображений, используемых Visual FoxPro в системном меню.
7. Установите опцию **Resource**.
8. Нажмите кнопку, расположенную справка от поля, которое находится под опцией. Открывается диалоговое окно **Insert System Menu Bar** (Вставить из системного меню) (рис. 11). Оно содержит список графических изображений и переключатель, позволяющий упорядочить значения списка по расположению в пунктах меню или по алфавиту.
9. Выберите из списка значение **Microsoft Visual FoxPro Help** и нажмите кнопку **ОК**. окно

закрывается. Выбранное значение переносится в область просмотра области **Picture** (рис. 12).

10. Нажмите кнопку ОК для закрытия диалогового окна **Prompt Options.**

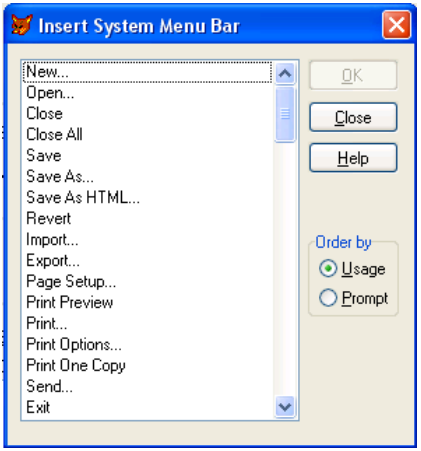

Рис. 11 Диалоговое окно **Insert System Menu Bar**

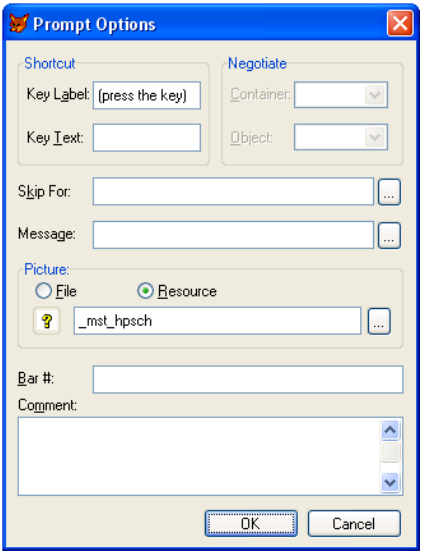

Рис. 12 Диалоговое окно **Prompt Options**

Определение параметров меню

Для строки меню можно определить процедуры, которые будут выполняться перед запуском меню, а также после выхода из него. Кроме того, вы можете указать месторасположение строки меню. Для этого используется диалоговое окно General Options (рис.13), которое открывается при выборе команды **General Options** из меню **View**.

В группе **Location** (Размещение) этого диалогового окна можно выбрать один из вариантов размещения строки созданного меню (табл. 1). В поле **Procedure** можно ввести фрагмент программы, которая будет выполняться при активизации строки меню.

Флажки Setup (Задать) и Cleanup (Очистить) используются для открытия окна редактирования процедуры, вызываемой при запуске сгенерированной этой программы.

Таблица 1.

Опции группы Location диалогового окна General Options

Опции	**Вид размещения**
Replace (Замещать)	Меню замещает основное меню Visual FoxPro
Append (Добавить)	Меню добавляется в основное меню Visual FoxPro
Before (Перед)	Меню вставляется перед указанным пунктом основного меню Visual FoxPro
After (После)	Меня размещается за указанным пунктом основного меню Visual FoxPro

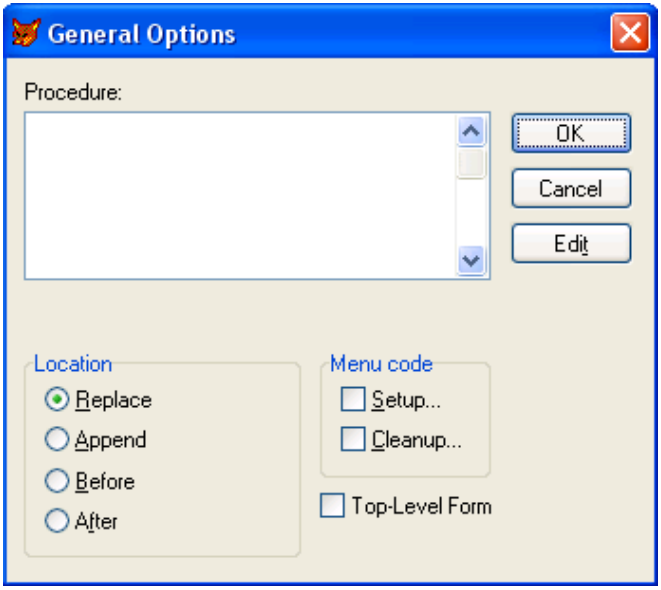

Рис.13 Диалоговое окно **General Options**

При установленном флажке **Top-Level Form** (Форма высокого уровня) меню будет отображаться в отдельном окне. В противном случае меню можно использовать только в окне Visual FoxPro.

Создание всплывающего меню

В Visual FoxPro имеется возможность создания *всплывающего меню* средствами конструктора меню. Способ создания меню данного типа аналогичен созданию горизонтального меню в виде строки. Для этого меню, как и для обычного, можно определить оперативные клавиши и опции, устанавливаемые в диалоговом окне **Prompt Options.** Чтобы

создать всплывающее меню, выполните следующею последовательность действий:

1. Откройте проект.
2. Для открытия окна конструктора меню в окне проекта перейдите на вкладку **Other** и выберите группу **Menus**.
3. Нажмите кнопку **New** окна проекта.
4. В открывшемся диалоговом окне **New Menus** нажмите кнопку **Shortcut.** Откроется окно конструктора меню.
5. В поле Prompt последовательно введите тексты пунктов меню и определите для них выполняемые действия.
6. Выполните генерацию.
7. Запустите меню на выполнение. Вид данного меню при запуске представлен на рис.14.

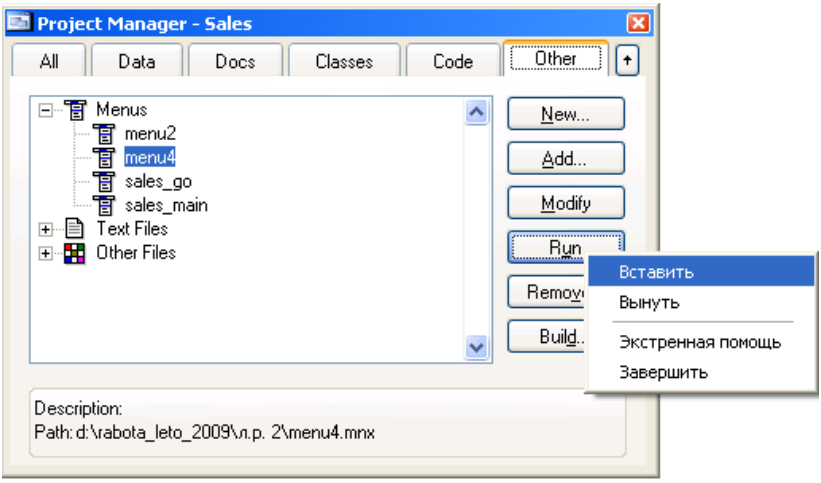

Рис.14 Меню типа **Shortcut**

Задание

1. Нарисуйте на бумажном носителе структуру меню Вашего проекта. Обоснуйте эту структуру и опишите подход, применяемый Вами при построении меню.
2. Создайте меню для Вашего проекта в конструкторе меню. Отобразите в отчете тексты процедур и команды, используемые в меню.
3. Просмотрите вид меню на экране, и отметьте поведение каждого пункта меню.
4. Проверьте правильность указания главного меню в классе приложения. Укажите главное меню приложения в программном коде главного файла проекта. Отобразите в отчете Ваши действия по установке главного меню приложения.

Лабораторная работа № 10

Тема: Управление проектом и создание приложений

Цель: Научиться создавать приложение

Определение свойств окна проекта

Вкладка «Project» окна диалога «Options» (рис. 1), вызываемая командой **Tools | Options**, позволяет определить два свойства окна проекта. Прежде всего, с помощью переключателя **Project double–click action** можно выбрать действие, выполняемое при двойном нажатии кнопки мыши. По умолчанию установлен признак модификации выбранного компонента проекта, поэтому при двойном нажатии кнопки мыши вызывается конструктор, используемый для модификации выбранного типа файла. Можно установить переключатель в положение **Run selected file**. В этом случае при двойном нажатии кнопки мыши выбранный файл запускается на выполнение. Для таблиц вместо запуска на выполнение осуществляется открытие окна таблицы в режиме BROWSE.

Если установлен флажок **Prompt for Wizard**, при создании нового компонента проекта открывается окно диалога с запросом об использовании мастера для его создания. Если флажок не установлен, то сразу же вызывается соответствующий создаваемому объекту конструктор.

Для настройки основных параметров управления проектом можно использовать область «Source control options» (Параметры хранилища данных) окна диалога «Options», предназначенные для управления крупным проектом, выполняемым группой разработчиков.

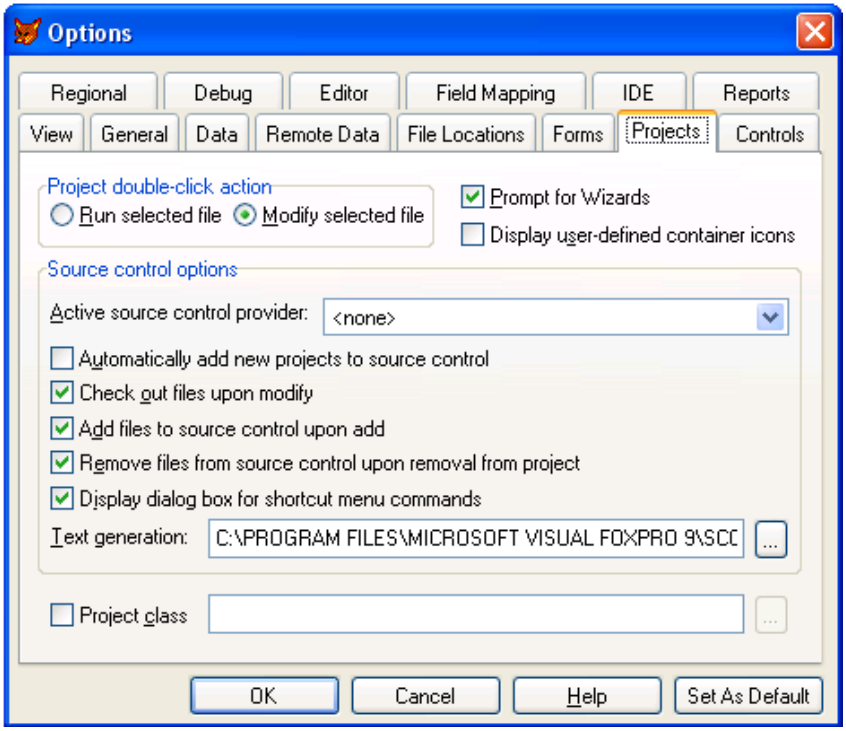

Рис. 1 Определение параметров окна проекта

Задание параметров проекта

Для каждого проекта можно задать свою специализированную информацию, которая будет использоваться в процессе построения проекта. Для определения этой

информации следует выполнить команду меню **Project | Project Info.** Открывшееся окно диалога «Project Information» содержит три вкладки «**Project**», «**Files**» и «**Servers**». Во вкладке «Project» (рис. 2) можно указать информацию о разработчике, место хранения проекта, параметры генерации приложения и определить пиктограмму для выполняемого файла приложения.

Информация о разработчике

Для того чтобы указать информацию о разработчике, необходимо ввести ее в поля **Author** (Автор), **Company** (Фирма)**, Address** (Адрес)**, City** (Город)**, Country** (Страна)**, State** (Область) и **Postal Code** (Почтовый индекс).

Место хранения проекта

Поле ввода Home используется для определения места хранения проекта. Для определения каталога используется окно диалога «**Select Directory**» (рис. 3), которое вызывается при нажатии на кнопку, расположенную правее поля ввода. Можно также указать полный путь к каталогу хранения проекта. В любом случае все ссылки на подкаталоги проекта происходят относительно текущего пути.

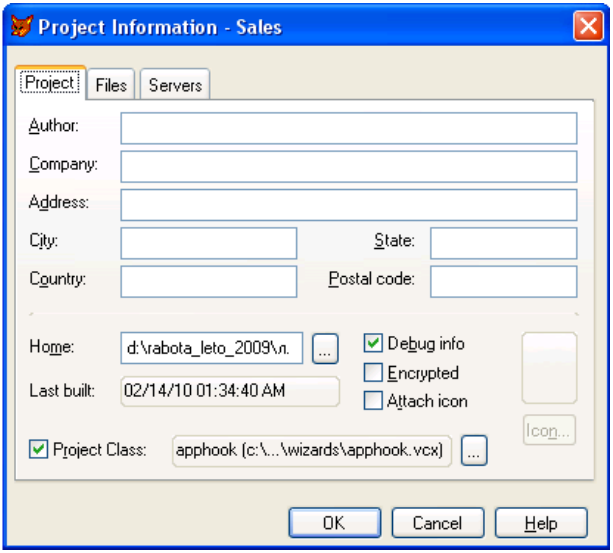

Рис. 2 Окно диалога **Project Information**

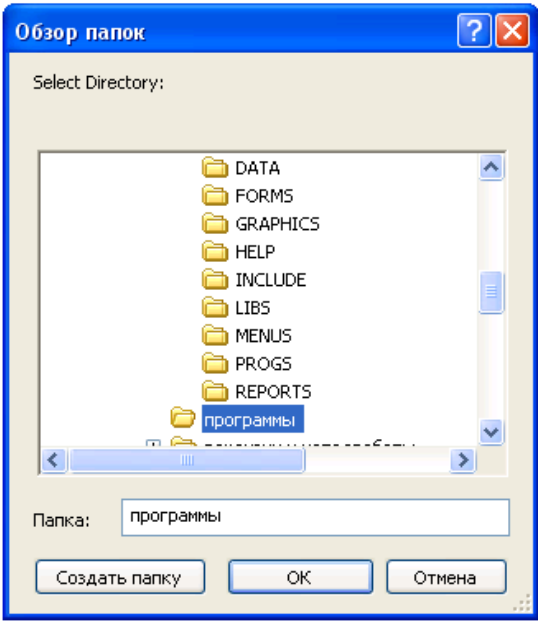

Рис. 3 Окно диалога **Select Directory**

Определение пиктограммы для выполняемого файла

При установке флажка **Attach icon** (Определить значок) в окне диалога «**Project Information**» открывается окно диалога «**Open**» (рис. 4), в котором можно выбрать пиктограмму для создаваемого приложения (файл с расширением .ICO).

В комплект поставки Visual FoxPro входит большое количество готовых пиктограмм, которые для удобства сгруппированы по категориям. Пиктограммы находятся в подкаталоге **ICONS** (Значки), подкаталога **GRAPHICS**, подкаталога **SAMPLES** корневого каталога Visual FoxPro. Общий путь к «иконкам» выглядит как:

<Каталог Visual FoxPro>\SAMPLES\GRAPHICS\ICONS\<Групповой подкаталог>

При желании можно создать собственную пиктограмму, воспользовавшись программой **Image Editor**, которая также входит в состав поставки Visual FoxPro и находится в его корневом каталоге. Запуск программы следует осуществлять средствами операционной системы, так как она представляет собой законченное приложение Windows и может использоваться отдельно от Visual FoxPro.

Информацию в поле **Last built** содержит дату последнего построения проекта. Для включение отладочной информации в используемый код, нужно установить флажок **Debug info** (Информация об отладке). Эта информация поможет при отладке программы и исправлении ошибок, возникших на компьютере конечного пользователя. Используйте флажок **Debug info**

(Информация об отладке) только в случае необходимости, т.к. его установка может привести к декомпиляции программы.

Чтобы сохранить права на интеллектуальную собственность разработанного проекта, используйте флажок **Encrypted** (Шифровать). Установите этот флажок, если хотите, чтобы Visual FoxPro зашифровал исполняемый код проекта, повышая тем самым степень защиты вашей программы от декомпиляции.

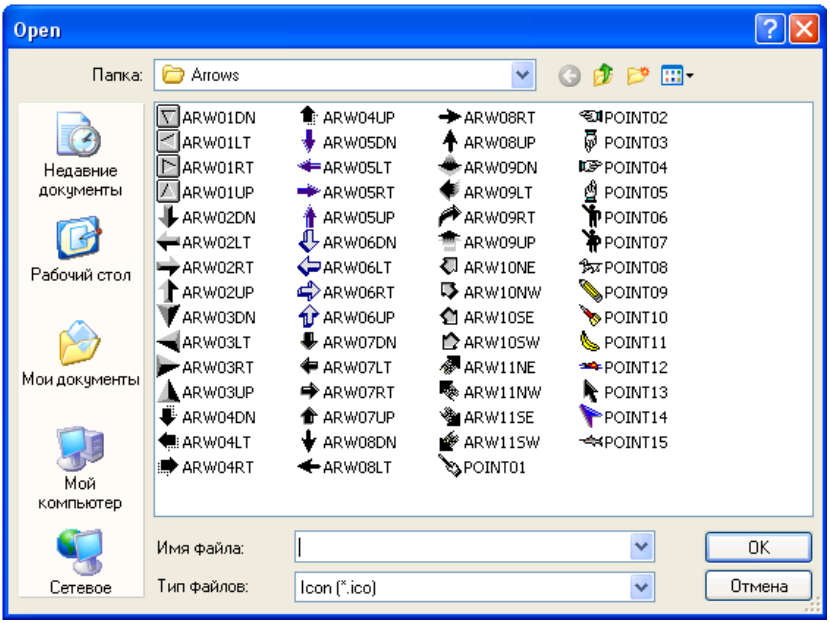

Рис. 4 Выбор пиктограммы

Просмотр файлов проекта

Вкладка «**Files**»окна диалога «**Project Information**» содержит список файлов, входящих в проект (рис. 5). Кнопки в

верхней части окна диалога предназначены для упорядочения отображения файлов в окне диалога.

Название столбца	**Назначение**
Type	Показывает тип файла в виде пиктограммы
Name	Содержит имя файла
Last Modified	Указывает системную дату и время последней модификации файла
Included	Определяет включение файла в проект
Code Page	Содержит значение кодовой страницы для каждого файла

Для упорядочивания данных необходимо щелкнуть на заголовке соответствующего столбца.

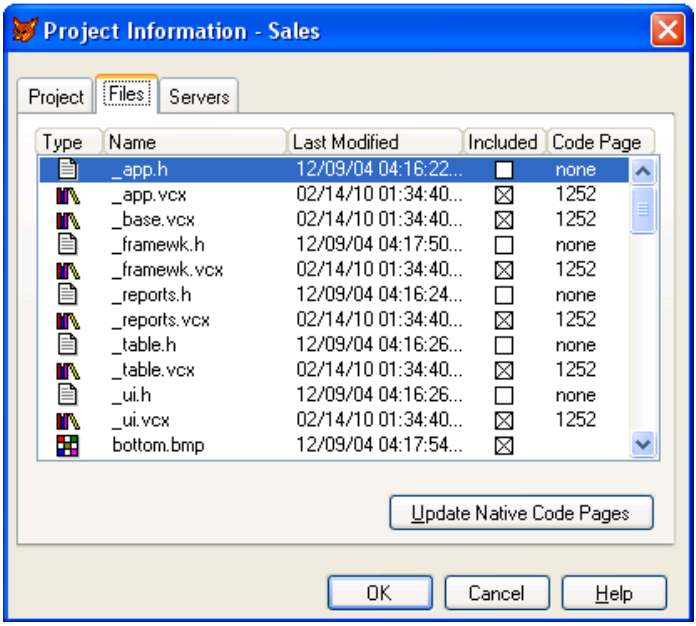

Рис. 5 Вкладка «**Files**» окна диалога «**Project Information**»

Предупреждение! Внимательно следите за тем, чтобы файлы базы данных и таблиц *не были включены в проект*, так как в противном случае при запуске конечного приложения они будут открыты только на чтение. Это допустимо только в тех случаях, когда таблицы содержат информацию, которая не изменяется в процессе работы программы.

Установка основной программы проекта

Проект обязательно должен содержать программу, которая запускает приложение и управляет его работой. Такой файл называется *основной программой,* и им чаще всего является меню приложения, но можно определить в качестве основной программы созданную пользователем программу управления приложением. Для того чтобы сделать компонент основным, следует выбрать его в окне диалога «**Project Information**» или в окне проекта и выполнить команду контекстного меню **Set Main** (Основная программа). В окне проекта имя файла выделится жирным шрифтом, указывая, что данный компонент является основной программой (рис. 6). При создании проекта мастером, он автоматически генерирует главную программу управления приложением и устанавливает ее основной. Рекомендуется без особой нужды не менять эту установку для избежания некорректной работы приложения в дальнейшем.

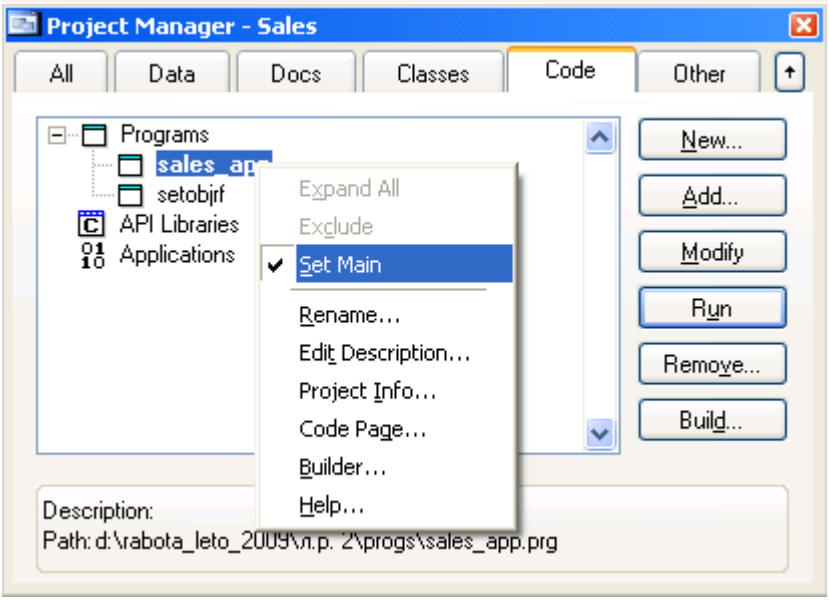

Рис. 6 Установка основной программы

Использование опции Exclude

Выполняемые компоненты проекта (программы, формы, меню) при разработке приложения с помощью проекта объединяются для создания кода приложения. Невыполняемые компоненты, такие как таблицы, индексы включаются в приложение доступными только для чтения. Можно разрешить конечному пользователю модифицировать компоненты, исключив их из проекта.

Чтобы исключить компоненты из проекта, можно воспользоваться следующими средствами:

- командой контекстного меню **Exclude** (Исключить);
- командой **Exclude** из меню **Project.**

Рядом с исключенным компонентом в окне проекта появляется кружок, перечеркнутый линией. Исключенные из проекта компоненты остаются в списке проекта, но не включаются в состав приложения при его построении. Поэтому необходимо самостоятельно отслеживать их наличие и доступ к ним из приложения.

Для включения в проект исключенного компонента необходимо воспользоваться командой контекстного меню **Include** (Включить) или командой **Include** из меню **Project.**

Очистка проекта от удаленных файлов

Вся информация о проекте храниться в системной таблице Visual FoxPro. При разработке приложения после удаления компонента из проекта в этой таблице ставится лишь метка о его удалении. Чтобы окончательно удалить из проекта информацию обо всех помеченных на удаление компонентах, необходимо упаковать таблицу проекта, воспользовавшись командой **Clean Up Project** (Упаковать проект) из меню **Project.**

Построение проекта и создание приложения

При построении проекта Visual FoxPro просматривает все компоненты, перечисленные в проекте, и формирует проект. Если какие–либо компоненты вызывают в свою очередь другие компоненты, то они также должны быть включены в проект. Например, в проект должны быть включены все программы, формы и отчеты, которые вызываются при выборе пунктов меню.

Для создания проекта необходимо нажать кнопку **Build** в окне проекта. На экране откроется окно диалога «**Build Options**» (рис. 7), которое содержит опции:

- **Rebuild Project** (Построитель проекта) – собирает проект, проверяя наличие в нем всех необходимых файлов;
- **Application (app)** (Приложение) **–** создает исполняемое приложение с расширением **app.** Данный файл может запускаться на выполнение в Visual FoxPro командой **Do** (Выполнить) из меню **Program** (Программы);
- **Win32 executable/COM server (exe)** (Исполняемый файл) – создает исполняемое приложение с расширением exe. Данный файл может запускаться на выполнение как в главном окне Visual FoxPro, так и вне его при наличии соответствующих библиотек;
- **Single-threaded COM server (dll)** (Однопоточный COM-сервер) – создает однопоточную динамическую библиотеку (Dynamic Link Library) с расширением dll;
- **Multi-threaded COM server (dll)** (Многопоточный COM-сервер) – создает многопоточную динамическую библиотеку (Dynamic Link Library) с расширением dll.

В диалоговом окне **Build Options** (Опции построителя) расположены флажки, позволяющие задать параметры создаваемого проекта:

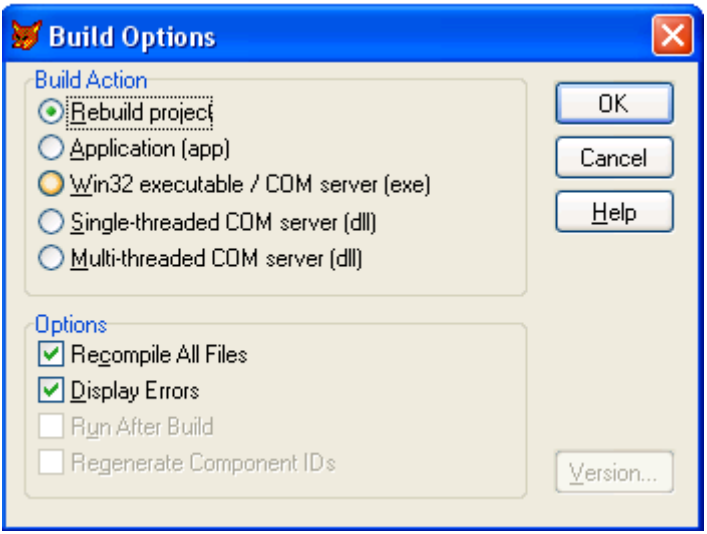

Рис. 7 Окно диалога «**Build Options**»

- **Recompile All Files** (Обновлять все файлы) – устанавливается для обновления всех компонентов проекта. По умолчанию обновляются только те компоненты проекта, которые были изменены после предыдущего построения;
- **Display Error** (Показывать ошибки) – устанавливается в том случае, если после завершения построения необходимо в отдельном окне отобразить все ошибки, встретившиеся в процессе построения (рис 8). Если же флажок не

установлен, ошибки построения можно просмотреть, выполнив команду **Project | Errors**;

- **Run After Build** (Запустить после построения) – устанавливается в том случае, если необходимо после создания приложения сразу запустить его на выполнение;
- **Regenerate Component IDs** (Перестроить идентификаторы Automation-серверов) – устанавливается в том случае, если необходимо установить и регистрировать Automation-серверов, содержащие в проекте.

Кнопка **Version** (Версия) открывает одноименное диалоговое окно, в котором можно указать информацию о номере и типе версии приложения.

Флажок **Regenerate Component IDs** доступен только при установленной опции **Win32 executable/COM server (exe), Single-threaded COM server** или **Multi-threaded COM server.**

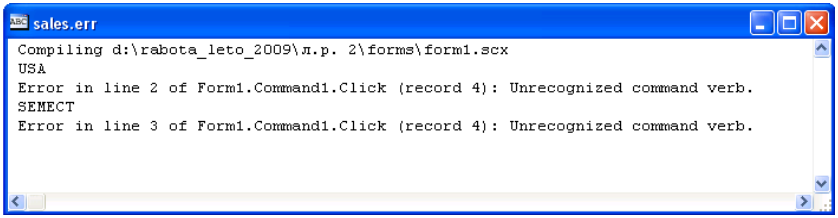

Рис.8. Диалоговое окно, сообщающее об ошибках, возникающих при построении приложения

Для построения проекта необходимо выбрать опцию **Rebuild Project** и нажать кнопку **OK**. При появлении запроса о необходимости сохранения проекта нажать кнопку **Yes**.

Если в процессе построения проекта диспетчер проектов обнаружит компонент, не описанный в проекте, на экране появится окно диалога «Locate File» (рис. 9), содержащее имя ненайденного файла. Если файл был скопирован в другой каталог, то для поиска компонента следует нажать кнопку **Locate** и найти на диске необходимый файл. Вы можете проигнорировать данную ошибку. В этом случае нажмите кнопку **Ignore.** Если файл был удален вне окна проекта, его можно удалить из списка файлов проекта кнопкой **Remove**. Если принято решение игнорировать данную ошибку, следует нажать кнопку **Ignore**.

Список всех обнаруженных ошибок сохраняется в файле с именем файла проекта и расширением **err**.

Когда все необходимые компоненты включены в проект, можно выбрать опции **Application** или **Win32 executable/COM server**, которые создают приложение, выполняемое под управлением Visual FoxPro, и автономное приложение соответственно. Для создания файла с расширением app, который может запускаться на выполнение из программы Visual FoxPro, необходимо использовать опцию **Application.** Если необходимо создать файл с расширением exe, который может запускаться автономно, воспользуйтесь опцией **Win32 executable/COM server** .

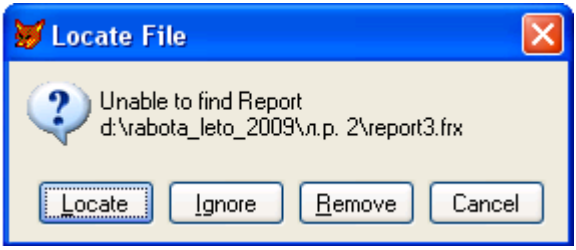

Рис. 9 окно диалога «Locate Options»

После создания, приложение можно запустить его используя команду **DO**, которую нужно ввести в окне **Command** Visual FoxPro, или команду **DO** из меню **Program.** При выполнении этой команды открывается диалоговое окно **DO.** Откройте в нем нужную папку, в списке файлов выберете созданный вами файл проекта, имеющий расширение app, и нажми кнопку **DO.** Для выполнения приложения, созданного с опцией **Win32 executable/COM server** необходим доступ к библиотеке VFP300.ESL.

Созданное приложение может работать автономно от Visual FoxPro только в случае корректной настройки всех необходимых параметров системы. Сюда входит настройка драйверов ODBC, установка всех используемых сервисных приложений, регистрация в системе всех используемых динамических библиотек. Большинство проблем настройки можно решить, установив на компьютере Visual FoxPro. Однако такой способ не всегда приемлем как по техническим причинам, так и по соображениям безопасности. Для решения таких проблем в стандартную поставку Visual FoxPro включен мастер создания

установки приложения «Setup Wizard». Он создает стандартную инсталляцию системы Windows, берет на себя организацию автоматического расположения всех компонентов проекта, регистрацию всех используемых сервисов и т.п. Мастер позволяет настроить параметры установки приложения, а также начальную конфигурацию самого приложения. Предоставляется возможность создания либо целой инсталляции, либо инсталляции, разбитой на образы дискет. В последнем случае все установочные компоненты разбиваются на блоки размером ровно в емкость дискеты, что предельно упрощает перенос и установку приложения на удаленный компьютер. Однако необходимо помнить, что все регистрируемые сервисы включаются мастером в тело инсталляции. Поэтому установочный пакет даже небольшого приложения, использующего очень много внешних сервисов, может занимать очень много места, и на его переноску может потребоваться не один десяток дискет.

Галерея компонентов Visual FoxPro

Галерея компонентов является средством организации разработки в Visual FoxPro, позволяющим разработчику группировать и упорядочивать компоненты, такие как используемые в работе проекты, библиотеки, классы, формы, отчеты, кнопки и т.п. созданное вами упорядочение можно динамически изменять, что позволит использовать разные подходы к классификации компонентов разработки.

В Галереи компонентов для работы можно разместить различные элементы Visual FoxPro, локальные и удаленные документы, файлы или папки, Automation-серверов, например, Microsoft Excel, Microsoft Word и HTML_файлы. В ней содержаться также новые базовые классы Visual FoxPro.

Галерея компонентов призвана создать рабочею среду. Она является для разработчика хранилищем данных, которое используется для быстрой разработки приложений. В ней можно разместить не только компоненты, из которых создается приложение, но и созданные проекты, статьи и другие документы, содержащие информацию, полезную для разработки. В окне Галереи компонентов можно также хранить ссылки на используемые при разработке Web- страницы.

Галерея компонентов располагает средствами для создания новых проектов, форм, а также для изменения свойств объектов и классов. Объекты, размещенные в Галереи компонентов, можно переносить в проекты и формы (и наоборот) с помощью метода «перенести-и-оставить».

Запуск Галереи компонентов

Для запуска Галереи компонентов выполните одно из перечисленных ниже действий:

- в меню **Tools** команду **Component Gallery**;
- в командном окне введите команду **DO (_GALLERY)**

При этом на экране открывается окно Галереи компонентов (рис. 10). Оно разделено на две области. В левой

области размещен иерархический список каталогов, а в правой – содержимое каталога, выбранного в левой области.

В верхней части окна Галереи компонентов находится панель инструментов, кнопки которой позволяют настроить параметры окна Галереи компонентов, найти нужный объект, а также управляют отображением объектов в окне.

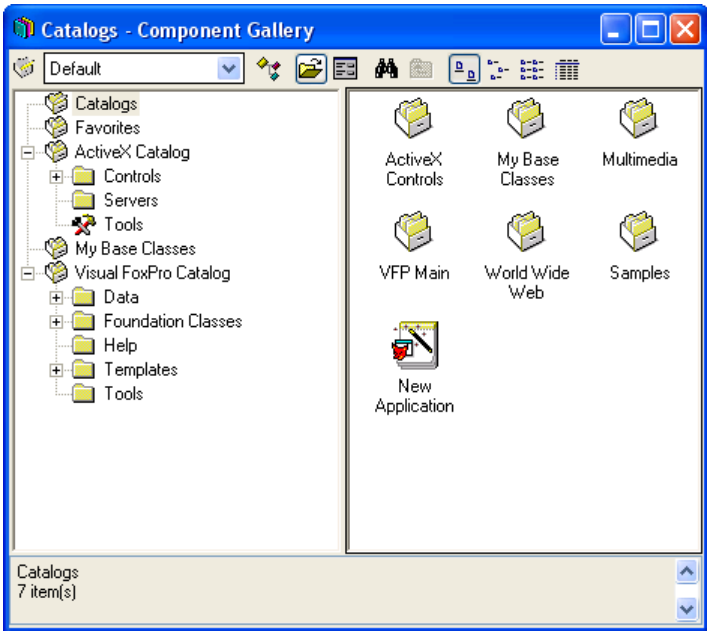

Рис. 10. Галерея компонентов

Расширенные свойства проекта

СУБД VFP 9.0 содержит встроенное средство управления параметрами компонентов проекта **Application Builder**. Частично оно предоставляет возможности управления параметрами проекта описанные в разделе информации о проекте на рис.2. Вызвать

Application Builder можно при помощи комбинации клавиш Alt+F11. Перед этим следует убедиться, что пути к системным файлам настроены должным образом. Проверить настройку путей можно во вкладке *File Locations* диалогового окна *Options* (рис. 11), вызвать которое можно выполнив команду главного меню **Tools | Options**.

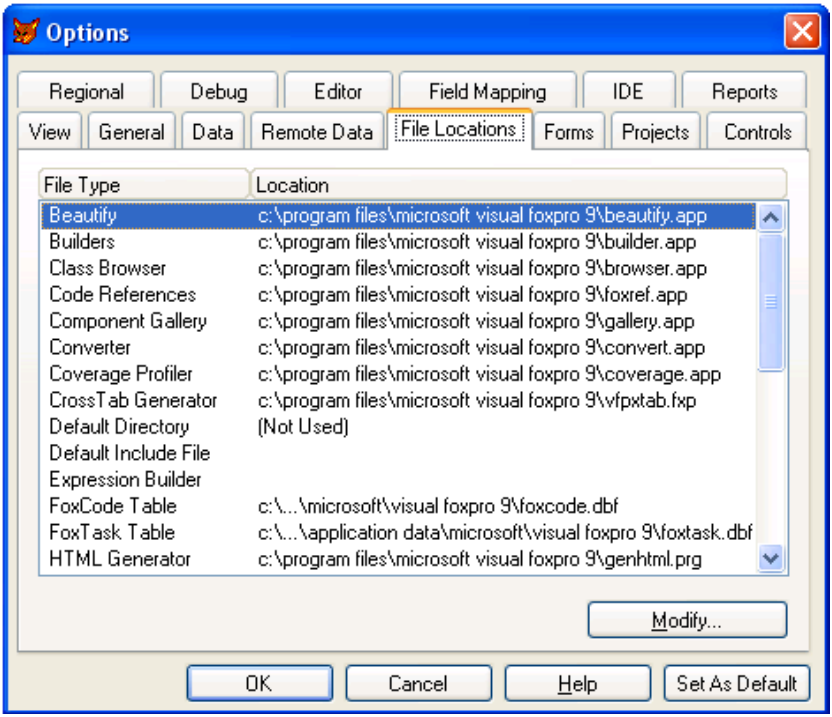

Рис. 11. Диалоговое окно Options с выбранной вкладкой определения путей доступа

При вызове **Application Builder** на экране появляется диалоговое окно с несколькими листовыми вкладками. По

умолчанию активной является первая вкладка General (рис. 12). В ней можно определить некоторые глобальные свойства проекта.

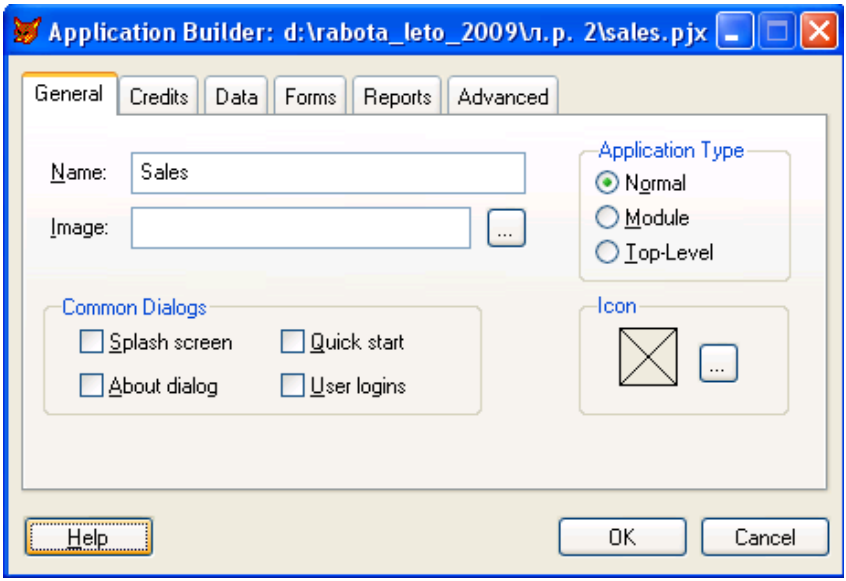

Рис. 12. Вкладка General диалогового окна Application Builder

Поле ввода **Name** позволяет определить название, которое выводится в заголовке главного окна приложения, а так же в стандартном окне *About dialog*.

В поле **Image** можно указать графический файл (.ICO, .BMP, .GIF, .JPG), который будет отображаться в окне приветствия приложения и стандартном окне About dialog.

Блок выключателей Common Dialogs позволяет управлять некоторыми стандартными компонентами проекта:

Splash screen – окно приветствия. Появляется на короткое время в центре экрана при запуске приложения в операционной

системе. На работу абсолютно не влияет, является сугубо декоративным элементом.

About dialog – окно с информацией о проекте. Является простой формой, которая автоматически генерируется построителем и добавляется в список форм проекта. Запустить ее на исполнение можно стандартной командой **DO <имя формы>.** Почти всегда требует дополнительного редактирования при разработке.

Quick start – окно быстрого запуска форм и отчетов проекта. Интерактивная форма, в которой содержатся имена всех форм проекта, а так же кнопка запуска. Позволяет пользователю запускать произвольную экранную форму минуя главное меню. Использовать следует с осторожностью, так как отображаются имена форм, заданные на этапе разработки проекта, которые не всегда отражают выполняемые функции.

User logins – окно запроса пользовательского имени и пароля. При запуске приложения на экране появляется диалоговое окно с запросом аутентификации пользователя. На самом деле, создается и добавляется в список форм проекта шаблон формы. Процедуру авторизации всяко придется прописывать вручную используя конструктор форм.

Блок переключателей **Application Type** позволяет определить тип приложения. Возможны три варианта:

Normal – в дальнейшем будет сгенерировано обычное приложение, запускаемое на базе платформы VFP.

Module – будет сгенерирован автономный модуль, который может быть добавлен в другой проект или вызван из другого приложения.

Top–Level – будет сгенерировано приложение, запускаемое на базе платформы Windows, без доступа к компонентам VFP.

Блок **Icon** позволяет выбрать пиктограмму, отображаемую в строке заголовка приложения и форм.

Вкладка **Credits** (рис. 13) позволяет указать информацию о разработчике.

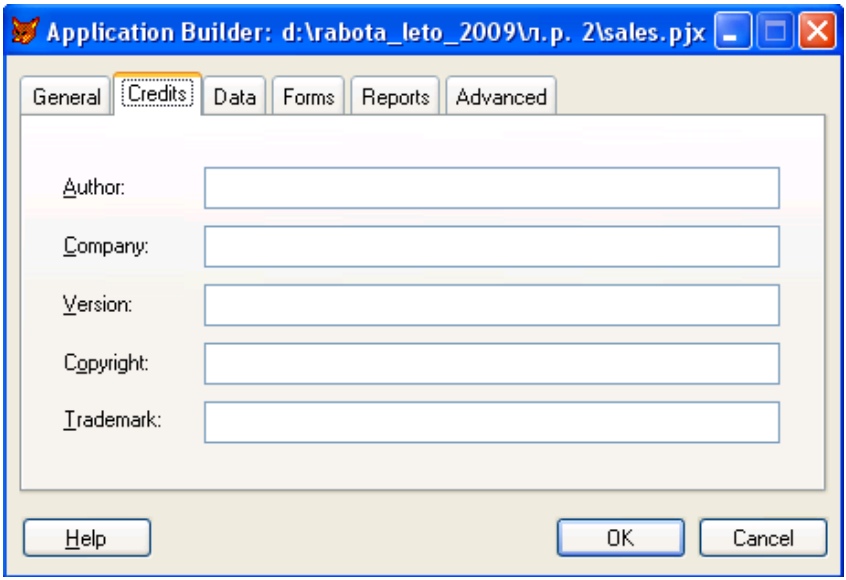

Рис. 13. Вкладка с информацией о разработчике окна Application Builder

Обратите внимание, что указываемая информация несколько отличается от той, что можно указать в окне информации о проекте на рис. 2.

Вкладка **Data** (рис. 14) представляет обширнейший набор средств быстрой разработки приложения. Вверху справа находятся две кнопки **Database Wizard** и **Table Wizard**, которые позволяют вызвать мастер построения базы данных и таблиц соответственно. Созданные элементы автоматически добавляются в проект. Работают как и обычные мастера построения. Кнопка **Select** позволяет выбрать уже существующую базу данных или таблицу. При выборе базы данных все, входящие в нее таблицы и представления данных будут включены в табличную область вкладки. Кнопка **Clear** очищает эту табличную область.

Выключатели в колонках **Form** и **Report** позволяют указать источники данных, для которых потом будут автоматически сгенерированы экранные формы и отчеты. Используя ниспадающие списки в нижней части экрана **Form Style** и **Report Style** можно выбрать один из стандартных системных стилей для создаваемых элементов, которые аналогичны стилям в мастерах построения форм и отчетов.

Основным функциональным элементом вкладки является кнопка **Generate**. При нажатии на нее система произведет автоматическое построение форм и отчетов для указанных таблиц и представлений данных. Сгенерированные компоненты будут автоматически добавлены в проект.

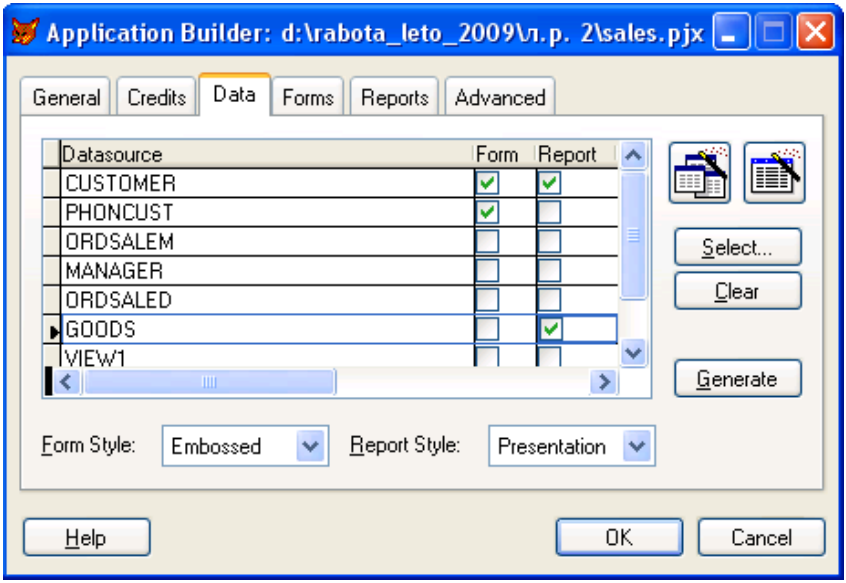

Рис. 14. Вкладка Data окна Application Builder

Примечание: следует использовать возможности автоматической генерации крайне осмотрительно.

Во–первых, созданные компоненты отличаются крайним аскетизмом. Формы позволяют производить только основной набор операций и только над одним источником данных, а отчеты – просто представляют собой набор данных из таблиц и представлений. Никакая логика обработки данных автоматически не генерируется.

Во–вторых, средства автоматической генерации очень чувствительны к настройкам системы. Достаточно небольшой неточности при указании путей к построителям и вместо набора форм и отчетов можно получить набор сообщений об ошибках.

В–третьих, построители используют стандартные библиотеки классов. При необходимости изменения автоматически построенных форм и отчетов, скорее всего, придется менять практически все компоненты и переписывать весь программный код.

Вкладка **Forms** (рис. 15) представляет возможности по настройке некоторых свойств экранных форм приложения.

В левой части вкладки расположен список всех экранных форм, включенных в проект. При помощи кнопок **Add…**, **Edit** и **Remove** разработчик может осуществлять основные операции с формами: добавить существующую форму в проект, отредактировать в конструкторе выбранную форму, удалить выбранную форму из проекта.

Поле ввода **Name** позволяет задать альтернативное имя экранной форме. Автоматически для всех форм альтернативное имя формируется путем конкатенации собственного имени формы и дополнительного слова Form, например: для формы с именем Form1 система автоматически сгенерировала альтернативное имя Form1 Form, для формы Customer – Customer Form и т.п.

Блок выключателей позволяет управлять визуальными особенностями каждой формы отдельно:

Single Instance – определяет возможность повторного запуска формы в приложении. Если выключатель включен, то форму можно будет запускать только в одном экземпляре, запуск

этой же формы из другого места будет невозможен, т.е. получить на экране две одинаковые формы не получится.

Use Navigation Toolbar – определяет, будет ли добавлена на форму стандартная кнопочная панель навигации.

Use Navigation Menu – определяет, будет ли добавлено на форму стандартное меню навигации.

Appear in File New dialog – определяет, будет ли включаться имя формы в пункт создания нового файла в меню приложения.

Appear in File Open dialog – определяет, будет ли включаться имя формы в пункт открытия файла в меню приложения.

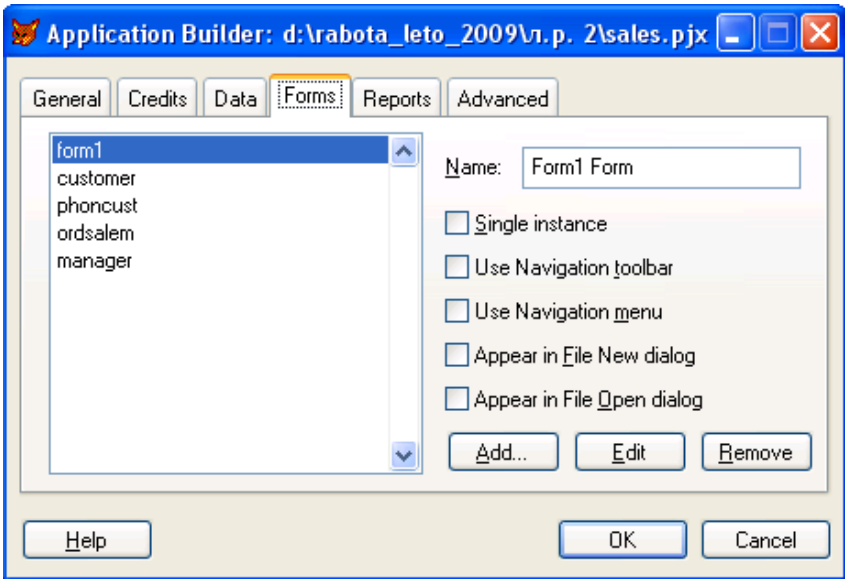

Рис. 15. Вкладка Forms диалогового окна Application Builder

Вкладка **Reports** (рис. 16) предоставляет возможность по настройке отчетов приложения.

Как и в предыдущей вкладке слева находится список всех отчетов проекта. При помощи кнопок **Add…**, **Edit** и **Remove** разработчик может осуществлять основные операции с отчетами: добавить существующий отчет в проект, отредактировать в конструкторе выбранный отчет, удалить выбранный отчет из проекта.

Поле ввода **Name** позволяет задать альтернативное имя отчета. Автоматически для всех отчетов альтернативное имя формируется путем конкатенации собственного имени отчета и дополнительного слова Report.

Выключатель **Appear in Print Reports dialog** определяет для каждого отчета, будет ли отображаться альтернативное имя в диалоговом окне Print Reports приложения.

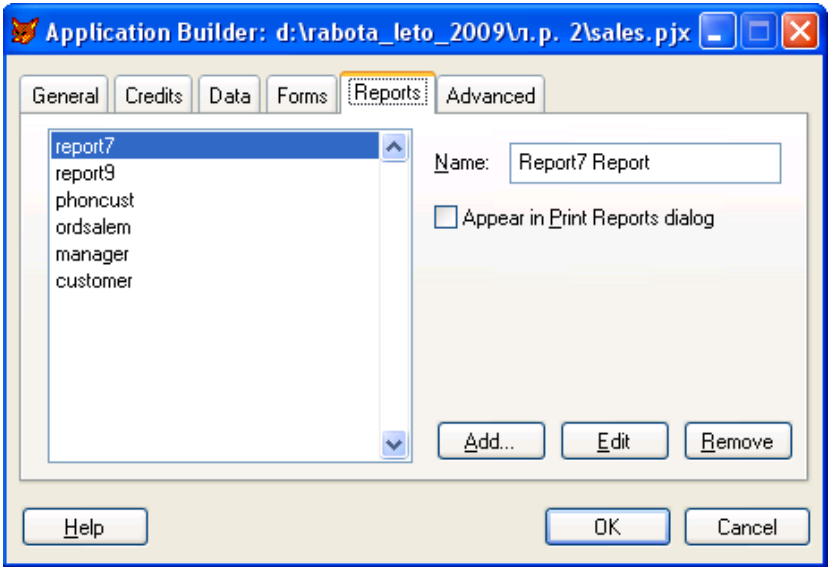

Рис. 16. Вкладка Reports диалогового окна Application Builder

Вкладка **Advanced** (рис. 17) позволяет произвести дополнительные настройки приложения.

Поле ввода **Help file** позволяет указать путь и файл помощи приложения. Система VFP поддерживает два формата файлов помощи .CHM и .HLP. Создавать файлы помощи необходимо в сторонней программе, например *HelpWorkshop*.

В поле ввода **Default data directory** можно указать каталог по–умолчанию для файлов данных. Обычно используется в случае, если проект оперирует данными, находящимися вне каталога проекта. Почти всегда это многопользовательские приложения с единым хранилищем данных.

Блок **Menus** определяет, будет ли включена в приложение стандартная кнопочная панель (**Standard toolbars**) и будет ли доступно меню *Favorites* из приложения.

Кнопка **Cleanup** используется для синхронизации изменений, сделанных в окне **Application Builder** с компонентами проекта. При этом синхронизируются все элементы проекта, а так же производятся соответствующие изменения в таблицах компонентов проекта.

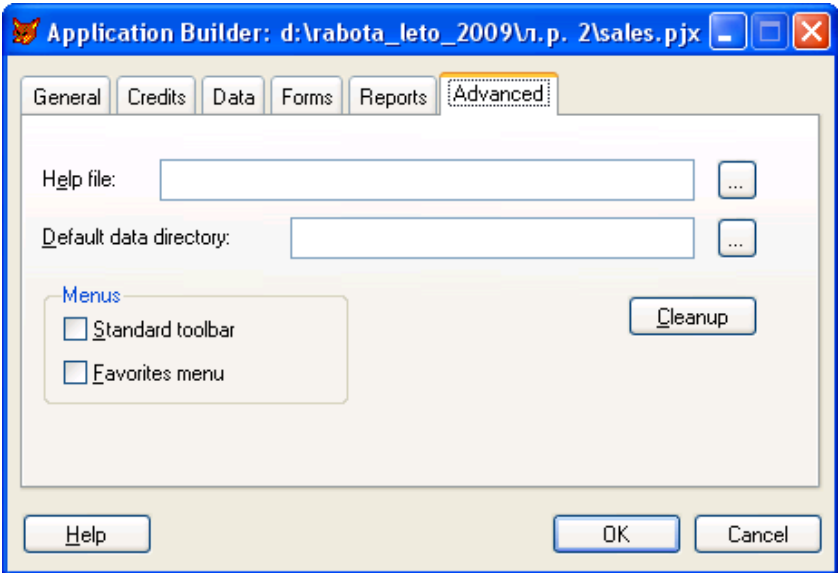

Рис. 17. Вкладка Advanced диалогового окна Application Builder

Задания

1. Укажите для Вашего проекта информацию о разработчике. Информация должна быть реальной и отражать именно данные конкретного разработчика именно этого проекта.
2. Создайте пиктограмму для своего проекта и определите ее для выполняемого файла. Для создания пиктограммы следует использовать встроенное средство Visual FoxPro.
3. Отметьте в виде таблицы все файлы, включенные в проект, их типы и кодовые страницы.
4. Создайте на основе проекта неавтономное приложение, работающее под управление Visual FoxPro и автономное

приложение. Отметьте разницу при работе с обеими конечными программами.

5. Исследуйте возможности блока Common Dialogs вкладки General окна Application Builder. Для этого поочередно включайте по одному выключателю, компилируйте проект и запускайте его. В отчете, для каждого выключателя, отразите, какие изменения происходят в интерфейсе приложения при его запуске, а так же какие изменения возникают в компонентах проекта (появляются ли новые формы и т.д.).
6. Проведите такое же исследование для блока Menus вкладки Advanced. Результаты отразите в отчете.

Лабораторная работа № 11

Тема: Использование пользовательских классов

Цель: Изучить создание и работу с пользовательскими классами

Пользовательские классы являются мощным средством организации программ в любом объектно–ориентированном языке программирования. VFP представляет обширный набор инструментальных средств для работы с пользовательскими классами, начиная от явного объявления класса в программном коде и заканчивая мощным визуальным построителем классов. Преимущество использования пользовательских классов заключается в том, что программист может использовать объекты произвольного уровня сложности, наследующие свойства и методы других объектов, а так же содержащие дополнительные, определенные самим программистом.

В VFP пользовательские классы объединяются в библиотеки классов и хранятся в файлах с расширением VCX. Допускается подключение в проект произвольных библиотек классов, написанных на любом языке программирования, в том числе и на самом VFP. Подключенные в проект библиотеки классов можно просмотреть/отредактировать во вкладке **Classes** окна менеджера проекта (рис. 1).

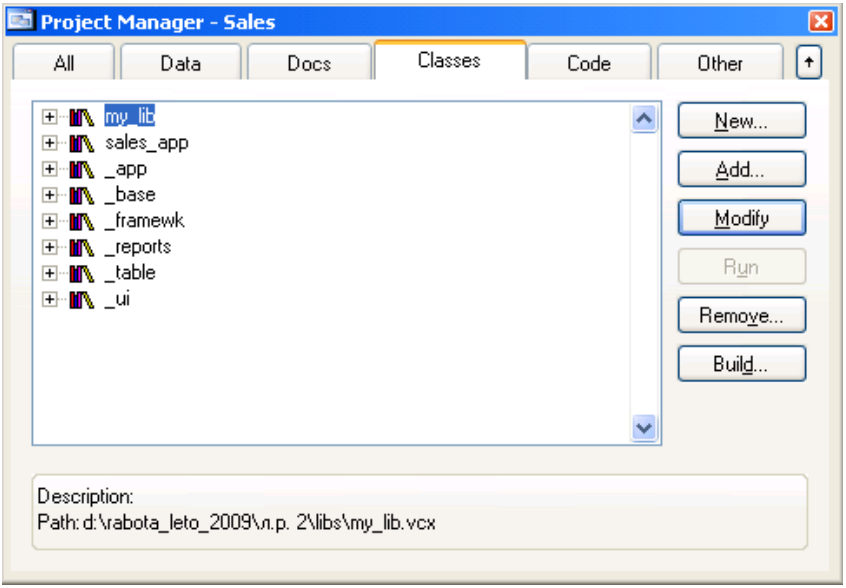

Рис. 1. Просмотр библиотек классов, подключенных в проект

В случае создания проекта мастером, базовая структура библиотек классов генерируется автоматически. Все библиотеки, имена которых начинаются с подчерка, являются системными и изменять их не рекомендуется. Так же не рекомендуется редактировать содержимое библиотеки с именем проекта и добавленным «_APP», т.к. в ней хранятся основные классы, используемые мастерами построений и базовый класс самого приложения.

Рекомендуется все дополнительные пользовательские классы хранить в отдельной библиотеке. Это позволит не только грамотно структурировать компоненты текущего проекта, но так же использовать эту библиотеку в других проектах, что положительно сказывается на сокращении времени разработки.

Обычно в отдельные библиотеки выносят классы, использующиеся не в одном проекте, причем возможны подразделения типов библиотек, т.е. отдельно библиотека с классами, ориентированными на обработку данных, отдельно с классами визуализации и т.д.

Следует помнить, что библиотека не может существовать хотя бы без одного класса в ней, а так же класс не может существовать отдельно от библиотеки.

Все визуальные классы в VFP, в обязательном порядке, должны базироваться на каком–либо стандартном классе, при этом базовым может быть как класс, уже определенный в системе, так и внешний класс, сохраненный в файле VCX. Изначально, пользовательский класс наследует все свойства и методы родительского класса, которые впоследствии могут редактироваться. Допускается добавление программистом собственных свойств и методов, но только в пользовательский класс.

Наиболее простым методом создания пользовательского класса является использование визуального построителя. Для создания нового класса необходимо перейти в менеджере проекта во вкладку **Classes** и нажать **New**, при этом появится диалоговое окно задания базовых параметров класса (рис. 2).

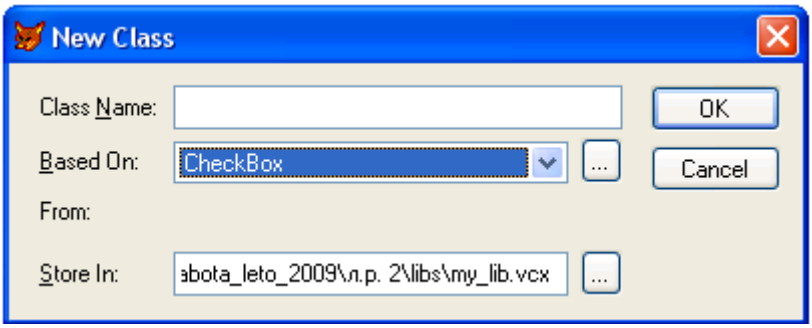

Рис. 2. Диалоговое окно создания класса

В появившемся диалоговом окне, в поле **Class Name** необходимо указать имя класса, которое будет в дальнейшем использоваться для доступа к объекту этого класса. При помощи ниспадающего списка в поле **Based On** указывается базовый класс. Используя элемент управления, расположенный правее списка можно выбрать произвольную библиотеку классов, расположенную на внешнем носителе. При использовании внешней библиотеки классов в поле **From** отображается полный путь к ней. В поле ввода **Store In** указывается путь к библиотеке, куда будет сохранен создаваемый класс.

После нажатия кнопки **OK**, созданная библиотека добавиться в проект, и откроется окно редактирования класса (рис. 3).

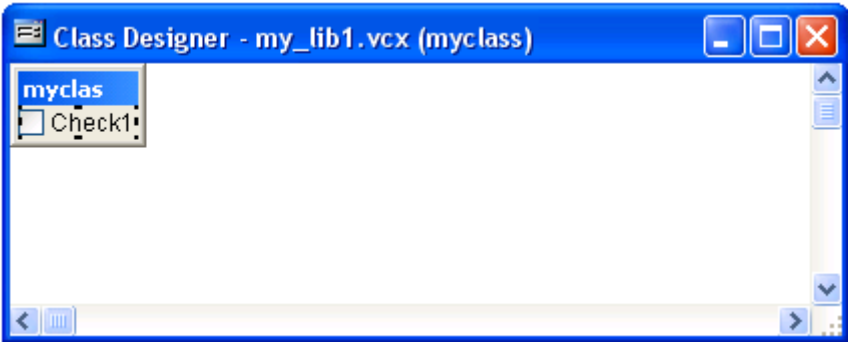

Рис. 3. Диалоговое окно редактирования класса

По сути, окно редактирования класса ничем не отличается от окна редактирования формы, только в данном случае в качестве формы выступает элемент, класс которого был выбран базовым. В окне **Properties** можно просмотреть и, при необходимости, изменить любые свойства и методы, унаследованные от базового класса.

По завершению редактирования окно следует закрыть, что приведет к автоматическому сохранению всех изменений в библиотеке класса. Для использования созданного класса на экранной форме достаточно открыть форму на редактирование и мышкой перетащить класс из вкладки **Classes** менеджера проекта на редактируемую форму. Однако технология Drag&Drop не всегда удобна, особенно если форма большая, а классов много. В окне **Form Controls** существует элемент *View Classes*, позволяющий выбрать библиотеку классов, элементы которой будут отображены в самом окне (рис. 4).

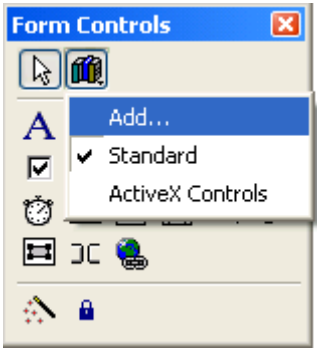

Рис. 4. Выбор отображаемых классов в окне Form Controls

В появившемся контекстном меню, при помощи пункта **Add…** можно выбрать произвольную библиотеку классов. При этом ее имя появится в списке, а так же изменится содержимое самого окна **Form Controls** (рис. 5). Работа с появившимися компонентами проводится так же, как и со стандартными системными компонентами при редактировании форм. Для возврата к стандартному виду следует выбрать пункт **Standard**.

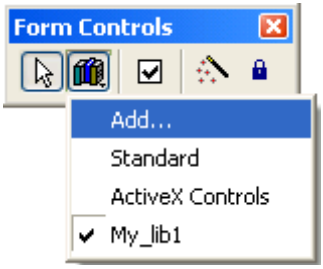

Рис. 5. Окно Form Controls, отображающее компоненты пользовательской библиотеки классов

Цветной выключатель

Рассмотрим более подробно пример создания пользовательского класса, основанного на простом выключателе. Предположим, что от стандартного объекта он должен отличаться цветом надписи, когда выключатель включен, надпись должна быть красной, а когда выключен – синей. Организовать такое отображение объекта на форме достаточно просто и в программном коде методов стандартного компонента, но что делать, если таких объектов будут десятки? Гораздо удобнее разработчику один раз создать новый класс, учитывающий отличия от стандартного, а затем просто тиражировать экземпляры данного класса на любой форме в произвольном количестве.

На первом шаге необходимо создать новый класс, базирующийся на стандартном классе *CheckBox*. Для этого необходимо перейти во вкладку **Classes** окна менеджера проекта и нажать кнопку **New**. При этом на экране появится диалоговое окно создания нового класса (рис. 6). Укажите в поле ввода новое имя для создаваемого класса (на рисунке это *MyClass*). В поле ввода **Store In** укажите имя библиотеки, в которой класс будет храниться (на рисунке указана новая библиотека *My_lib1.vcx*). В качестве базового класса в поле **Based On** выберите класс *CheckBox*. После указания всех необходимых данных и нажатия кнопки **OK** на экране должно появиться диалоговое окно редактирования класса (см. рис. 3).

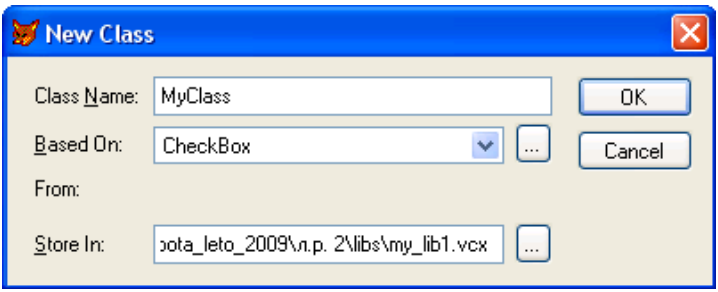

Рис. 6. Окно создания нового класса, базирующегося на стандартном классе CheckBox

Для просмотра свойств и методов созданного класса следует в окне редактирования класса щелкнуть правой кнопкой мыши на объекте и выбрать пункт **Properties** (рис. 7), при этом откроется окно свойств, в заголовке которого указывается имя библиотеки и имя класса, который редактируется в настоящий момент (рис. 8).

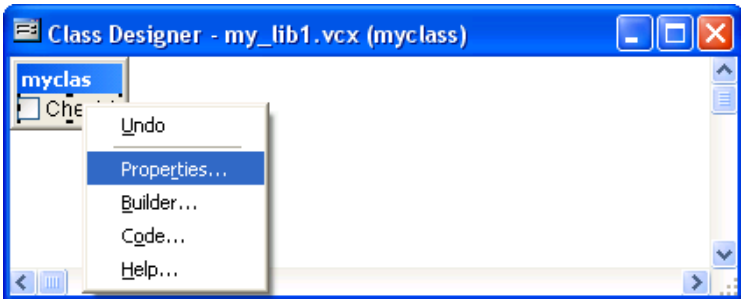

Рис. 7. Вызов окна свойств для редактируемого класса

В окне свойств доступны для редактирования все свойства и методы, унаследованные от базового класса. Кроме того, разработчик может добавлять свои свойства и методы, используя

команды пункта главного меню **Class | New property** и **Class | New metod**. Добавление новых свойств и методов к существующему классу полностью аналогично добавлению таковых к форме в редакторе экранных форм. Следует помнить, что независимо от присваиваемых имен, все пользовательские свойства и методы появляются в окне свойств в самом конце списка и их значения выделяются ярко красным цветом (хотя это и зависит от цветовой палитры, примененной в системе VFP).

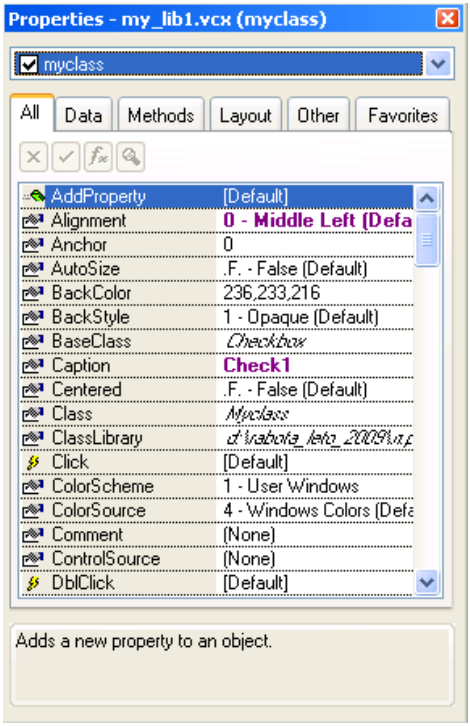

Рис. 8. Окно свойств пользовательского класса

Для организации изменения цвета надписи элемента необходимо прописать программный код, осуществляющий это.

Следует учесть, что программный код должен отрабатывать не только при изменении значения самого выключателя, но так же и при первом запуске формы, на котором этот выключатель будет расположен. Для этого следует отредактировать методы *Init* и *InteractiveChange* созданного класса. Для этого перейдите во вкладку **Methods** окна свойств и добавьте в указанные методы следующую пользовательскую программу

```
*если значение текущего элемента 0, т.е. выключено
*то устанавливаем цвет надписи на красный
*в противном случае – на синий

IF this.Value=0 then
        this.ForeColor=RGB(255,0,0)
ELSE
        this.ForeColor=RGB(0,0,255)
ENDIF

*перерисовываем элемент на экране

this.Refresh()
```

Обратите внимание, что для указания элемента используется служебное слово ***this***, т.к. разработчик не может заранее знать какое именно имя будет у элемента на форме. Кроме того, если таких элементов будет несколько, то и имена у них

будут разные и, соответственно, заранее невозможно организовать управление поведением объекта используя какое-либо конкретное имя. (Используя указатель ***this*** разработчик дает указание системе обрабатывать тот элемент, который является текущим с точки зрения системы).

Таким образом, в приведенном программном коде проверяется значение свойства *Value* текущего элемента (в нашем случае это сам выключатель). Если значение равно нулю (т.е. элемент выключен), то для надписи устанавливается красный цвет, иначе – синий. Для указания цвета надписи используется свойство *ForeColor*. Свойству присваивается не просто цвет, а значение цветовой палитры в формате RGB (Red, Green, Blue). Преобразование тройки чисел, определяющих интенсивность основных цветов, осуществляется функцией RGB(*nRedValue, nGreenValue, nBlueValue*). В скобках, через запятую, обязательно указываются интенсивности трех основных цветов! Последняя строка программного кода перерисовывает объект на экране. При этом обновление внешнего вида происходит только для данного объекта, а не для всей формы, что существенно ускоряет работу.

По завершению редактирования методов закройте окно редактирования класса. Затем создайте новую форму конструктором. В панели **Form Controls** выберите созданную библиотеку (см. рис. 4), при этом в панели должен появиться только один элемент. Добавьте два экземпляра этого элемента на форму и поменяйте у них свойство **Caption** на нечто разумное (рис. 9). Не обращайте внимание на то, что надпись в редакторе

форм отображается черным цветом, изменения проявятся после запуска формы.

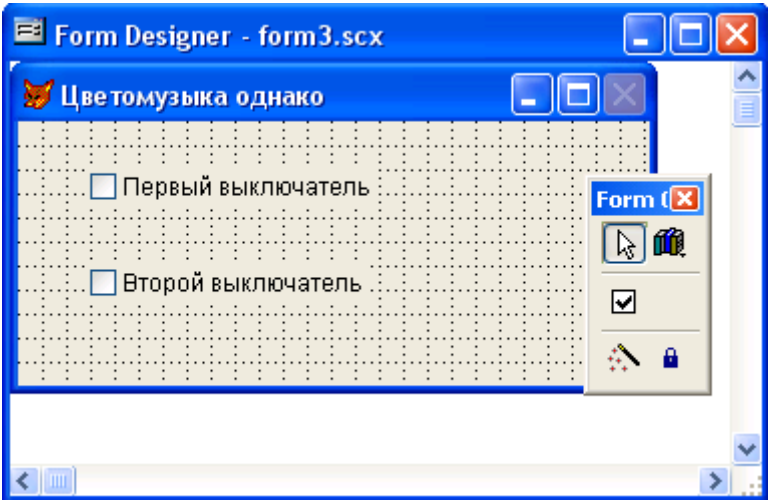

Рис. 9. Окно редактора форм с вынесенным пользовательским классом

Сохраните созданную форму и запустите ее на выполнение. Если все сделано правильно, то можно наблюдать на экране выключатели, цвета надписей которых зависят от их состояния (рис. 10).

Создайте еще две формы, на каждой из которых разместите по пять объектов созданных на базе нового класса. Запустите все три формы с новыми выключателями на исполнение и расположите их так, чтобы они не перекрывали друг друга. В произвольном порядке попереключайте выключатели на разных формах, и отметьте их поведение.

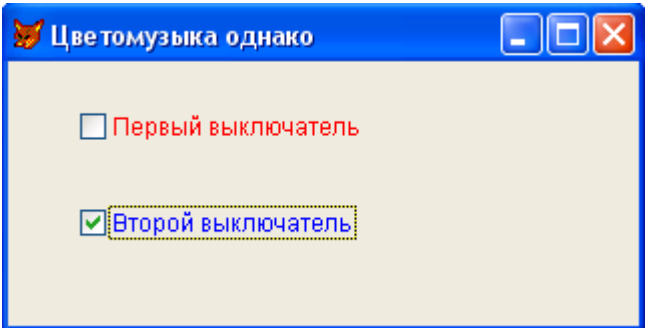

Рис. 10. Запущенная форма с «цветными» выключателями

Важно: Если форма, на которой расположен экземпляр пользовательского класса, запущена или открыта на редактирование, изменение самого класса не допускается. Для редактирования класса необходимо закрыть все формы, на которых располагаются объекты, созданные на базе пользовательского класса.

Классы, содержащие несколько объектов

Рассмотрим пример создания сложного пользовательского класса, содержащего не один объект, а несколько. Предположим, что разработчику понадобилось расположить на экранных формах цифровые таймеры, отсчитывающие время с указанной точность и предполагающие возможность перезапуска в произвольный момент. Объект должен предусматривать так же возможность задания стартового значения времени и шаг временных интервалов.

Задача достаточно простая, решается использованием на форме объектов типа надписи и таймера. Однако из условия видно, что количество таймеров величина неопределенная, их может быть произвольно число от нуля и до пределов фантазии разработчика. Поэтому целесообразно создать один раз пользовательский класс, объединяющий в себе надпись и таймер, а затем просто тиражировать его необходимое количество раз на экранных формах.

Вначале создайте новый пользовательский класс, выбрав в качестве базового класса *Container* (рис. 11). (Контейнер – это объект, содержащий в себе некоторые подобъекты). Обратите внимание, на то, что в одной библиотеке имена классов повторяться не могут. Если необходимо создать класс, имя которого уже используется, его необходимо сохранять в отдельную библиотеку.

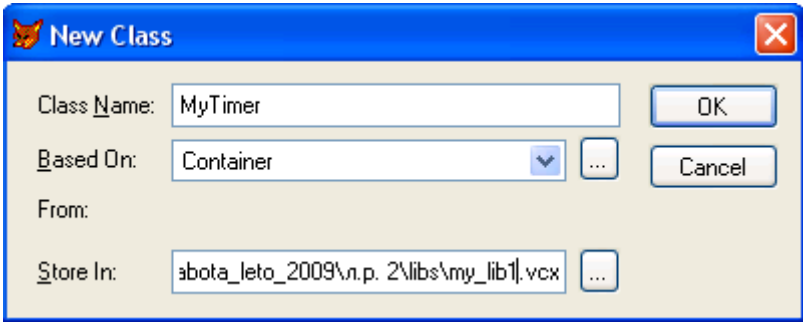

Рис. 11. Создание нового класса на базе контейнера

В появившемся окне редактирования класса (рис. 12) разработчик может расположить произвольные объекты внутри области контейнера. Допускается произвольно менять визуальную

область контейнера, для этого достаточно захватить его край мышкой и просто растянуть до требуемой величины. Рекомендуется в начале процесса редактирования увеличивать размер контейнера для удобного расположения входящих в него элементов, а в конце работы – сжать визуальную область до необходимого размера. Помните, на экранной форме контейнер будет занимать точно такой размер, какой определен при редактировании класса.

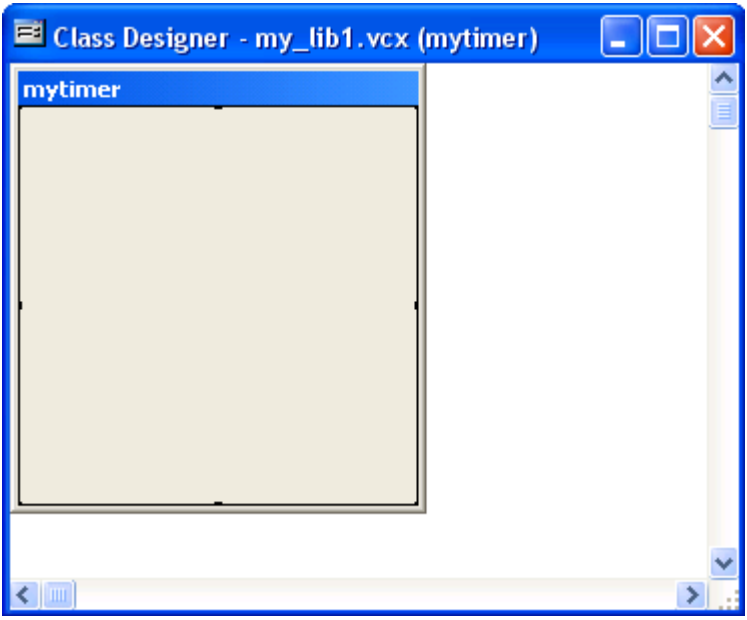

Рис. 12. Окно редактирования класса типа Контейнер

При помощи панели **Form Controls** расположите внутри контейнера объекты *Label* и *Timer*, как показано на рис. 13. К сожалению, границы надписи в режиме редактирования контейнера заметить практически невозможно, видны только

точки изменения ее размеров, когда надпись выделена. Поэтому не следует присваивать свойству *Caption* надписи пустое значение, иначе ее можно визуально просто потерять, хотя в окне свойств выбрать сам объект *Label* можно.

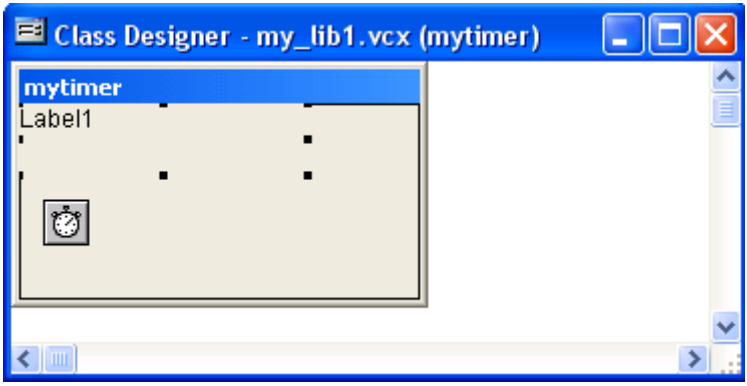

Рис. 13. Контейнер с размещенными объектами Label и Timer

Затем добавьте при помощи команды **Class | New property** два новых свойства к классу с именами *st_val* и *st_ep*, которые будут содержать в себе стартовое значение счетчика и его шаг соответственно. **Важно** задавать стартовые значения новым свойствам, которые определяют их типы по умолчанию. В нашем случае обязательно присвойте стартовые значения «**0.0**»

В методе *Init* создаваемого класса необходимо прописать присвоение стартовых параметров самого таймера:

```
*задание интервала таймера в миллисекундах
this.timer1.Interval=1000*this.st_ep
*присвоение надписи начального значения счетчика
*всего будет выведено 10 знаков, после запятой - 1
```

```
this.label1.Caption=STR(this.st_val,10,1)
*обновление контейнера на форме
this.Refresh()
```

Сам объект таймер в системе VFP работает так же, как и в остальных языках программирования, т.е. таймер отсчитывает указанное количество миллисекунд от нулевого значения, а затем вызывает собственное событие *Timer*. После этого начальное значение таймера сбрасывается и отсчет начинается заново. Для управления отображением счетчика следует в окне свойств, при помощи ниспадающего списка выбрать объект *Timer* (или просто выделить его мышкой в окне редактирования класса) и во вкладке *Methods* выбрать событие *Timer*, после чего, двойным щелчком мыши вызвать окно редактирования программного кода. В нем следует прописать поведение таймера при достижении им порогового значения:

```
*увеличиваем свойство класса, отвечающее за стартовое значение
*на величину, указанную в свойстве шага счетчика
this.Parent.st_val = this.Parent.st_val+this.Parent.st_ep
*выводим значение в надписи
this.Parent.label1.Caption=STR(this.Parent.st_val,10,1)
*обновляем контейнер
this.Parent.Refresh()
```

В данном программном коде, кроме указателя на текущий объект ***this,*** используется указатель на объект более верхнего уровня, являющийся родительским по отношению к текущему объекту – ***Parent***. В нашем случае родительским объектом в иерархии является контейнер, а надпись и таймер – его дочерние подобъекты. Поэтому, находясь в объекте *Timer*, обращение к контейнеру будет выглядеть как ***this.Parent***. А обращение к надписи из таймера – ***this.Parent.label1***.

В принципе, работа над счетчиком почти завершена. При запуске устанавливаются стартовые значения, в процессе работы по указанному таймингу значения изменяются. Однако, задача содержала в себе еще и требование на возможность сброса значения счетчика и установки нового шага.

Для этого следует добавить новый метод в созданный класс при помощи команды **Class | New method**. Добавьте новый метод с именем Reset. Далее следует описать действия класса при вызове этого метода. Дважды щелкните в окне свойств на имени созданного метода и пропишите следующий программный код:

```
*переустанавливаем шаг таймера
this.timer1.Interval=this.st_ep *1000
```

По окончанию редактирования нового класса не забудьте сжать видимую область контейнера до размеров надписи! Не пугайтесь, все элементы не вошедшие в видимую область все равно будут работать, а при последующем редактировании класса к ним можно легко добраться просто увеличив размер контейнера.

Убедитесь в полноте и правильности выполненных действий и закройте окно редактирования класса.

Затем создайте новую экранную форму и поместите на нее два объекта созданного класса. Для этого либо просто перетащите дважды класс на форму, либо подключите собственную библиотеку в окне **Form Controls** (см. рис. 4, см. рис. 5). Кроме того, разместите на этой же форме кнопку, которая должна будет отвечать за рестарт одного из счетчиков. В методе *Click* кнопки пропишите следующий программный код:

```
*установка нового стартового значения для первого счетчика
thisform.my1.st_val =0
*установка нового шага для первого счетчика
thisform.my1.st_ep = 0.5
*сброс первого счетчика
thisform.my1.reset()
```

Запустите форму на исполнение. Оба счетчика должны синхронно работать. После того, как значение счетчиков станут больше 10, нажмите кнопку сброса счетчика (Рестарт). Обратите внимание на поведение первого счетчика. Если все сделано правильно, то его значения должны начать отсчитываться с нуля с шагом 0.5 секунд (рис. 14).

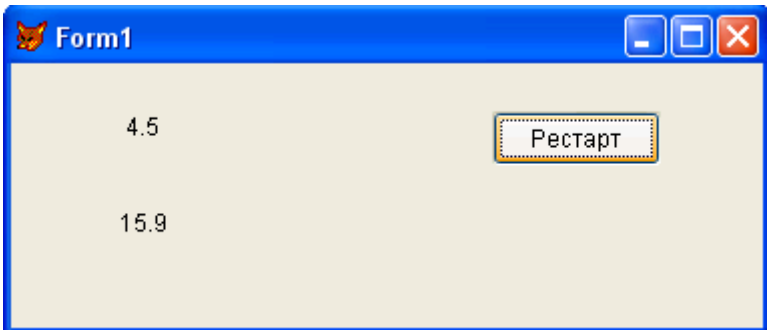

Рис. 14. Форма с запущенными счетчиками

Примечания:

1. Во вкладке **Methods** окна свойств отображаются как методы объекта, так и его события. Основное отличие в том, что методы необходимо явно вызывать, а события вызываются сами при некоторых внешних условиях.

Задания

1. Создайте пользовательский класс выключателя, меняющего цвет надписи на синий в выключенном состоянии и желтый – во включенном.
2. Создайте пользовательский класс, выводящий на форме текущую дату и время с точностью до секунд. Обновление значений должно происходить ежесекундно. Функция, возвращающая текущую дату и время в VFP, это DATETIME(). При создании класса рекомендуется использовать в качестве базового – контейнер.

Лабораторная работа № 12

Тема: Организация взаимодействия с приложениями MS Office

Цель: Изучить способы передачи данных и команд в приложения MS Office

Возможностей по организации работы на локальном месте и отображению данных системой VFP много, но их не всегда достаточно. Иногда возникает необходимость использования сторонних программ, в которые можно было бы передать данные из пользовательского приложения, а так же выполнить некоторые команды, доступные в сторонних программах. Для организации взаимодействия программ VFP и сторонних приложений удобно использовать технологию IntelliSense. Она позволяет получить доступ к свойствам и методам, поддерживаемым сторонним приложением, в режиме разработки из среды VFP.

IntelliSense является системной технологией, ее могут использовать различные среды разработок. Для подключения технологии в системе VFP необходимо выполнить команду главного меню Tools | IntelliSense Manager, при этом откроется диалоговое окно управления настройками IntelliSense в системе VFP (рис.1).

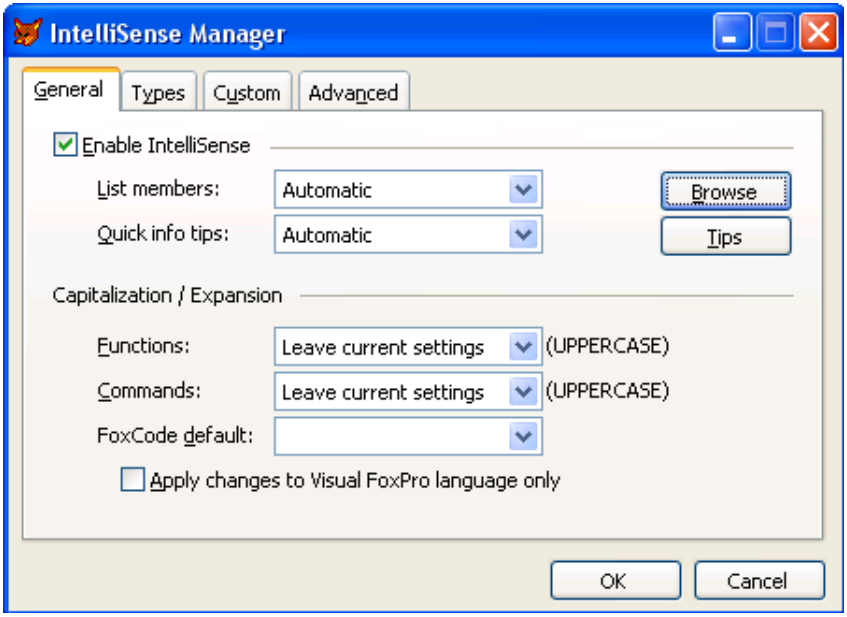

Рис. 1. Диалоговое окно управления настройками технологии IntelliSense

Вкладка **General** предназначена для настройки общих свойств технологии IntelliSense.

Выключатель **Enable IntelliSense** определяет доступность интерактивной подсказки по свойствам объектов и методов стороннего приложения в режиме разработки программ VFP. При включенном режиме, в окне создания программного кода, будут автоматически раскрываться ниспадающие списки с доступными командами объектов, после введения символа–разделителя (в системе VFP символом разделителем является *точка*).

Блок **Capitalization/Expansion** используется для определения синтаксиса команд и функций. По умолчанию все

вводимые команды преобразовываются в строчные буквы. Выключатель **Apply changes to Visual FoxPro language only** определяет, будут ли использованы настройки ко всем командам или только к встроенным командам языка VFP.

Кнопки **Browse** и **Tips** предназначены для редактирования таблицы свойств IntelliSense (лучше не трогать!).

Во вкладке **Types** отображаются все используемые типы переменных и стандартные классы системы VFP (рис.2).

Выключатели в левой части списка определяют будет ли поддерживаться выбранный тип/класс технологией IntelliSense.

В нижней части диалогового окна расположен набор кнопок, предоставляющий возможности проведения основных операций с объектами IntelliSense.

Edit – позволяет просмотреть и отредактировать некоторые свойства выбранного типа/класса.

Type Libraries… – вызывает интерактивное окно управления типами объектов, зарегистрированных в операционной системе (рис. 3).

Classes… – позволяет подключать внешние библиотеки классов (файлы .VCL).

Web Services… – предоставляет доступ к XML Web–сервисам.

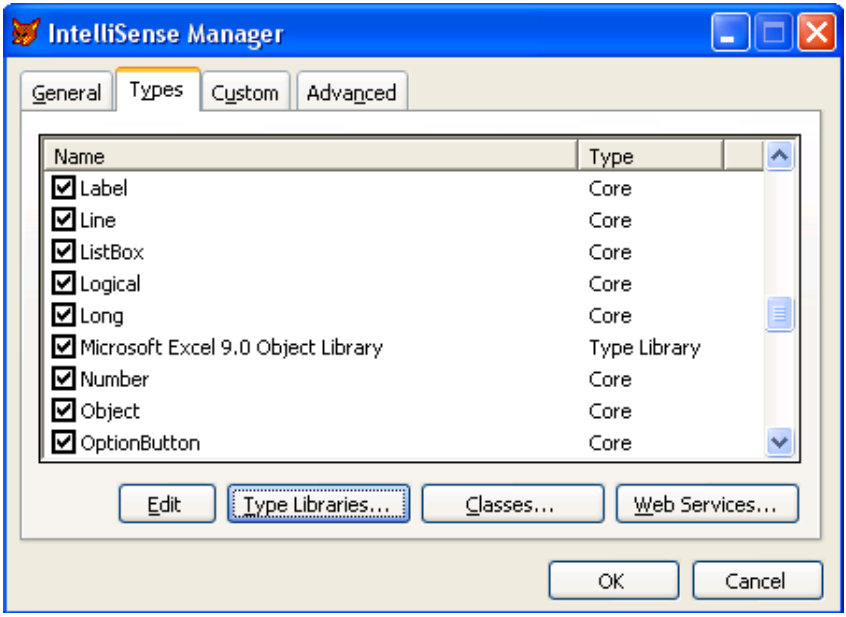

Рис. 2. Вкладка Types окна IntelliSense Manager

Диалоговое окно управления типами объектов, зарегистрированных в операционной системе, предназначено для подключения стандартных объектов в технологию IntelliSense системы VFP.

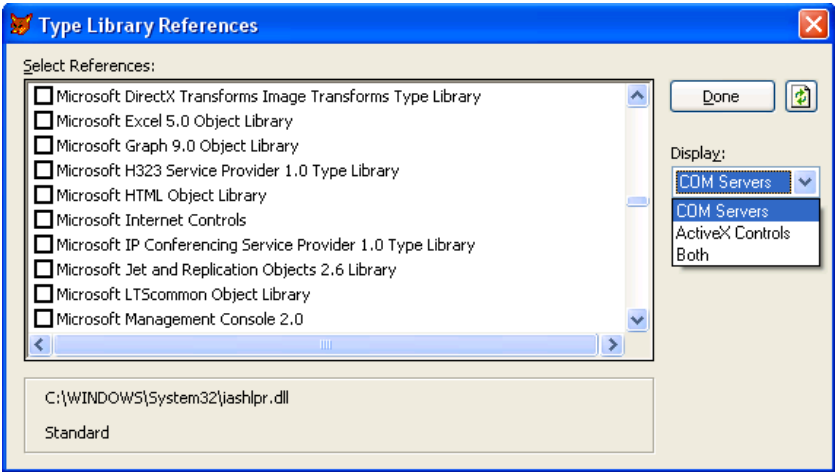

Рис. 3. Диалоговое окно управления подключением внешних объектов

В левой части окна расположен список всех объектов, зарегистрированных в операционной системе. Столбец выключателей определяет объекты, которые будут добавлены в список стандартных типов. Уже добавленные объекты из общего списка исключаются и добавляются к списку поддерживаемых типов
(см. рис. 2).

Ниспадающий список предназначен для управления отображением типов объектов.

COM Servers – при выборе будут отображены только объекты, поддерживающие технологию COM.

ActiveX Controls – при выборе отображаются только зарегистрированных ActiveX компоненты.

Both – отображаются все возможные объекты.

Кнопка **Done** завершает работу диалогового окна и перемещает выбранные объекты в список стандартных типов.

Вкладки **Custom** и **Advanced** используются при добавлении пользовательского типа объектов IntelliSense.

Примечание: взаимодействие с внешней программой можно осуществить, не зависимо от того, подключен объект к технологии IntelliSense или нет. Однако лишняя подсказка при программировании еще никому не мешала, поэтому рекомендуется подключать все используемые объекты.

Запуск MS Excel средствами VFP

Первым шагом при работе с внешним приложением является подключение его объекта к технологии IntelliSense системы VFP. Для подключения объекта MS Excel необходимо в диалоговом окне управления типами объектов (см. рис. 3) найти строку **Microsoft Excel 9.0 Object Library** и подключить ее к стандартным типам.

Следует помнить о том, что при запуске стороннего приложения, с точки зрения VFP, оно представляет собой объект и для дальнейшей работы необходимо присвоить ссылку на этот объект произвольной переменной. Соблюдая венгерскую нотацию в именовании переменных, рекомендуется начинать их имена с символа «**o**», например, для объекта Excel целесообразно использовать имя переменной oExcel.

Синтаксис команды, создающей объект произвольного класса в системе VFP следующий:

CREATEOBJECT(cClassName [, eParameter1, eParameter2, ...])

где, параметр *cClassName* является текстовым именем класса, описывающего создаваемый объект. После указания имени класса, через запятую допускается указывать произвольное количество параметров, с какими будет создан объект.

Таким образом, команда создания объекта типа приложения MS Excel в синтаксисе VFP будет выглядеть следующим образом:

```
oExcel=CREATEOBJECT('Excel.Application')
```

Дальнейшую работу с приложением следует осуществлять посредством явного указания имени переменной, ссылающейся на объект приложения, например, команда *oExcel.Quit* закроет Excel. Не забывайте, что созданный объект является «голым» приложением. В нем нет ни рабочих книг, ни, тем более, рабочих листов. Их придется создавать или открывать методами созданного объекта.

Рабочая книга Excel является подобъектом самого объекта Excel.Application. Для возможности дальнейшей работы объект типа «Рабочая книга» должен быть присвоен некоторой переменной, например:

```
oMyBook=oExcel.Workbooks.Add()
```

данная команда создает новую рабочую книгу и присваивает ее переменной *oMyBook*. Если необходимо открыть

уже существующую книгу, следует использовать команду Open, а не Add.

```
oMyBook=oExcel.Workbooks.Open('C:\MyTestBook.xls')
```

Примечание: обратите внимание, что при написании команд, после символа разделителя система автоматически раскрывает список всех доступных свойств и методов (рис. 4)

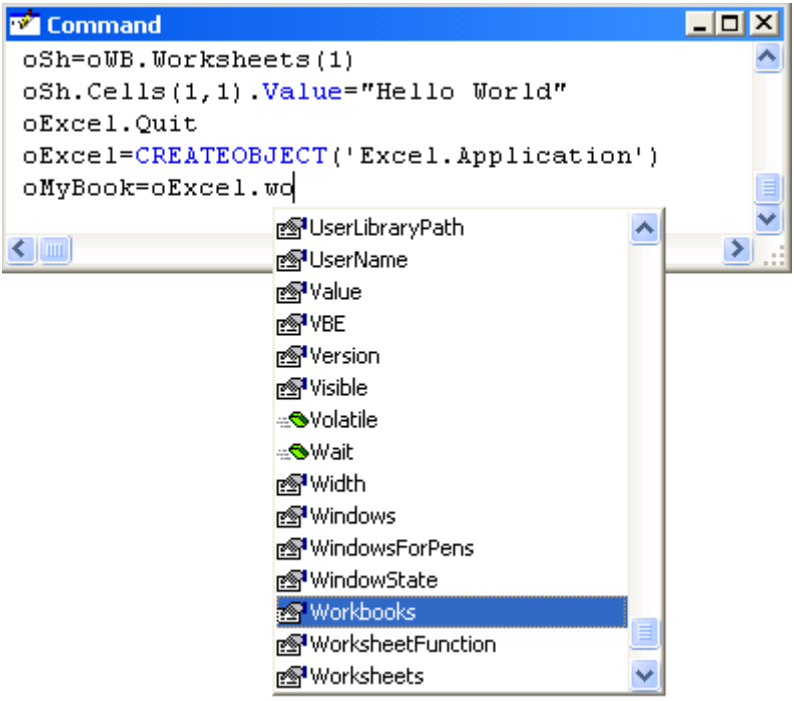

Рис. 4 Ниспадающий список доступных команд

Если создается новая рабочая книга, то ее параметры такие же, как и у книги, созданной непосредственно в Excel. В

большинстве случаев эта книга пуста и содержит в себе три чистых рабочих листа, причем активным является первый лист.

Для возможности непосредственной работы с данными, следует создать переменную, со ссылкой на объект типа рабочего листа (ячейки в Excel являются подобъектами именно рабочего листа, а не книги). Это можно сделать следующей командой:

oMySheet=oMyBook.Worksheets(1)

число в скобках указывает на индекс рабочего листа в книге. Т.к. нумерация листов начинается с 1, то указанная команда присваивает переменной oMySheet ссылку на первый рабочий лист.

Для непосредственного обращения к ячейке следует использовать свойство рабочего листа Cells, например:

OMySheet.Cells(1,1).Value = "Hello World"

Числа в скобках, разделенные запятой, являются координатами ячейки в формате RC. Первое число задает номер строки, второе – номер столбца. Вместо чисел допускается указывать имена целочисленных переменных, определенных в VFP.

После совершения необходимых операций с данными на рабочем листе приложение Excel следует сделать видимым, что производится командой *oExcel.Visible=.T.*

Таким образом, следующий программный код создает новую книгу Excel и передает в нее текстовую строку:

```
*создаем объект приложения
oExcel=CREATEOBJECT('Excel.Application')
```

```
*создаем в приложении новую книгу
oMyBook=oExcel.Workbooks.Add()
*получаем ссылку на первый рабочий лист
oMySheet=oMyBook.Worksheets(1)
*заносим в ячейку A1 текстовую надпись Hello World
oMySheet.Cells(1,1).Value="Hello World"
*делаем приложение видимым на экране
oExcel.Visible=.T.
```

Программный код можно прописать либо в командном окне (при этом каждая команда немедленно выполняется и доступна интерактивная подсказка синтаксиса), либо в программном модуле, например в событии нажатия кнопки на форме (при этом команды будут выполнены только после запуска модуля и, к сожалению, интерактивная помощь не доступна, т.к. VFP просто не знает о чем подсказывать). Рекомендуется все малознакомые команды вначале пробовать в командном окне, а только потом прописывать их в программных модулях (только без фанатизма, прописать цикл в командном окне просто не получится).

Передача данных из таблиц VFP в MS Excel

Часто возникает необходимость передачи данных из таблиц VFP на рабочий лист Excel. Это может быть вызвано необходимостью либо расширенного анализа данных средствами табличного процессора, либо формирования отчета в специфической форме, либо представлении пользователю

возможности изменения данных в отчете без их изменений в таблицах VFP.

Для передачи данных необходимы две вещи. Во–первых источник, откуда данные будут получаться. В этой роли может выступать таблица, представление данных или курсор. Во–вторых приемник, куда данные будут помещаться. Это может быть только рабочий лист Excel. Следовательно, перед началом передачи необходимо позаботиться о создании объекта Excel, открытии/создании рабочей книги и получения ссылки на рабочий лист. Кроме того, следует открыть или создать источник данных в системе VFP.

Для организации перебора записей в источниках данных VFP существует множество вариантов. Одним из наиболее простых является использование команды SCAN. Синтаксис команды приведен ниже, все параметры, приведенные в квадратных скобках, необязательны и могут пропускаться.

SCAN [NOOPTIMIZE] [Scope] [FOR lExpression1] [WHILE lExpression2]

[Commands]

[LOOP]

[EXIT]

ENDSCAN

[NOOPTIMIZE] – включение этого слова отключает оптимизацию технологии Rushmore.

[Scope] – позволяет указать диапазон перебираемых записей. Допустимые значения – ALL (перебираются все записи), NEXT nRecords (перебираются следующие nRecords записей от текущей), RECORD nRecordNumber (перебираются только записи с указанным номером), REST (перебираются все записи от текущей до конца таблицы).

[FOR lExpression1] – позволяет указывать условие фильтрования записей. Цикл SCAN будет перебирать только те записи, которые удовлетворяют указанному условию lExpression1.

[WHILE lExpression2] – позволяет указать условие выполнения цикла lExpression2. Пока оно истинно цикл будет выполняться.

[Commands] – блок команд, которые будут выполняться на каждом шаге цикла.

[LOOP] – команда, передающая управление в начало цикла. Может использоваться в произвольном месте цикла.

[EXIT] – команда, передающая управление за конец цикла. Может использоваться в произвольном месте цикла. Аналог экстренного завершения.

Примечание: не следует считать, что действия служебных слов FOR и WHILE одинаковы. FOR накладывает фильтр на записи таблицы. В разговорной интерпретации это звучит как «Выполнить для всех записей у которых поле ЦЕНА больше 20». WHILE задает общее логическое условие, т.е. «Пока солнце зеленое выполнять».

Ниже приведен программный код, организующий формирование на рабочем листе Excel списка товаров и их цен из таблицы Goods. Для перебора записей таблицы используется цикл SCAN. Данные помещаются в новую рабочую книгу с первой строки в две колонки. По завершению переноса ширина колонок рабочего листа форматируется по содержимому.

```
*создаем объект приложения
oExcel=CREATEOBJECT('Excel.Application')
*создаем в приложении новую книгу
oMyBook=oExcel.Workbooks.Add()
*получаем ссылку на первый рабочий лист
oMySheet=oMyBook.Worksheets(1)
*выбираем источник данных, в нашем случае - таблицу goods
SELECT goods
*перемещаем указатель в таблице в начало
GO top
*переменная i будет определять номер строкина рабочем листе
Excel
i=1
*цикл SCAN перебирает все записи в таблице
SCAN
        *помещаем в i-ю строку в первый столбец название
        товара
        oMySheet.cells(i,1).value=goods.cnmgoods
```

```
        *помещаем в i-ю строку во второй столбец цену товара
        oMySheet.cells(i,2).value=goods.nunitprice
        *переходим к следующей строке
        i=i+1
ENDSCAN
*выделяем весь лист Excel
oMySheet.cells.select
*подгоняем ширину столбцов по содержимому
oMySheet.cells.EntireColumn.AutoFit
*переводим курсор на ячейку A1
oMySheet.cells(1,1).select
*делаем приложение видимым на экране
oExcel.Visible=.T.
```

Манипуляции с данными на рабочем листе Excel из приложения VFP

После переноса данных на рабочий лист иногда возникает необходимость их корректировки или формирования новых данных непосредственно на самом листе. Существует два варианта. Данные могут либо предварительно просчитываться в приложении и помещаться на лист в явном виде, либо просчитываться на самом рабочем листе.

Организация манипулирования данными на рабочем листе аналогична организации переноса данных из таблицы на лист, за тем исключением, что в качестве источника выступает сам рабочий лист. Ниже приведен фрагмент программного кода,

который вычисляет 10-типроцентную величину от столбца цен. (Это вполне может быть праздничная скидка).

Для корректной работы предполагается, что данные на рабочем листе сформированы предыдущим программным кодом. Вполне допустимо слияние кодов, т.е. текущий можно просто добавить в конец предыдущего и выполнить как единое целое.

```
*переменная i будет определять номер строки на рабочем листе Excel
i=1
*цикл будет выполняться пока в очередной строке не встретится пустое значение
DO while oMySheet.cells(i,1).value<>" "
*помещаем в 3-й столбец 10% от второго столбца (в нашем случае от цены)
oMySheet.cells(i,3).value=oMySheet.cells(i,2).value*0.1
*переходим к следующей строке
i=i+1
ENDDO
```

При изучении рабочего листа Excel, можно заметить, что в 3-й колонке помещены непосредственные данные. Иногда это не очень удобно, т.к. при изменении исходных данных второго столбца расчетные данные не изменятся. В таком случае более выгодным является не расчет явных значений, а помещение на рабочий лист формул расчета. Соответственно нагрузка по

перерасчету значений ложится целиком на Excel, а он это делает автоматически по умолчанию.

Формулы, помещаемые на рабочий лист, должны соответствовать синтаксису Excel, в них не допускаются лишние пробелы, именование ячеек должно быть стандартным (A1, а не (1,1)). При помещении на рабочий лист формул следует использовать свойство ячейки не *VALUE*, а *FORMULA*. Переделаем предыдущий пример таким образом, чтобы на рабочем листе кроме списка товаров и их цен формировался столбец, в котором автоматически перерасчитывались бы 10-типроцентные скидки при любых изменениях данных в самом Excel. Для этого следует строку oMySheet.cells(i,3).value=oMySheet.cells(i,2).value*0.1 заменить на блок команд, формирующих формулу расчета, а именно:

```
s="=B"+ALLTRIM(STR(i))+"*0.1"
oMySheet.cells(i,3).Formula = s
```

Переменная S содержит в себе текстовый эквивалент формулы, который затем помещается на рабочий лист. Ее значение формируется как конкатенация (слияние) трех текстовых подстрок "=B", ALLTRIM(STR(i)) и "*0.1". Функция STR преобразует числовую переменную I в текстовый вид (а в ней как раз и хранится номер строки). Функция ALLTRIM обрезает от полученного значения все внешние пробелы. Таким образом, если переменная I будет равна 1, переменная S будет равна «=B1*0.1», если I=2, S=«=B2*0.1» и т.д.

В итоговом виде программный код будет выглядеть следующим образом:

```
*переменная i будет определять номер строкина рабочем листе Excel
i=1
*цикл будет выполняться пока в очередной строке не встретится пустое значение
DO while oMySheet.cells(i,1).value<>" "
*фромируем текстовый эквивалент формулы
s="=B"+ALLTRIM(STR(i))+"*0.1"
*помещаем его в 3-й столбец рабочего листа
oMySheet.cells(i,3).formula = s
*переходим к следующей строке
i=i+1
ENDDO
```

Теперь при изменении значений цен во втором столбце значения скидок в третьем столбце будут рассчитываться автоматически. Так же, при рассмотрении значений ячеек третьего столбца видно, что в них находятся не конкретные числа, а формулы, записанные по правилам Excel.

Передача данных из таблиц VFP в MS Word

Существует целый ряд отчетов, в которых не требуется дополнительных вычислений, но выполненных в формате,

который нецелесообразно или невозможно представить стандартным построителем отчетов VFP. Кроме того, стандартные отчеты можно только посмотреть и распечатать, а редактировать никак нельзя. В таких случаях удобно использовать возможности стандартного текстового процессора Microsoft Word.

Работа с текстовым процессором мало чем отличается от работы с табличным процессором. Вначале следует подключить **Microsoft Word 9.0 Object Library** к списку стандартных типов в настройках технологии IntelliSense. Как это сделать рассмотрено выше, на примере библиотеки Excel.

Затем следует создать объект типа приложения, но в качестве класса указать приложение Word. Это выполняется командой *oWord=CREATEOBJECT('Word.Application').* В отличии от Excel, с его развитой структурой подобъектов, в Word достаточно создать новый документ и уже можно с ним работать. Создание нового документа осуществляется методом Add() подобъекта Documents.

Операции с контентом (содержимым документа) в Word осуществляются посредством обращения к подобъекту приложения *Selection*. Поэтому любые операции будут выглядеть как *oWord.Selection.<необходимые действия>*. Запись текстовой строки осуществляется методом *TypeText()*. К сожалению, структурной единицей текста верхнего уровня в Word является не строка, а абзац. Для принудительного перехода к новой строке в документе введенный текст необходимо объявить параграфом. При этом к параграфу применятся настройки, объявленные по умолчанию (отступы, выравнивания и т.п.). Ниже приведен

простейший пример программного кода, создающего документ Word и записывающий в него текстовую строку:

```
*создаем объект типа приложения Ворда
oWord=CREATEOBJECT('Word.Application')
*создаем новый документ
oWord.Documents.Add()
*записываем в документ строку текста
oWord.Selection.TypeText("Hello World")
*объявляем ее как параграф
oWord.Selection.TypeParagraph
*делаем Ворд видимым
oWord.visible=.T.
```

Кроме текстовых абзацев документ Word может содержать в себе более сложные объекты. Часто используемым подобъектом текста является таблица. Для вставки новой таблицы в документ следует использовать метод Add() объекта Tables. Этот метод требует указания обязательных параметров, а именно: диапазона (туманное понятие, рекомендуется указывать текущий диапазон), количества строк и столбцов в таблице. На самом деле можно указывать еще множество параметров создаваемой таблицы, но все они являются необязательными и рассматриваться в рамках данной работы не будут. В последствии к таблице можно обращаться по ее индексу, например *oWord.Selection.Tables(1).*

Таблица в Word является двумерным объектом, состоящим из более простых подобъектов типа ячейки. Соответственно, каждая ячейка имеет две координаты, номер строки и номер столбца. Таким образом, обращение к ячейке третьей строки пятого столбца выглядит как *oWord.Selection.Tables(1). Cell(3,5).*

Ниже приведен пример программного кода передающего данные из таблицы товаров в текстовый процессор. Набор данных аналогичен примеру с Excel. Оператор WITH … ENDWITH использован сугубо для сокращения размера программного кода. Более подробно с его работой можно ознакомиться в системе помощи VFP.

```
*создаем объект типа приложения Ворда
oWord=CREATEOBJECT('Word.application')
*создаем новый документ
oWord.Documents.Add()
*записываем в документ его заголовок (строку текста)
oWord.Selection.TypeText("Список товаров")
*объявляем ее как параграф
oWord.Selection.TypeParagraph
*две строки пропуска для красоты
oWord.Selection.TypeParagraph
oWord.Selection.TypeParagraph
SELECT goods
GO top
*добавляем в документ таблицу
```

```
*первый параметр - в качестве диапазона указываем текущий
*второй параметр - количество строк указываем равным
количеству записей в таблице GOODS
*третий параметр - количество столбцов равно 2
oWord.Selection.Tables.Add(oWord.Selection.Range,RECCOUNT(),2)
*i - переменная с номером строки таблицы в Ворде
i=1
*перебираем все записи в GOODS
SCAN
	WITH oWord.Selection
		*в таблице с индексом 1 выбираем ячейку в первом
столбце строки i
		.Tables(1).Cell(i,1).Select()
		*в выбранную ячейку записываем название товара
		.TypeText(goods.cnmgoods)
		*во второй столбец строки заносим цену товара
		.Tables(1).Cell(i,2).Select()
		.TypeText(ALLTRIM(STR(nunitprice)))
	ENDWITH
	*переходим к следующей строке
	i=i+1
ENDSCAN
*для второго столбца таблицы указываем авторазмер
oWord.Selection.Tables(1).Columns(2).autofit()
*делаем Ворд видимым
oWord.visible=.T.
```

Примечания:

1. Получение данных из офисных продуктов осуществляется аналогичными действиями, только в программном коде источником данных следует прописывать офисное приложение.
2. Синтаксис команд VFP для работы с внешним приложением практически повторяет синтаксис VBA в офисных продуктах. В случае возникновения затруднений рекомендуется записывать макросы в приложениях MS Office и смотреть, какими же командами выполняется нужное действие. После небольшой доработки напильником можно получить вполне рабочий код в формате VFP.
3. Действие практически любой команды следует проверять вначале в командном окне VFP, а только затем включать в программный код. При этом не следует забывать, что для выполнения некоторых команд надо совершить предварительные действия, например, прежде чем пересылать значение в ячейку Excel, было бы неплохо предварительно создать в том же командном окне само приложение, рабочую книгу в нем и получить ссылку на рабочий лист.
4. Описанный метод можно применять к любым объектам, зарегистрированным в операционной системе. Набор команд для работы с этими объектами необходимо изучать/доставать своими силами. К счастью, для

продуктов MS Office набор команд выбран на основе интуитивно понятного и простого языка VBA. Для остальных объектов – интернет Вам в помощь.

5. Не забывайте, что для большинства пользователей, использование знакомых программных продуктов значительно повышает юзабилити Вашего приложения. Перереализовывать всем известные интерфейсы текстового процессора (Word) и табличного редактора (Excel) слишком трудоемко. Гораздо более простым решением является использование того, что уже сделано и легко доступно.
6. Все приведенные примеры работают в среде MS Office 2000/2003. Для 2007 версии работоспособность не гарантируется.

Задания

1. Передайте в Word полный список менеджеров, зарегистрированных в базе данных.
2. Передайте в Excel данные об именах и телефонах заказчиков. Если телефонов у одного заказчика несколько, его имя должно дублироваться на рабочем листе.
3. Сформируйте в текстовом процессоре Word прайс–лист товаров. Документ должен содержать заголовок в виде отдельной строки текста, а также таблицу из трех колонок. В первой должны располагаться названия

товаров, во второй – цена, в третьей – праздничная 15-типроцентная скидка.

4. Для указанного номера заказа сформируйте в табличном процессоре Excel бланк заказа. Документ должен содержать имя заказчика, дату и номер заказа. Ниже в табличном виде должна быть представлена информация о товарах в заказе, а именно название товара, цена за единицу, заказанное количество. Кроме того, в виде формул на рабочем листе должен быть организован столбец, в котором по каждой позиции товаров вычислялась бы итоговая сумма (цена за единицу умноженная на количество). В самом низу, так же в виде формулы Excel, должна подсчитываться общая сумма заказа.

Список литературы

1. Информационные системы: Учебное пособие для студентов вузов по специальности «Информационные системы в экономике» / под ред. Волкова В.Н. – СПб.: изд – во СПБГТУ, 1998.
2. Каратыгин С.А., Тихонов А.Ф., Тихонова Л.Н. «Visual FoxPro 6.0» – М.: ЗАО «Издательство БИНОМ», 2000.
3. Ложе И. Информационные системы. Методы и средства. – М.: изд-во «Мир», 1979.
4. Тихонов А.Ф., Тихонова Л.Н. «Visual FoxPro 5.0 (без проблем!)» – М.: Восточная Книжная Компания, 1997.
5. Методические указания для выполнению лабораторных работ по Visual FoxPro 9.0. / сост. Л.К. Скодорова, А.А. Ляху. Рыбница, 2010.
6. http://referat.ru/pub/item/23122
7. http://www.bankreferatov.ru
8. http://referat.ru/pub/item/15853
9. http://yurkovs.narod.ru/Ekolek/Tema5.htm
10. http://www.studzona.com/referats/view/13697
11. http://www.studzona.com/referats/view/13697
12. http://www.bntu.info/referat/work_6450.html
13. http://www.5ballov.ru/referats/preview/38041
14. http://www.aup.ru/books/m67/4.htm

Учебное издание

Visual FoxPro 9.0 SP1

Часть II

Лабораторный практикум

Учебное пособие

Формат А4. Уч.-изд., 6,4 п.л.

Тираж – 100 экз.