Генератор отчетов FastReport© v2.4

Руководство разработчика

Содержание

Введение 5
Введение 6
Этапы построения отчета 7
Данные 7
Параметры 8
Форма 8
Обработка 9
Готовый отчет 10
Построение простейшего отчета 11

Ядро FastReport 14
Палитра компонентов Delphi 15
Закладка "FastReport" 15
Закладка "FR Tools" 16
Компонент TfrReport 18
Компонент TfrDBDataSet 21
Компонент TfrUserDataSet 22
Объекты FastReport 23
Объект «Текст» 24
Объект «Секция» (Бэнд) 27
Объект «Рисунок» 29
Объект «Вложенный отчет» (SubReport) 29
Объект «Линия» 30
Объект «CheckBox» 30
Объект «RichText» 30
Объект «OLE» 31
Объект «Диаграмма» 32
Объект «Фигура» 34
Объект «Штрихкод» 34
Объект RichText 2.0 34
Построение отчетов 36
Бэнды в FastReport 37
Отчет с одним уровнем данных (список) 41
Отчет с двумя уровнями 41
Отчет с тремя уровнями 41
Cross-tab отчет 41
Динамические отчеты 42
Разрываемые бэнды 43
Многоколоночный отчет 44
Отчет с титульным листом 44
Вложенный отчет 45
Master-Detail-Detail отчет 45
Композитный (составной) отчет 45
Отчет с BLOb-полями 46
Отчет без бэндов 46
Отчет с группами 46
Отчет с диаграммами 46

Дизайнер 48
Дизайнер 49

Клавиши управления ... 50
Управление мышью ... 50
Параметры отчета ... 51
Параметры страницы ... 52
Опции дизайнера ... 55
Инспектор объектов ... 57
Диалог вставки полей БД в отчет ... 58
Словарь данных ... 59
Конструктор выражений ... 64
Диалог вставки поля БД ... 64
Диалог вставки переменной ... 64
Диалог вставки функции ... 65
Панели инструментов ... 67
Панель инструментов "Стандартная" ... 67
Панель инструментов "Текст" ... 68
Панель инструментов "Прямоугольник" ... 69
Панель инструментов "Выравнивание" ... 70

Построение автономных отчетов ... 71
Введение ... 72
Диалоговые формы ... 73
Элементы управления ... 74
Label ... 75
Edit ... 75
Memo ... 76
Button ... 76
CheckBox ... 77
RadioButton ... 77
ListBox ... 77
ComboBox ... 78
Диалоговая форма ... 79
Передача информации в отчет ... 80
Компоненты доступа к данным ... 81
Описание компонентов ... 82
TfrBDELookupComboBox ... 83
TfrBDETable ... 83
TfrBDEQuery ... 86
TfrBDEDataBase ... 87
Построение отчетов ... 89
Простой отчет типа "Список" ... 89
Отчет с запросом параметров ... 90
Компонент TfrDataStorage ... 92
Создание базы данных ... 93
Создание таблицы ... 93
Создание запроса ... 94
Редактор полей ... 95
Создание lookup-поля ... 96
Редактор параметров запроса ... 97
Связывание данных ... 97
Диалог ввода значений параметров ... 98
Дизайнер диалога ... 98
Языковые средства ... 101

Скрипты и объекты ... 102
Написание кода ... 102
Модификация объектов ... 105
Описание встроенных процедур и функций ... 106
Свойства и методы объектов ... 109
Применение ... 119

Программирование ... 120
Обработчики событий ... 121
Прочие события объекта TfrReport ... 121
Переменные ... 123
Расширение функциональных возможностей FastReport ... 125
Создание собственных окон предварительного просмотра ... 125
Расширение списка функций ... 126
Разработка собственных фильтров экспорта ... 128
Написание дополнительных мастеров ... 130
Разработка собственных визуальных компонент ... 131
Класс TfrObject ... 131
Класс TfrView ... 134
Класс TfrControl ... 140
Класс TfrStretcheable ... 140
Подключение новых компонент к FastReport ... 141

Примеры отчетов ... 142
Примеры отчетов ... 143
Включение в отчет графиков и диаграмм ... 144
Программное управление печатью отчетов при помощи события OnManualBuild ... 147
Построение отчета при помощи кода ... 149
Печать табличных отчетов с переменным или неизвестным количеством столбцов ... 151
Табличные отчеты с переменной шириной столбца ... 153

Введение

Введение

Этапы построения отчета

Построение простейшего отчета

Введение

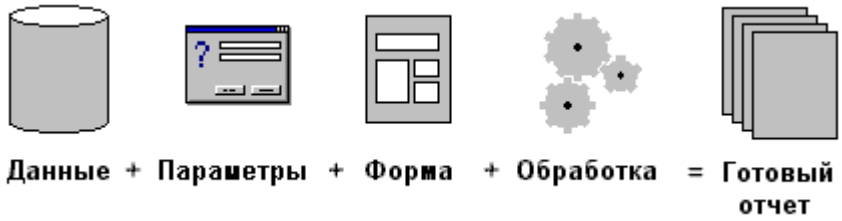
Необходимость написания собственного компонента для построения отчетов возникла в 1997 году, во время разработки системы учета заработной платы. Спецификой такой системы является большое количество форм отчетности, необходимость легкой настройки их внешнего вида. Существующие доступные системы построения отчетов не обладали нужными характеристиками, поэтому было принято решение о разработке собственного генератора отчетов. Концепция была заимствована из 1С-Бухгалтерии 6.0 для Windows - основным элементом отчета является произвольным образом обрамленный прямоугольник с многострочным текстом внутри; в тексте могут содержаться переменные - строки в квадратных скобках. Эта концепция оказалась весьма жизнеспособной - несмотря на то, что в самой первой версии генератора применялся только один бэнд - строка данных, это позволяло строить многоуровневые отчеты.

Позже, в 1998 году, генератор отчетов оформляется в виде компонента (первоначально это был просто набор подключаемых модулей). С этого момента он получает название "FastReport" и начинает стремительно наращивать функциональность. На сегодняшний день FastReport - это полностью визуальный генератор отчетов, т.е. большинство отчетов можно построить, пользуясь только мышью. Его возможности перечислены ниже:

- Бэнд-ориентированный генератор отчетов.
- Встроенный мощный дизайнер, доступный и в run-time.
- Preview как в MS Word.
- Скорость работы сравнима с QuickReport.
- Компактность кода - без дизайнера меньше QR3.
- Неограниченное количество страниц сформированного отчета.
- Многостраничные отчеты; составные (композитные) отчеты; вложенные отчеты; группы; многоколоночные отчеты; master-detail-detail отчеты; cross-tab отчеты; двухпроходные отчеты; "живые" отчеты.
- Полный контроль над процессом печати, поддержка всех типов бумаги.
- Набор наиболее популярных компонентов: Текст, Линия, Рисунок, Фигура, OLE объект, RichText, RX Rich 2.0, Диаграмма, Штрих-код.
- Экспорт в TXT, RTF, CSV, HTML (в RTF, HTML - с картинками).
- Поиск текста в сформированном отчете.
- Редактирование сформированного отчета.
- Встроенный интерпретатор Pascal-подобного языка для управления процессом построения отчета.
- Набор визуальных компонентов для создания диалоговых форм;
- Набор невизуальных компонентов для создания таблиц, запросов и баз данных;
- Add-in компонент TfrDataStorage (менеджер данных), предназначенный для создания запросов и таблиц в run-time - специально для т.н. enduser - отчетов.
- Работа в менеджере данных с BDE, Interbase Express (IBX), ActiveX Data Objects (ADO).
- Работа как с Database - ориентированными источниками данных, так и с любыми данными.
- Форма отчета может храниться как в DFM, так и во внешнем файле.
- Функциональность может быть расширена за счет написания собственных компонент - визуальных объектов, мастеров, библиотек функций.

Этапы построения отчета

Можно выделить следующие компоненты процесса построения типичного отчета:



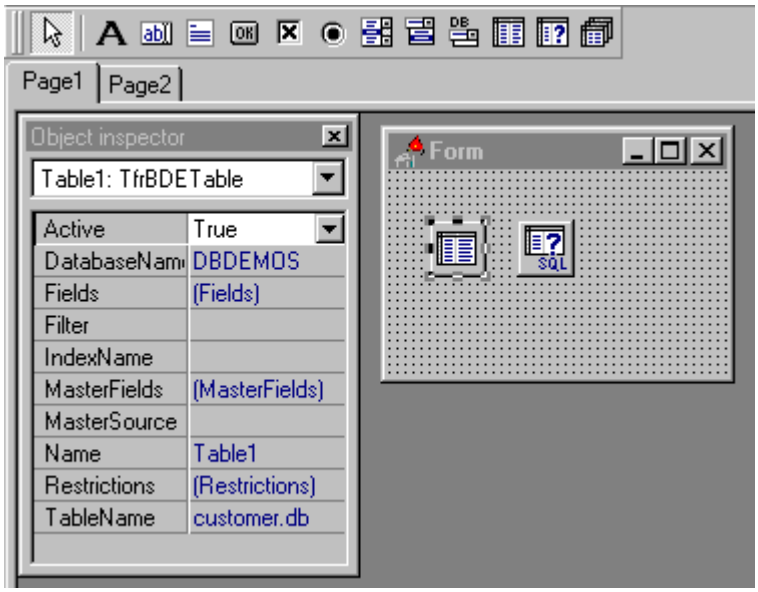
Рассмотрим каждый из компонентов и его реализацию в FastReport, а также в некоторых других генераторах отчетов.

Данные

Представляют собой совокупность данных, на основе которых строится отчет. Большинство отчетов, как правило, основано на данных из БД. Для доступа к таким данным Delphi предоставляет эффективные механизмы, которые и используются в FastReport. Речь идет о компонентах TTable и TQuery, которые могут выступать в качестве источников данных для отчета. Вообще, можно использовать с этой целью любые компоненты - наследники TDataSet. Организация доступа к данным из БД осуществляется ядром FastReport без участия программиста.

Кроме данных, хранимых в БД, FastReport может использовать практически любые источники (массив, файл, содержимое StringGrid и пр.). В этом случае программист должен сам позаботиться о доступе к такой информации. Для этого имеется набор событий, позволяющих легко осуществить передачу данных в ядро FastReport.

Реализация доступа к данным примерно одинакова во всех генераторах отчетов. Все генераторы умеют обращаться с компонентами доступа к данным, расположенными на формах проекта. Кроме доступа к данным, определенным в проекте, и FastReport, и ReportBuilder, и QR+QRDesigner позволяют создавать новые компоненты в run-time. В FastReport принципы создания компонентов доступа к данным максимально приближены к тем, что используются в среде Delphi. Так же, как и в Delphi, на форму кладется компонент и в инспекторе объектов настраиваются его свойства. Компонентная идеология очень гибкая: можно легко создавать новые компоненты для поддержки разных движков доступа к данным.

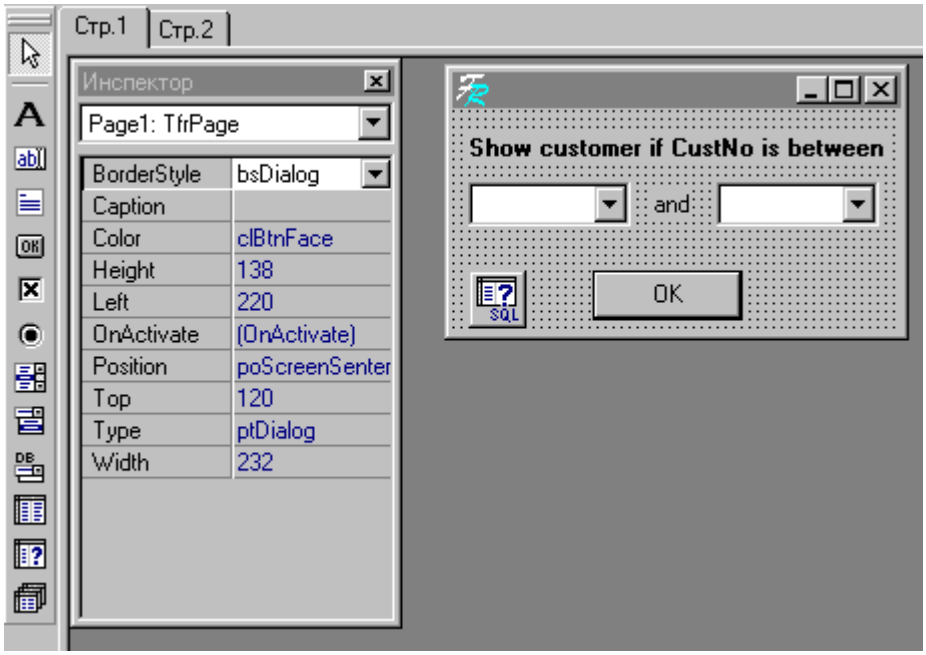


Параметры

На этом этапе осуществляется запрос параметров у пользователя (например, диапазон дат, по которому необходимо вывести данные). Некоторые отчеты обходятся без этого этапа либо используют фиксированную установку параметров (без запроса их значений в диалоге).

Этот этап реализован в разных генераторах отчетов по-разному. Так, в ReportBuilder и QR+QRDesigner есть возможность запроса параметров, если отчет использует данные из запроса (Query). Для диалога с пользователем используется "штатное" диалоговое окно. Кроме того, для запроса параметров можно использовать форму, разработанную в среде Delphi. Правда, при необходимости каких-либо изменений в логике работы придется перекомпилировать проект.

FastReport, помимо этого, позволяет конечному пользователю самому разрабатывать форму диалога. Процесс напоминает построение формы в среде Delphi: имеется набор стандартных элементов управления, которые можно располагать на форме диалога и настраивать их свойства. С помощью встроенного языка FastReport позволяет реализовать необходимую логику работы диалога и передать введенные значения ядру генератора.

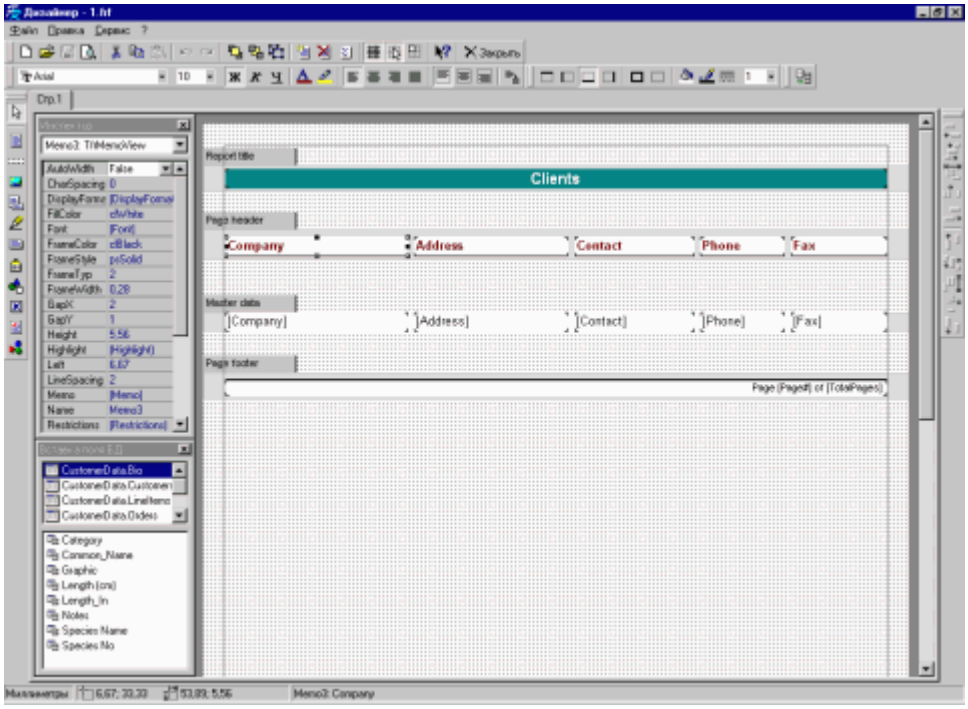


Возможность создания собственных диалогов очень полезна - вкупе с остальными возможностями (создание источников данных, использование встроенного языка) она позволяет создавать "самодостаточные" отчеты, т.е. отчеты, максимально отвязанные от среды Delphi. Это позволяет создавать новые отчеты и модифицировать существующие без переписывания или перекомпиляции проекта.

Форма

Форма отчета представляет собой набор элементов, описывающих то, как должен выглядеть готовый отчет. Для группировки элементов по их расположению в готовом отчете применяются бэнды (англ. band - полоска). Бэнды разделяются на два вида: служебные (заголовок отчета, страницы и пр.) и бэнды, образующие многострочную часть отчета (далее - дата-бэнды). Дата-бэнды подключаются к источникам данных, и их содержимое выводится столько раз, сколько имеется строк данных в источнике.

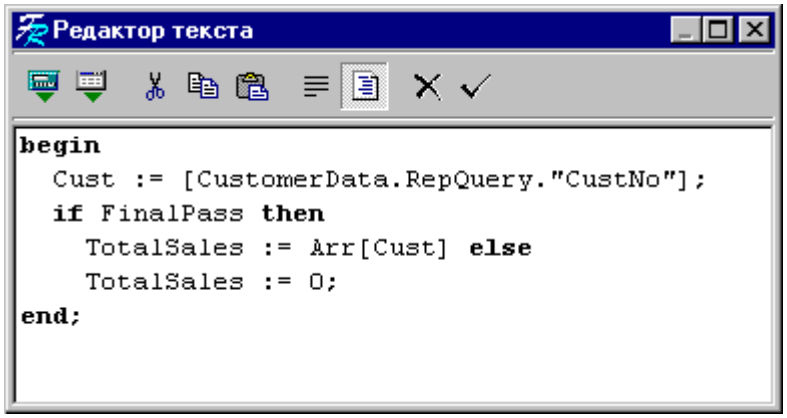
Для построения формы отчета используется визуальная среда разработки - дизайнер. В FastReport дизайнер - это средство разработки отчетов, сочетающее мощь, простоту и удобство использования. Интерфейс дизайнера выполнен на современном уровне с использованием панелей инструментов (toolbars), расположение которых можно изменять по своему вкусу. Для удобства манипуляции свойствами объектов отчета используется инспектор объектов, аналогичный делфийскому.



Обработка

Под обработкой понимается обработка входных данных, модификация формы отчета или отдельных ее компонентов в процессе построения отчета. Простейший пример такой обработки - вывод отрицательных сумм красным цветом. Более сложный пример обработки - печать суммы, которая подсчитывается в подвале группы, в ее заголовке.

Реализовать подобную обработку можно, прибегнув к написанию обработчиков событий в Delphi - именно так и сделано в FastReport, QR, ReportBuilder. Этот способ не является универсальным, т.к. не позволяет создавать новые отчеты вне среды Delphi без переписывания и перекомпиляции проекта. Именно поэтому в FastReport и ReportBuilder применен встроенный язык - упрощенный аналог Pascal. Скрипты, написанные на этом языке, по сути дела, являются обработчиками событий, вызывающимися перед прорисовкой объектов. Это дает возможность выполнять довольно сложную обработку информации без написания кода в Delphi, и, соответственно, без жесткой привязки отчета к проекту.

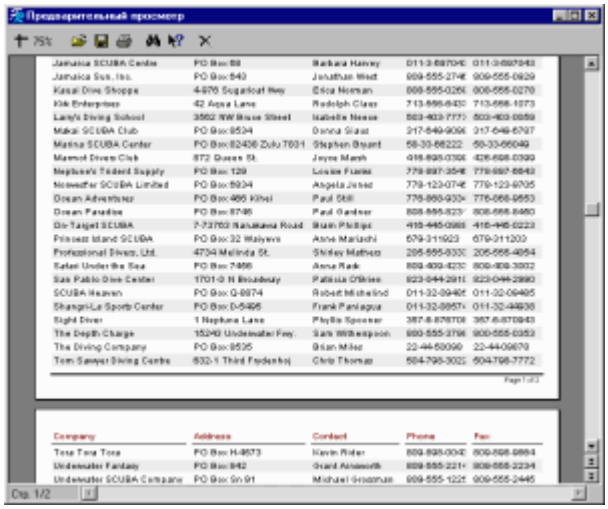


Возможности встроенного языка FastReport довольно широки. Из скрипта доступны все свойства и методы объектов отчета, а также переменные, поля таблиц БД. В скрипте можно создавать переменные и массивы, которые будут доступны во всем отчете. О возможностях встроенного языка говорит тот факт, что такая довольно сложная задача, как печать сумм группы в ее заголовке (сама сумма считается в подвале группы) средствами языка FastReport делается элементарно.

Готовый отчет

Готовый отчет представляет собой продукт деятельности ядра FastReport - то, что мы видим при нажатии кнопки "Предварительный просмотр". В отличие от многих генераторов отчетов, которые хранят содержимое страниц отчета в виде метафайла (т.е. изображения в формате EMF), в FastReport готовый отчет представлен набором объектов, описывающих содержимое каждой страницы отчета. Это позволяет модифицировать готовый отчет, загружая нужную страницу в дизайнер. Кроме того, можно описывать реакцию на щелчок мыши на нужном объекте в режиме предварительного просмотра отчета. Это позволяет легко организовать работу приложения, при которой щелчок на объекте отчета вызывает генерацию нового отчета с более детальными сведениями по выбранному объекту.

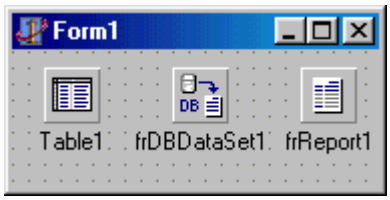
Окно предварительного просмотра FastReport отличается от используемых в других генераторах. В частности, в FastReport используется схема просмотра документа и навигации, принятая в Microsoft Word: в окне можно видеть сразу несколько страниц. Кроме того, реализована возможность поиска текста во всем документе.



Построение простейшего отчета

Рассмотрим построение простейшего отчета средствами FastReport. Для создания простейшего отчета по данным существующей базы данных Вам потребуется выполнить следующие шаги:

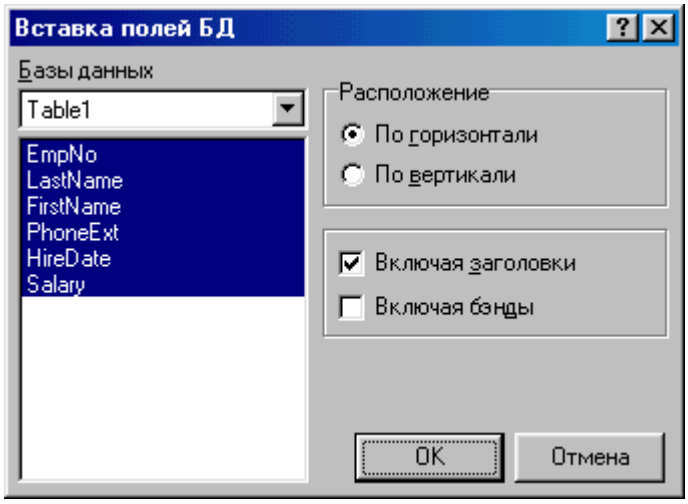
1. поместить на форму три компонента: TTable (TQuery), TfrDBDataSet, TfrReport. Теперь Ваша форма должна выглядеть подобно тому, как показано на рисунке;



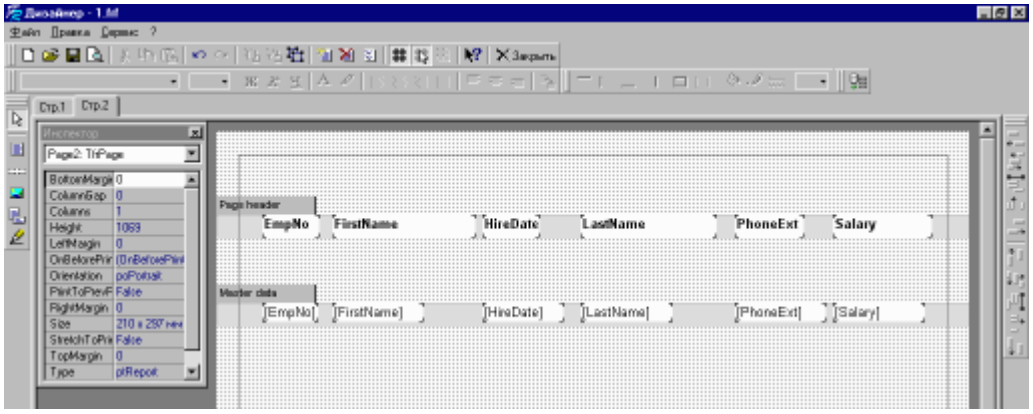
2. связать компонент TTable с нужной базой данных, установив требуемые значения свойств *DatabaseName* и *TableName*;
3. связать свойство *DataSet* объекта TfrDBDataSet с настроенным объектом TTable. На данный момент определение формы должно выглядеть подобно тому как показано ниже:

```
object Form1: TForm1
  Caption = 'Form1'
  object frReport1: TfrReport
    Left = 16
    Top = 8
  end
  object frDBDataSet1: TfrDBDataSet
    DataSet = Table1
    Left = 56
    Top = 8
  end
  object Table1: TTable
    Active = True
    DatabaseName = 'DBDEMOS'
    TableName = 'employee.db'
    Left = 96
    Top = 8
  end
end
```

4. вызвать дизайнер отчетов, щелкнув два раза мышью по объекту TfrReport;
5. щелкнуть в дизайнере мышью по кнопке “Вставка полей БД в отчет” на панели инструментов "Мастера";
6. в открывшемся диалоговом окне выбрать поля, которые нужно включить в отчет и нажать кнопку "ОК".

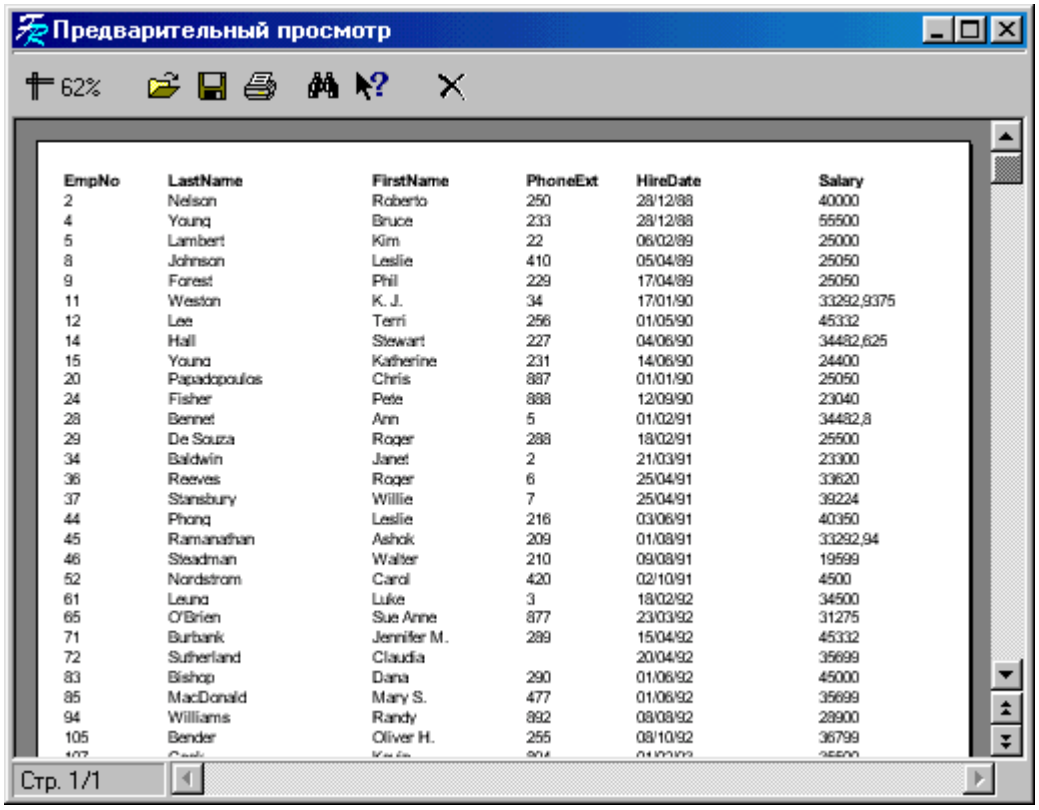


Теперь окно дизайнера будет содержать определение готового отчета с выбранными Вами полями базы данных. Примерный вид окна дизайнера приведен на рисунке.



Автоматически сгенерированный отчет можно использовать в качестве заготовки, для дальнейшего добавления в него графики, задания более смысловых заголовков полей, вставки номеров страниц и т.д. Например, можно модифицировать подготовленный отчет так, чтобы данные выводились на фирменном бланке Вашей организации.

Для того, чтобы посмотреть, как будет выглядеть сформированный отчет на экране нажмите кнопку на панели инструментов дизайнера или выберите пункт меню "Файл|Предварительный просмотр". На экран будет выведено окно с изображением страницы сформированного отчета, так, как она будет выведена на печать:



EmpNo	LastName	FirstName	PhoneExt	HireDate	Salary
2	Nelson	Roberto	250	28/12/88	40000
4	Young	Bruce	233	28/12/88	55500
5	Lambert	Kim	22	06/02/89	25000
8	Johnson	Leslie	410	05/04/89	25050
9	Forest	Phil	229	17/04/89	25050
11	Weston	K. J.	34	17/01/90	33292,9375
12	Lee	Terri	256	01/05/90	45332
14	Hall	Stewart	227	04/06/90	34482,625
15	Young	Katherine	231	14/06/90	24400
20	Papadopoulos	Chris	887	01/01/90	25050
24	Fisher	Pete	888	12/09/90	23040
28	Bennet	Ann	5	01/02/91	34482,8
29	De Souza	Roger	288	18/02/91	25500
34	Baldwin	Janet	2	21/03/91	23300
36	Reeves	Roger	6	25/04/91	33620
37	Stansbury	Willie	7	25/04/91	39224
44	Phong	Leslie	216	03/06/91	40350
45	Ramanathan	Ashok	209	01/08/91	33292,94
46	Steadman	Walter	210	09/08/91	19599
52	Nordstrom	Carol	420	02/10/91	4500
61	Leung	Luke	3	18/02/92	34500
65	O'Brien	Sue Anne	877	23/03/92	31275
71	Burbank	Jennifer M.	289	15/04/92	45332
72	Sutherland	Claudia		20/04/92	35699
83	Bishop	Dana	290	01/06/92	45000
85	MacDonald	Mary S.	477	01/06/92	35699
94	Williams	Randy	892	08/08/92	28900
105	Bender	Oliver H.	255	08/10/92	36799

Ядро FastReport

Палитра компонентов Delphi

Объекты FastReport

Построение отчетов

Палитра компонентов Delphi

При установке FastReport в палитре компонентов Delphi появляются две закладки: FastReport и FR Tools. Первая содержит основные компоненты FastReport, на второй собраны некоторые элементы управления, использованные в FastReport, которые могут оказаться полезными для применения в собственных приложениях.

Закладка "FastReport"

Иконка	Название	Описание
	TfrReport	Основной компонент - генератор отчета. В design-time двойной щелчок на компоненте открывает дизайнер.
	TfrCompositeReport	Композитный (составной) отчет. Предназначен для «склеивания» нескольких отчетов в один. Для этого программно заполняется список Reports ссылками на нужные объекты TfrReport.
	TfrDBDataSet	Источник данных для отчета, ориентированный на данные из таблицы БД или запроса. Для этого у компонента имеется свойство DataSet, которое ссылается на TTable (или TQuery).
	TfrUserDataSet	Источник данных, ориентированный на пользовательские данные, например, массив или содержимое StringGrid. У компонента имеются события OnFirst, OnNext, OnCheckEOF, которым можно назначить свои обработчики для навигации по набору данных.
	TfrOLEObject	Подключаемые объекты. Назначение каждого из объектов будет рассмотрено ниже.
	TfrRichObject	
	TfrCheckBoxObject	
	TfrShapeObject	
	TfrBarcodeObject	
	TfrChartObject	
	TfrRoundRectObject	
	TfrTextExport	Фильтры экспорта. Преобразуют сформированный отчет в файл соотвествующего формата.

	TfrRTFExport	
	TfrCSVExport	
	TfrHTMExport	
	TfrDesigner	Дизайнер отчета. Этот компонент необходимо подключать, если дизайнер нужен в run-time.
	TfrDataStorage	Компонент для т.н. enduser - отчетов. Позволяет в run-time создавать таблицы БД и запросы, настраивать связи между ними - то есть выполнять практически ту же работу, что и в среде Delphi. В настоящее время устарел - пользуйтесь TfrDialogControls + TfrBDEComponents.
	TfrPreview	Компонент для создания пользовательских окон предварительного просмотра.
	TfrPrintTable	Компонент для печати содержимого таблицы БД "на лету".
	TfrPrintGrid	Компонент для печати содержимого DBGrid "на лету".
	TfrDialogControls	Подключаемые элементы управления для создания диалоговых окон с элементами управления в run-time.
	TfrBDEComponents	Подключаемые DB-aware компоненты для создания таблиц, запросов и баз данных в run-time.

Закладка "FR Tools"

Иконка	Название	Описание
	TfrSpeedButton	Аналог TSpeedButton. Может быть плоской (Flat) и менять цвет при наведении курсора мыши (GrayedInactive).
	TfrDock	Панель, к которой могут быть пристыкованы панели инструментов (TfrToolBar).
	TfrToolBar	Панель инструментов, аналогичная MS Office toolbar.
	TfrTBButton	Кнопка для панели инструментов.
	TfrTBSeparator	Разделитель для панели инструментов.
	TfrTBPanel	Область на панели инструментов, которая может содержать иные элементы управления, кроме TBButton & TBSeparator.

	TfrOpenDBDialog	Диалог открытия таблицы локальной БД (Paradox, DBase) с выбором алиаса.
	TfrComboBox	Плоский ComboBox.
	TfrFontComboBox	ComboBox для выбора шрифта.
	TfrComboEdit	Строка редактирования с кнопкой справа.

Рассмотрим коротко основные компоненты.

Компонент TfrReport

Этот компонент содержит методы для считывания/записи формы отчета, выполнения предварительного просмотра и печати. Один компонент TfrReport может хранить один отчет.

Свойства TfrReport.

Свойство	Значение по умолчанию	Описание
DataSet		Источник данных TfrxxxDataSet, количество записей в котором определяет, сколько раз строить отчет (см. ReportType).
GrayedButtons	False	Если True, кнопки в окне предварительного просмотра будут вести себя как в Internet Explorer (т.е. будут выглядеть серыми при отсутсвии над кнопкой курсора мыши).
InitialZoom	pzDefault	Определяет масштаб, в котором будет отображаться отчет в окне предварительного просмотра.
MDIPreview	False	Позволяет окну предварительного просмотра работать в составе MDI-приложения.
ModalPreview	True	Если True, окно предварительного просмотра будет модальным.
ModifyPrepared	True	Если True, то построенный отчет можно редактировать.
Preview		Ссылка на компонент TfrPreview. Если это свойство не пустое, то вывод отчета будет происходить в подключенный компонент TfrPreview.
PreviewButtons	all	Определяет, какие кнопки показывать в окне предварительного просмотра.
ReportType	rtSimple	Определяет, как трактовать источник данных, подключенный к свойству DataSet. Если ReportType = rtMultiple, то построение отчета будет выполнено столько раз, сколько записей в подключенном DataSet. Удобно при печати формы отчета типа "бланк" (напр., платежное поручение).
ShowProgress	True	Если True, показывает форму прогресса при построении отчета.
StoreInDFM	False	Определяет, надо ли хранить форму отчета в форме Delphi (файле DFM). Отчеты, хранимые в DFM, нельзя модифицировать в run-time.
Title		Заголовок отчета. Отображается в окне предварительного просмотра, назначается печатаемому заданию.

Следует отметить, что по умолчанию свойство StoreInDFM = False. Это означает, что все манипуляции с формой отчета в компоненте TfrReport не будут сохранены в скомпилированном приложении и приведет к тому, что при построении отчета будет выведен чистый лист. Это сделано потому, что концепция FastReport отдает предпочтение хранению формы отчета во внешних файлах или базах данных. В этом случае раскрываются все возможности FastReport, а именно возможность изменения формы отчета в run-time и создание новых отчетов. Кроме того, отчеты, хранимые во внешних файлах, можно модифицировать и распространять без перекомпиляции проекта.

События TfrReport.

Событие	Описание
OnBeforePrint	Событие вызывается перед печатью содержимого любого визуального объекта.
OnBeginBand	Событие вызывается перед печатью бэнда.
OnBeginColumn	Событие вызывается перед печатью строки cross-tab отчета.
OnBeginDoc	Событие вызывается перед началом формирования отчета.
OnBeginPage	Событие вызывается перед началом формирования страницы.
OnEndBand	Событие вызывается после вывода содержимого бэнда.
OnEndDoc	Событие вызывается после окончания формирования отчета.
OnEndPage	Событие вызывается после окончания формирования страницы отчета.
OnGetValue	Событие вызывается, если в отчете встретилась неизвестная переменная. Обработчик события должен вернуть значение переменной.
OnManualBuild	Если событие имеет обработчик, то формирование отчета (вывод бэндов) будет осуществляться этим обработчиком (см. демо MANUAL).
OnMouseOverObject	Событие вызывается, если указатель мыши находится над объектом в режиме предварительного просмотра. Обработчик события может вернуть тип курсора, который должна принимать мышь при нахождении над тем или иным объектом.
OnObjectClick	Событие вызывается при щелчке левой кнопкой мыши на объекте в режиме предварительного просмотра.
OnPrintColumn	Событие вызывается перед печатью ячейки cross-tab отчета. Обработчик события может вернуть ширину ячейки.
OnProgress	Это событие вызывается при выполнении длительных операций, таких как построение отчета, печать или экспорт. Обработчик события может

	выводить процент выполненной работы в собственном окне прогресса.
OnUserFunction	Событие вызывается, если в отчете встретилась неизвестная функция. Обработчик события должен вернуть значение функции.

Основные методы TfrReport.

Метод	Описание
LoadFromFile, LoadFromStream, LoadFromBlobField	Считывает форму отчета из файла, потока, BLOb-поля таблицы БД.
SaveToFile, SaveToStream, SaveToBlobField	Записывает форму отчета в файл, поток, BLOB-поле.
DesignReport	Запускает дизайнер отчета. При этом в проект должен быть включен дизайнер (компонент TfrDesigner).
ShowReport	Строит отчет и выводит его на предварительный просмотр.
PrepareReport	Строит отчет без вывода на предварительный просмотр.
PrintPreparedReport	Печатает предварительно построенный отчет.
PrintPreparedReportDlg	Печатает предварительно построенный отчет. Перед печатью выводит диалоговое окно с опциями печати.
ExportTo	Выполняет экспорт предварительно построенного отчета.

Компонент TfrDBDataSet

Невизуальный компонент TfrDBDataSet представляет собой источник данных для отчета, ориентированный на данные из таблицы БД или запроса. Компонент выполняет одну-единственную задачу: предоставить генератору отчетов методы для навигации по набору данных (под навигацией понимается установка на начальную запись и последовательный выбор записей, пока не будет достигнут конец набора данных). Компонент TfrDBDataset, по сути, является "переходником" для подключения к TDataSet. В отличие от ReportBuilder TppBDEPipeline, компонент не предоставляет информацию о полях набора данных, к которому он подключен.

Свойство	Описание
CloseDataSource	Закрывает источник данных после формирования отчета.
DataSet	Источник данных.
OpenDataSource	Открывает источник данных перед формированием отчета.
RangeBegin	Начальная запись источника.
RangeEnd	Конечная запись источника.

Ключевым свойством данного объекта является свойство DataSet, которое ссылается на объект типа TDataSet (TTable, TQuery). Дополнительно разработчик может указать на необходимость открыть источник данных перед построением отчета, установив значение OpenDataSource в True и/или закрыть источник, установив значение CloseDataSource в True.

Для ограничения диапазона используемых значений служат свойства RangeBegin и RangeEnd. Значение свойств позволяет определить первую и последнюю записи используемого набора данных. Свойство RangeBegin может принимать следующие значения:

- rbFirst – выбор данных из источника начнется с первой записи;
- rbCurrent – выбор данных из источника начнется с текущей записи;

Свойство RangeEnd может принимать следующие значения:

- reLast – выбор данных из источника будет продолжаться до тех пор, пока не достигнут конец набора данных;
- reCurrent – извлечение данных из источника завершается на текущей записи;
- reCount – данное значение используется в паре со свойством RangeEndCount объекта TfrDBDataSet. При выборе данного значения из источника выбирается указанное в свойстве RangeEndCount количество записей, начиная с позиции, определнной свойством RangeBegin.

Компонент TfrUserDataSet

Данный объект является аналогом TfrDBDataSet, но ориентирован на пользовательские данные (например массив или содержимое StringGrid). Еще одним аналогом данного объекта является «виртуальный» источник данных, доступный в дизайнере FastReport из редактора дата-бэндов.

У данного объекта отсутствуют свойства DataSet и Close/OpenDataSource. Для навигации по пользовательским данным служат четыре обработчика событий:

Событие	Описание
OnCheckEOF	Обработчик события принимает в качестве параметра переменную EOF логического типа, которая должна быть установлена в True при достижении конца данных и в False в противном случае.
OnFirst	Вызывается тогда, когда требуется переместить указатель на первую запись в наборе данных.
OnNext	Вызывается для перехода к следующей записи.
OnPrior	Вызывается при переходе к предыдущей записи. Это происходит только при печати отчета, содержащего группы.

Методы навигации вызываются в следующем порядке: OnFirst, OnCheckEOF, OnNext, OnCheckEOF, OnNext, ..., OnCheckEOF, OnNext. Если в отчете имеются группы, то при печати бэнда Group footer FastReport откатывается на предыдущую запись, вызывая метод OnPrior.

В случае, если число записей в наборе данных известно заранее, удобно установить свойство RangeEnd := reCount и RangeEndCount := количеству записей. В этом случае обработчик события OnCheckEOF можно не использовать. Свойство RecNo: Integer хранит текущую позицию в наборе данных (начальной позиции соответствует 0).

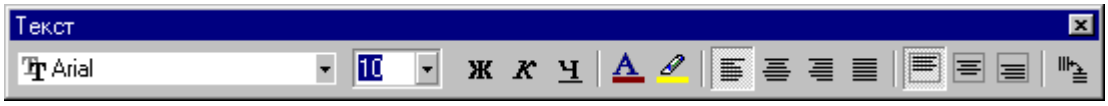
Объекты FastReport

Иконка	Название	Описание
	Текст	Предназначен для отображения текста (однострочного или многострочного). В тексте могут присутствовать переменные, выражения или поля БД (обрамляются квадратными скобками).
	Бэнд	Бэнд (англ. band - полоска). Как и во всех band-ориентированных генераторах отчетов, позволяет объединять объекты в группы (заголовок отчета, страницы, многострочная часть, подвал и пр.).
	Рисунок	Предназначен для отображения рисунка в форматах BMP, ICO, WMF, EMF, JPG. Также способен отображать рисунок из BLOb-поля БД.
	Вложенный отчет	Предназначен для создания вложенных отчетов. Будучи вставленным в отчет, приводит к созданию новой страницы, содержимое которой будет выведено вместо объекта при формировании отчета.
	Линия	При выборе этого объекта курсор принимает форму карандаша. На странице отчета при этом можно рисовать вертикальные и горизонтальные линии.
	Текст с тенью	Выполняет те же функции, что и объект "Текст". Кроме того, умеет отображать тень или градиентную заливку. Очень удобный объект для печати наклеек.
	Штрих-код	Отображает данные в форме штрих-кода. Способен отображать данные из поля БД.
	Фигура	Отображает один из четырех типов фигур: прямоугольник, прямоугольник с закругленными углами, эллипс и треугольник.
	Флажок	Отображает крестик или галочку внутри прямоугольника.
	RichText	Отображает форматированный текст (формат RTF). В тексте могут присутствовать переменные/выражения/поля БД. Также способен отображать данные из BLOb-поля БД.
	OLE объект	Отображает объект OLE. Также способен отображать данные из BLOb-поля БД.
	Диаграмма	Отображает данные в виде диаграммы (используется компонент TeeChart).

	RichText 2.0	Отображает форматированный текст с картинками и OLE-вставками (формат RTF 2.0, используется компонент TrxRichEdit). В тексте могут присутствовать переменные/выражения/поля БД. Также способен отображать данные из BLOb-поля БД.

Объект «Текст»

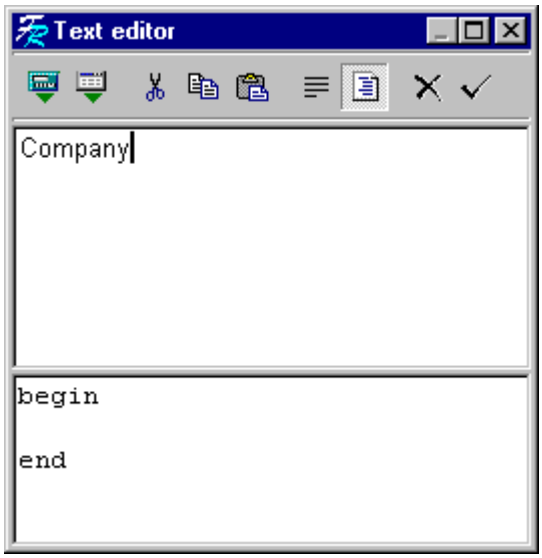
Представляет собой произвольным образом обрамленный прямоугольник с многострочным текстом внутри. Задается цвет текста/заливка, тип рамки, толщина рамки и цвет, все параметры шрифта текста, выравнивание текста внутри прямоугольника и ориентация текста (норм./90 градусов). Для этого предназначены панели инструментов "Текст" и "Прямоугольник":



В мемо объекта можно поместить многострочный текст с переменными. Переменные обрамляются квадратными скобками. При формировании отчета, когда в объекте обнаруживается переменная, вызывается событие, которому передается имя переменной (это может быть и не имя, а выражение) и ожидается возврат ее значения. Примеры использования переменных:

Длина, см: [Длина] – текст (Длина, см:) и переменная ([Длина])
Длина, см: [Table1."Length_cm"] – текст и поле таблицы БД
Длина, см: [[Длина в дюймах] * 2.54] – текст и выражение с переменной
Длина, см: [Table1."Length_in" * 2.54] – текст и выражение с полем БД

Редактор объекта позволяет редактировать содержимое мемо, быстро вставлять переменные, выражения и поля доступных таблиц БД в мемо, а также редактировать скрипт. Для более удобной работы можно воспользоваться быстрыми клавишами: Insert - вставка переменной; клавиша Ctrl+Enter - аналог кнопки OK.



Кнопки на панели инструментов выполняют следующие функции:

вызывает конструктор выражений (см. далее);

вызывает диалог вставки поля БД в текст (см. далее);

операции с буфером обмена;

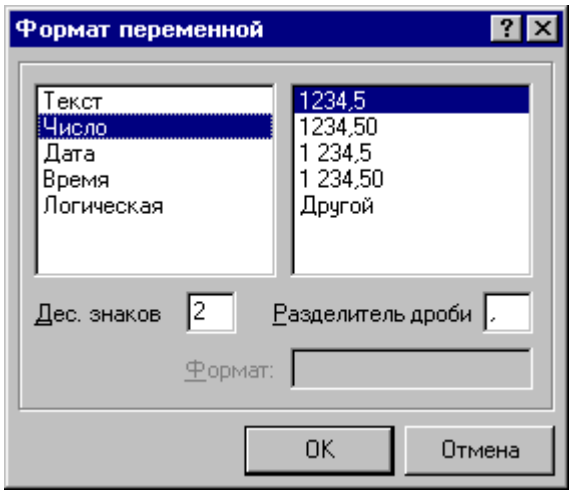
включает или выключает перенос слов в длинных строках;

включает или выключает окно со скриптом;

кнопка "Отмена";

кнопка "ОК".

Помимо основного редактора, имеется редактор, позволяющий выбрать формат отображения имеющихся в тексте переменных. Вызвать его можно либо из контекстного меню объекта, либо из инспектора объектов (свойство DisplayFormat).



Переменная может отображаться как текст, число, дата, время, булевское значение (список в левой части окна). Для каждого из этих типов (кроме текстового представления) можно выбрать один из нескольких способов форматирования (список в правой части окна), например, для числа можно задать количество знаков после запятой, знак-разделитель целой и дробной части, разделение на разряды.

Кроме того, для каждого типа может быть выбран произвольный формат, когда строка форматирования задается вручную (например, #,###,##0.000 для числа). Для этого необходимо в списке справа выбрать тип форматирования "Другой". Строки форматирования для каждого из типов описаны в справочной системе Delphi, раздел formatting strings (кроме булевского типа, где строка представлена в формате Значение1;Значение2). Если выбран текстовый тип (по умолчанию), то форматирование производиться не будет.

Указанные опции форматирования будут воздействовать на все переменные, содержащиеся в объекте. Если переменная не может быть отформатирована указанным способом, она выводится как текст.

Если в объекте несколько переменных, для которых нужно разное форматирование, то можно применить другой способ форматирования. Суть его заключается в том, что к переменной дописывается тэг формата:

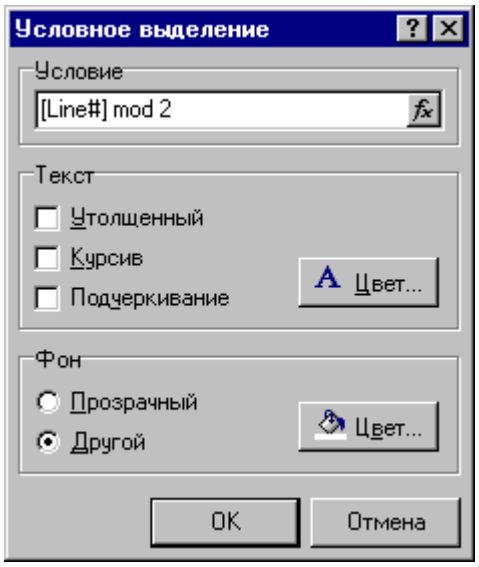
[Переменная формат], где формат - одно из следующих значений:

- #x.x или #Nx.x или #Nyyyyy - числовое представление. x.x - длина числа/число разрядов после запятой; yyyyy - строка типа #,##0.00 (более подробную информацию можно получить в справочной системе Delphi, раздел formatting strings). Если строка x.x или yyyyy содержит символы ".", ",", "-", то последний такой символ будет использован в качестве разделителя целой и дробной части форматированного числа.
- #Dxxxxx, #Txxxxx - дата и время. xxxxx - строка типа dd.mm.yy.
- #Bxxxxx;yyyyy - булевское значение. Если значение - ложь, то выводится строка xxxxx, иначе yyyyy.

Вот примеры использования форматирования:

[Table1."N1" #9.2] [Table1."N2" #N9-2] [Table1."N3" #N#,##0.00] - для числа
[Table1."Date1" #Ddd.mm.yyyy] [Table1."Time" #Thh:mm:ss] - для даты и времени
[Table1."Bool1" #BFalse;True] [Table1."Bool2" #BНет;Да] - для булевского значения

Для того, чтобы отобразить объект другим шрифтом или цветом в зависимости от выполнения какого-либо условия, можно воспользоваться диалогом "Условное выделение". Диалог можно вызвать с помощью кнопки на панели инструментов "Текст".



К примеру, нужно выделить суммы заказов, превышающие 1000$ жирным шрифтом. Для этого в объекте, содержащем сумму заказа, в диалоге указывается условие – Value > 1000 и задаются параметры шрифта. Условие выделения можно указать и по-другому – [Part total] > 1000. Другой пример использования выделения – для придания отчету презентабельного вида можно сделать чередующуюся раскраску строк данных. Для этого на бэнд надо поместить объект "Текст", растянуть его по ширине секции, в диалоге выделения набрать условие [LINE#] mod 2 = 0 и указать цвет фона (к примеру, серый).

В контекстном меню объекта можно задать следующие опции:

- "растягиваемый" (Stretched) - высота объекта будет зависеть от количества строк в нем. При этом опция "растягиваемый" должна быть включена у бэнда, на котором находится объект. При печати бэнда вычисляется его высота, и все объекты с включенным растягиванием выводятся таким образом, что их нижняя граница растягивается до нижней границы бэнда.
- "перенос слов" (WordWrap) - если слово не помещается в строке, оно переносится на следующую строку.
- "перенос по слогам" (WordBreak) - при переносе слова делается разбивка его на слоги. При этом опция "перенос слов" должна быть включена.

- "только текст" (TextOnly) - содержимое объекта трактуется как текст, переменные и выражения не обрабатываются.
- "скрывать повторяющиеся" (Suppress) - не выводить объекты с повторяющимися значениями.
- "автоширина" (AutoWidth) - при печати рамка растягивается по ширине объекта.

Кроме того, в инспекторе объектов можно задать следующие значения:

- CharSpacing - расстояние между буквами;
- GapX, GapY - отступы текста слева и сверху;
- LineSpacing - расстояние между строками текста.

Объект «Секция» (Бэнд)

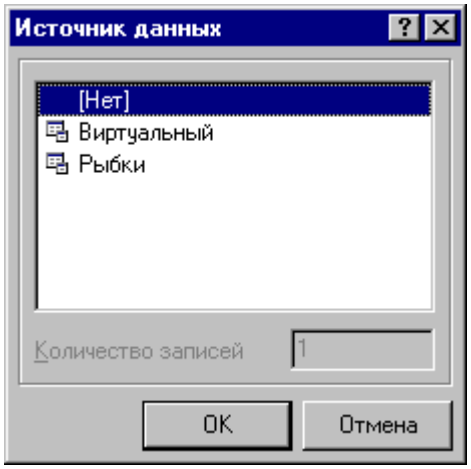
FastReport, как и все распространенные продукты, является бэнд-ориентированным построителем отчетов. Ниже приведен полный перечень бэндов и выполняемых ими функций.

Название	Функция
Report title	печатается один раз в начале отчета
Report summary	печатается один раз в конце отчета
Page header	печатается вверху на каждой странице
Page footer	печатается внизу на каждой странице
Master header	печатается в начале списка 1-го уровня
Master data	данные списка 1-го уровня
Master footer	печатается в конце списка 1-го уровня
Detail header	печатается в начале списка 2-го уровня
Detail data	данные списка 2-го уровня
Detail footer	печатается в конце списка 2-го уровня
Subdetail header	печатается в начале списка 3-го уровня
Subdetail data	данные списка 3-го уровня
Subdetail footer	печатается в конце списка 3-го уровня
Overlay	печатается на каждой странице нижним слоем
Column header	печатается в начале каждой колонки
Column footer	печатается в конце каждой колонки

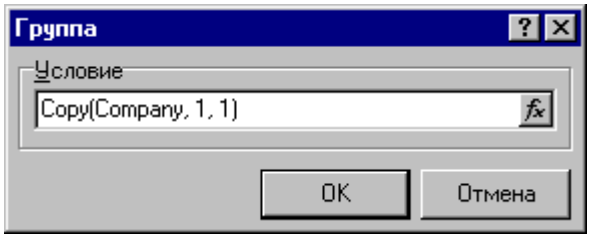
Group header	заголовок группы
Group footer	печатается после группы
Cross header Cross data Cross footer	эта группа бэндов предназначена для создания отчетов с переменным количеством столбцов и разбивкой на страницы
Child	может быть прикреплен к любому из бэндов, кроме Page footer, и выводиться вместе с ним.

Как видно из таблицы, набор бэндов несколько отличается от общепринятого. Так, в классической схеме (которой придерживается, в частности, ReportBuilder) имеется только один дата-бэнд Detail.

В зависимости от типа бэнда вызываются разные редакторы. Редактор для дата-бэндов позволяет выбрать источник данных из списка доступных либо выбрать виртуальный источник данных. При выборе виртуального источника необходимо указать, из скольких записей он состоит. При формировании отчета бэнд с виртуальным источником будет напечатан столько раз, сколько записей было задано.



Редактор для бэнда GroupHeader позволяет ввести условие группировки. Остальные бэнды не имеют редакторов.



В зависимости от типа бэнда в контекстном меню доступны следующие опции:

- "растягиваемый" (Stretched) - высота бэнда определяется максимальной высотой находящихся в нем объектов. У объектов также должна быть включена опция "растягиваемый";
- "разрываемый" (Breaked) - если бэнд не помещается целиком на странице, будут выведены только поместившиеся строки текста, а вывод остальных будет продолжен с новой страницы;

- "формировать новую страницу" (FormNewPage) - после печати бэнда формирование отчета продолжается с новой страницы. Если опция включена у дата-бэнда, то новая страница будет сформирована после вывода всех подбэндов;
- "на первой странице" (OnFirstPage) - если опция отключена, то бэнд не будет выведен на первой странице;
- "на последней странице" (OnLastPage) - если опция отключена, то бэнд не будет выведен на последней странице;
- "повторять на всех страницах" (RepeatHeader) - эта опция доступна у бэндов Master header, Detail header, Subdetail header, Group header, Cross header. При включении дублирует бэнд на новой странице или колонке.

Объект «Рисунок»

Предназначен для вставки графического файла в формате BMP/WMF/ICO (и JPG, если установлен соответствующий модуль) в документ. Редактор объекта позволяет вставить рисунок и очистить содержимое объекта.

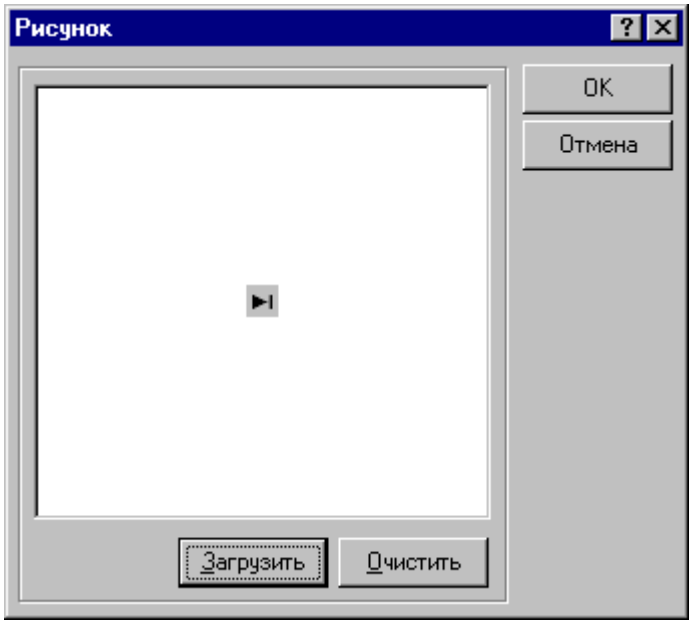


Рисунок может быть вставлен из поля таблицы БД, если в мемо объекта поместить ссылку на соответствующее поле, например: [Table1."GraphicField"]. Редактор мемо может быть вызван с помощью инспектора объектов либо комбинацией клавиш Ctrl+Enter. Эту опреацию можно также выполнить из инспектора объектов (свойство DataField).

В контекстном меню объекта и в инспекторе можно задать следующие опции:

- "растягиваемый" (Stretched) - рисунок растягивается по границам объекта;
- "сохранять пропорции" (KeepAspect) - если включено растягивание, пропорции рисунка сохраняются;
- "центровка" (Center) - рисунок выводится в центре объекта.
- BlobType - если рисунок хранится в поле таблицы БД, эта опция позволяет выбрать тип хранимого рисунка: BMP, WMF, ICO или JPG.

Объект «Вложенный отчет» (SubReport)

Вложенный отчет - это отчет с размещенным на нем объектом SubReport, который представляет собой ссылку на другой отчет, расположенный на отдельной странице. При

формировании такого отчета вместо объекта SubReport будет выведен соответствующий отчет.

Объекты SubReport можно располагать на листе друг под другом или рядом. В случае, если объекты надо расположить друг под другом, их необходимо разнести по разным дата-бэндам (т.е. первый отчет - в первом бэнде Masterdata, второй - во втором и т.д.). На использование SubReports накладываются следующие ограничения:

- нельзя использовать колонки;
- во вложенных отчетах (не в основном!) игнорируются бэнды ReportTitle, ReportSummary, PageHeader, PageFooter, ColumnXXX;
- нельзя использовать разрываемые бэнды;
- нельзя использовать группы.

Объект «Линия»

Представляет собой вертикальную либо горизонтальную линию с изменяемым цветом и толщиной. Дизайнер имеет удобные средства для рисования линий. В частности, с помощью линий можно быстро задать внешнее обрамление для таблицы, отличное от внутреннего.

Для рисования линии выберите инструмент "Линия" на панели объектов. При этом курсор мыши примет форму карандаша при перемещении его над поверхностью листа. Установите курсор мыши в место начала линии, нажмите левую кнопку мыши и двигайте мышь до нужной позиции. После отпускания кнопки будет нарисована линия. После окончания рисования нужно нажать кнопку "выбор объекта" либо Esc.

Объект «CheckBox»

Представляет собой прямоугольник с крестиком внутри. Если в мемо объекта поместить ссылку на переменную либо на поле таблицы БД, то крестик будет показан при ненулевом значении переменной. В мемо можно также поместить символ «0» либо отличный от нуля.

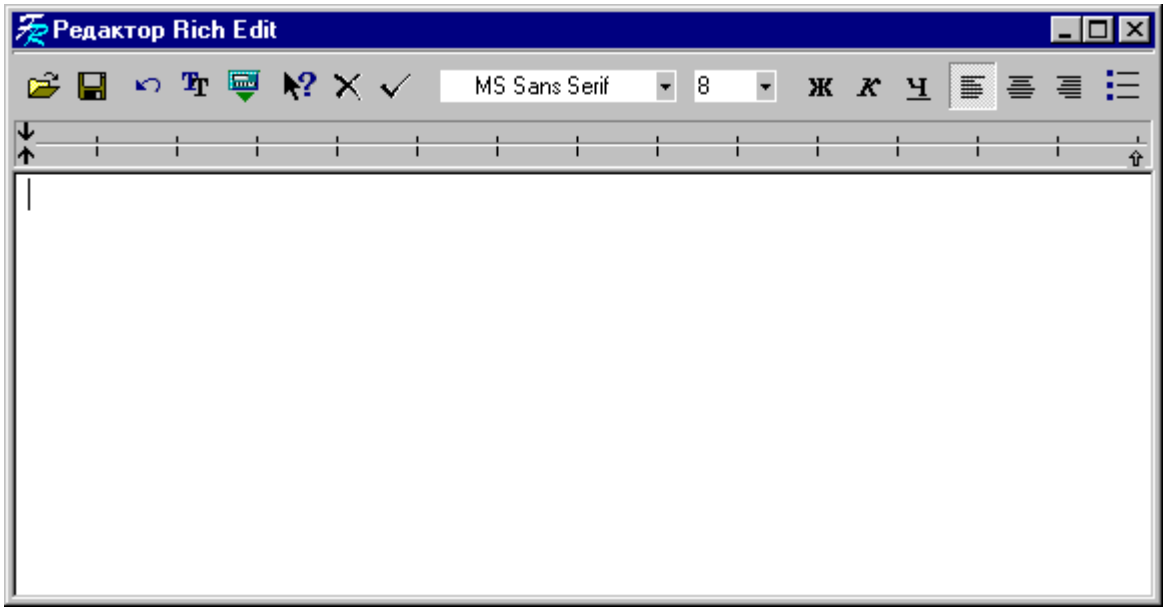
В инспекторе объектов можно выбрать тип CheckBox’a: с крестиком или с галочкой (свойство CheckStyle).

Примечание. Все вышеописанные объекты являются встроенными. Для использования этого объекта и всех прочих в run-time следует использовать соответствующий компонент из палитры компонентов FR. Если открываемый отчет содержит неподключенные компоненты, то выдастся сообщение об ошибке.

Объект «RichText»

Предназначен для вставки текста в формате RTF (Rich Text format) в документ. В частности, таким образом можно вставить текст, подготовленный в текстовом процессоре MS Word, предварительно сохранив его в формате RTF.

Редактор объекта взят из Delphi\Demos\Richedit и позволяет выполнять все базовые операции над текстом - менять шрифт, выравнивание, создавать маркированные списки.

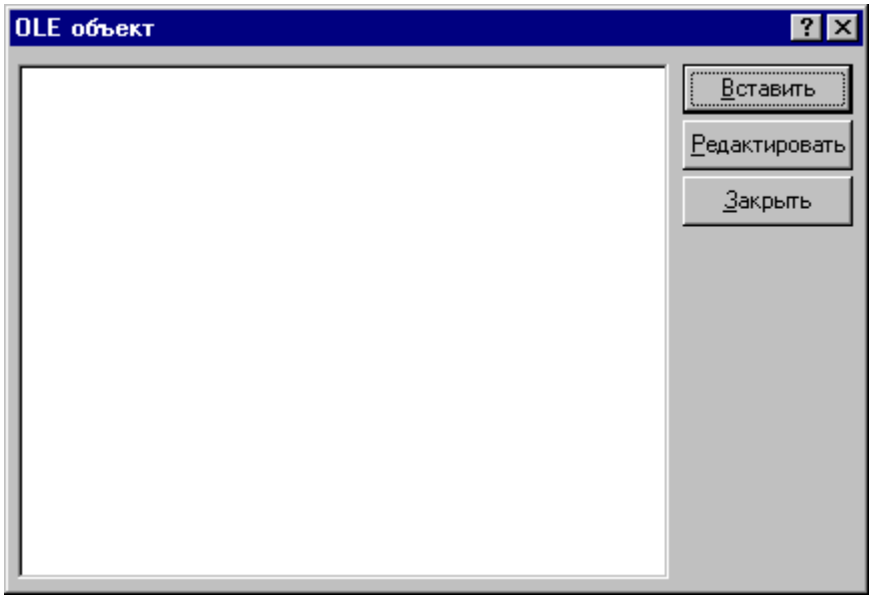


В тексте могут находиться переменные, обрамленные квадратными скобками - как и в объекте «Текст». RichText объект может быть вставлен из поля таблицы БД, если в мемо объекта (не в сам объект!) поместить ссылку на соответствующее поле, например: [Table1."RichField"]. Редактор мемо может быть вызван с помощью инспектора объектов либо комбинацией клавиш Ctrl + Enter. Эту опреацию можно также выполнить из инспектора объектов (свойство DataField).

Для использования этого объекта в run-time необходимо подключить компонент TfrRichObject из палитры компонент FastReport.

Объект «OLE»

Предназначен для вставки OLE объектов в документ. Редактор объекта «OLE» позволяет вставить новый объект и редактировать существующий.



Вставка нового объекта производится в стандартном диалоговом окне. OLE объект может быть вставлен из поля таблицы БД, если в мемо объекта поместить ссылку на соответствующее поле, например: [Table1."OLEField"]. Редактор мемо может быть вызван с

помощью инспектора объектов либо комбинацией клавиш Ctrl + Enter. Эту опреацию можно также выполнить из инспектора объектов (свойство DataField).

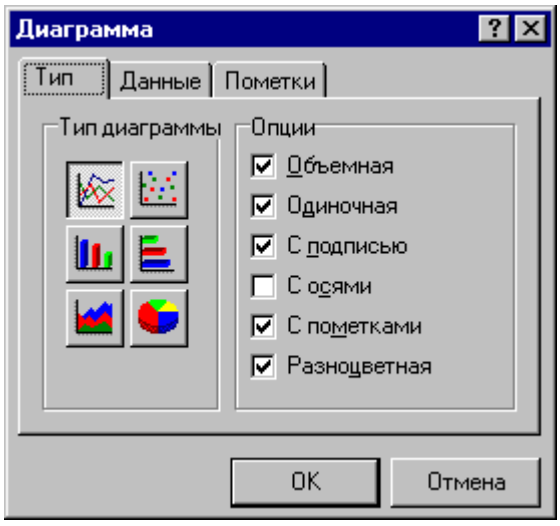
В контекстном меню объекта опция "растягиваемый" (Stretched) позволяет корректно отображать данные в некоторых случаях (например, при вставке объекта "Лист Excel").

Для использования этого объекта в run-time необходимо подключить компонент TfrOLEObject из палитры компонент FastReport.

Объект «Диаграмма»

Предназначен для вставки графика или диаграммы в документ. Возможно использование шести типов графиков: линии, точки, столбцы горизонтальные и вертикальные, области и круговые диаграммы. Редактор объекта представляет собой блокнот с тремя закладками: "Тип", "Данные" и "Пометки". Здесь можно задать различные опции для графика, а также указывать объекты, содержимое которых будет использовано при построении графика.

Закладка "Тип"



На этой закладке можно выбрать один из шести типов диаграммы, а также указать различные опции:

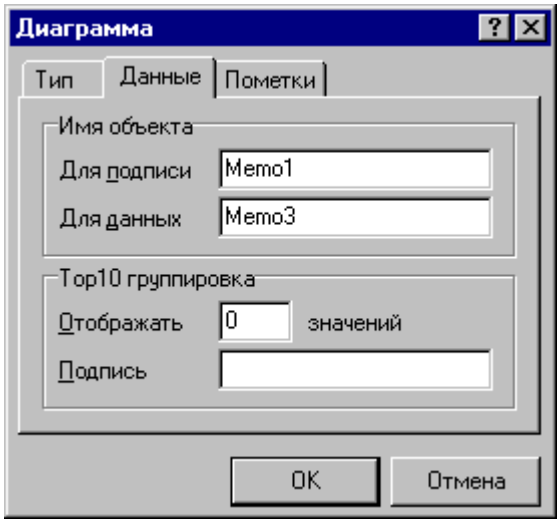
- "объемная" - при включении диаграмма становится трехмерной;
- "одиночная" - должна быть включена, если на диаграмме расположен один график;
- "с подписью" - при включенной опции выводит рядом с диаграммой поясняющую подпись;
- "с осями" - выводить оси (нужно отключать для круговой диаграммы);
- "с пометками" - выводит пометки над каждым значением
- "разноцветная" - если опция включена, каждое значение в диаграмме (столбец, сектор, линия) показывается разными цветами.

При построении отчета информация будет накапливаться в мемо:

Заголовок1;Заголовок2;Заголовок3
Число1;Число2;Число3

График можно построить и вручную, если занести в мемо объекта соответствующие значения. Если количество строк заголовка не соответствует количеству значений, график выведен не будет.

Закладка "Данные"

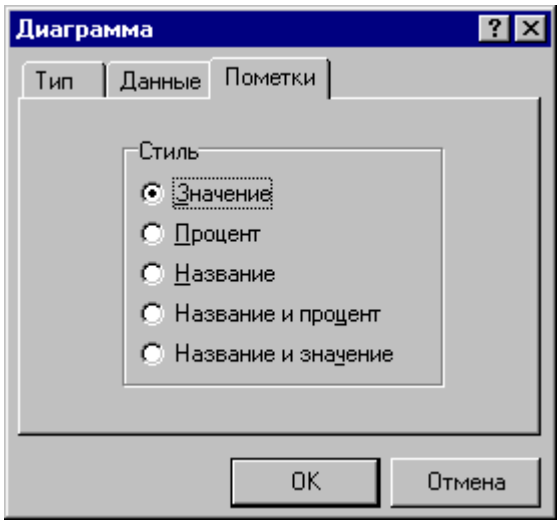


Для построения графика необходимо указать имена двух объектов типа "Текст", в которых содержится название и соответствующее числовое значение. При построении отчета содержимое этих объектов будет накапливаться в мемо диаграммы.

Если объект содержит число в форматированном виде или комбинацию числа и строки (например, "$1000" или "10 000"), то FR попытается выделить числовую часть из строки. Алгоритм выделения - удаляются все нецифровые символы в начале и в конце строки, после этого удаляются все символы - разделители разрядов (как правило, пробелы). То есть если задано более сложное форматирование, например: "10000км2", то такие объекты использовать уже нельзя. В этом случае следует создать невидимый объект с аналогичным содержимым, но без всякого форматирования, и указывать его имя. Невидимым объект можно сделать с помощью инспектора объектов в дизайнере, установив параметр Visible в 0.

Объект позволяет легко строить графики типа "Top10", т.е. график, в котором представлены только первые самые большие значения, а остальные значения просуммированы в отдельную группу с названием. Для этого необходимо указать количество выводимых значений и подпись (обычно в качестве подписи используется слово "Другие").

Закладка "Пометки"



На этой закладке можно указать, какие значения выводить в качестве пометок.

Для использования этого объекта в run-time необходимо подключить компонент TfrChartObject из палитры компонент FastReport.

Примечание. Объект доступен, начиная с 3-й версии Delphi. Возможно его использование и в Delphi 2, для этого необходимо иметь установленный компонент TeeChart.

Объект «Фигура»

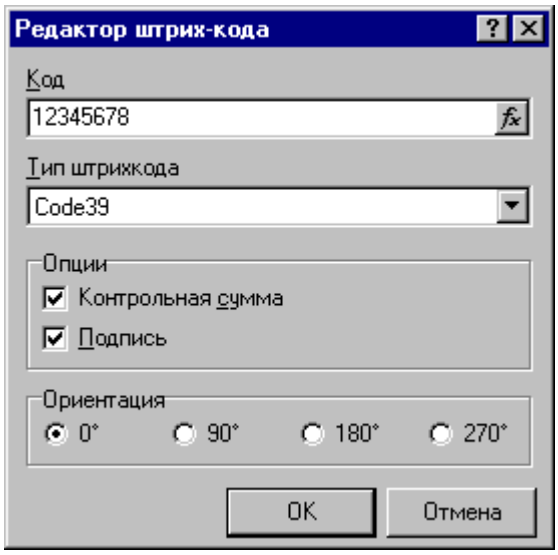
Предназначен для вставки геометрических фигур (прямоугольник, прямоугольник с закругленными углами, эллипс, треугольник) в документ. Тип фигуры можно установить в инспекторе объектов (свойство Shape).

Для использования этого объекта в run-time необходимо подключить компонент TfrShapeObject из палитры компонент FastReport.

Объект «Штрихкод»

Предназначен для вставки штрихкодов в документ. Можно использовать следующие типы штрихкодов:

- 2 of 5 interleaved
- Code39
- Code39 Extended
- Code128A-C
- Code93
- Code93 Extended
- MSI
- PostNet
- Codebar
- EAN8
- EAN13
- EAN128A-C
- UPC A, E0, E1, Supp2, Supp5



Для использования этого объекта в run-time необходимо подключить компонент TfrBarcodeObject из палитры компонент FastReport.

Объект RichText 2.0

Объект аналогичен объекту RichText, за исключением того, что он базируется на компоненте RXRichEdit, который поддерживает формат RTF 2.0 (RichText с OLE-вставками и картинками). Использование данного объекта возможно при установленной библиотеке

компонентов RXLib версии 2.6 или позже. Для этого в файле FR.inc необходимо снять комментарий со строки {$DEFINE RX} и переустановить пакеты FastReport.

Для использования этого объекта в run-time необходимо подключить компонент TfrRXRichObject из палитры компонент FastReport.

Построение отчетов

Для создания отчета предназначен компонент TfrReport из палитры компонентов FastReport. Это основной компонент, позволяющий загружать и сохранять отчеты, выполнять построение отчета, его печать и предварительный просмотр. Двойной щелчок мышью на компоненте вызывает дизайнер отчета.

Подавляющее большинство отчетов содержит информацию, хранящуюся в таблицах баз данных. Для доступа к таким данным FastReport использует стандартные делфийские компоненты доступа к БД – TTable или TQuery . Вообще говоря, можно использовать в этих целях любые компоненты – наследники TDataSet.

Для представления многострочной части отчета (т.е. данных из нескольких записей таблицы БД, к примеру) предназначены бэнды (band – полоса (англ.)). Бэнды в FastReport (да и в любом другом бэнд-ориентированном генераторе отчетов) подразделяются на две категории: дата-бэнды, которые собственно образуют многострочную часть отчета, и служебные бэнды (заголовок отчета, заголовок страницы и пр.). Для дата-бэнда необходимо назначать источник данных. В качестве такого источника выступают компоненты TfrDBDataset и TfrUserDataset из палитры компонентов FastReport. Компонент TfrDBDataset, по сути, является "переходником" для подключения к TDataSet. Компонент TfrUserDataset предназначен для навигации по пользовательскому набору данных, в роли которого может выступать массив, содержимое StringGrid и пр.

Таким образом, перед построением отчета необходимо:

а) поместить на форму компонент TfrReport ;

б) поместить компоненты доступа к данным TTable , TQuery или им подобные, если отчет будет использовать данные из БД;

в) для каждого дата-бэнда, который будет использован в отчете, необходимо создать соответствующий источник данных (TfrDBDataset или TfrUserDataset) и настроить его свойства.

Бэнды в FastReport

В FastReport имеется 22 бэнда. Полный их перечень приведен в таблице.

Название	Функция
Report title	печатается один раз в начале отчета
Report summary	печатается один раз в конце отчета
Page header	печатается вверху на каждой странице
Page footer	печатается внизу на каждой странице
Master header	печатается в начале списка 1-го уровня
Master data	данные списка 1-го уровня
Master footer	печатается в конце списка 1-го уровня
Detail header	печатается в начале списка 2-го уровня
Detail data	данные списка 2-го уровня
Detail footer	печатается в конце списка 2-го уровня
Subdetail header	печатается в начале списка 3-го уровня
Subdetail data	данные списка 3-го уровня
Subdetail footer	печатается в конце списка 3-го уровня
Overlay	печатается на каждой странице нижним слоем
Column header	печатается в начале каждой колонки
Column footer	печатается в конце каждой колонки
Group header	заголовок группы
Group footer	печатается после группы
Cross header Cross data Cross footer	эта группа бэндов предназначена для создания отчетов с переменным количеством столбцов и разбивкой на страницы
Child	может быть прикреплен к любому из бэндов, кроме Page footer, и выводиться вместе с ним.

Как видно, набор бэндов отличается от "классической" схемы, которой, в частности, придерживается ReportBuilder (здесь и далее имеется в виду ReportBuilder 4.2). По этой схеме, сам отчет имеет источник данных и является списком первого уровня. Список второго уровня образуется бэндом Detail. Таким образом, классическая схема позволяет строить списки с

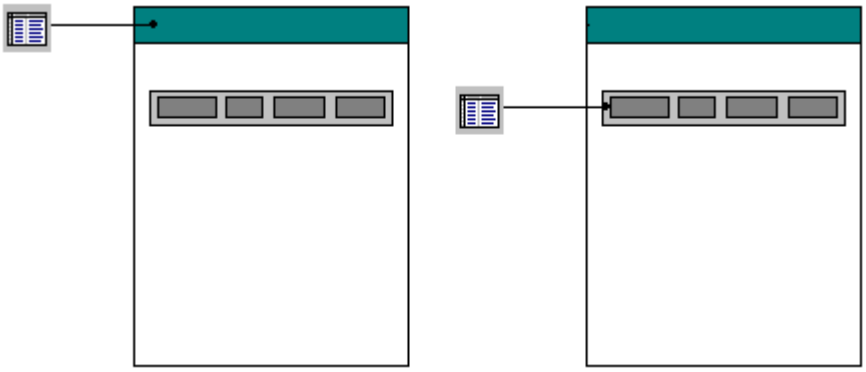
двумя уровнями детализации. Если нужно больше уровней, применяется компонент "Вложенный отчет", который, по сути, представляет собой новый отчет со своим набором бэндов. Этот же компонент применяется для организации отчетов вида master-detail-detail или master-master, т.к. классическая схема позволяет размещать на листе отчета только один бэнд определенного типа.

Схема, принятая в FastReport, позволяет в большинстве случаев обходиться без вложенных отчетов. Как видно из таблицы, для организации многоуровневых списков имеется три бэнда: Master data, Detail data и Subdetail data. Кроме того, каждый из этих бэндов имеет заголовок (header) и подвал (footer). Каждому дата-бэнду назначается источник данных, сам же отчет не имеет источника. Это дает возможность строить списки с тройной вложенностью, если нужно больше - применяется вложенный отчет.

Индивидуальное назначение источников данных дата-бэндам дает возможность размещать на одной странице отчета несколько однотипных бэндов, и, как следствие, строить отчеты типа master-detail-detail, master-master и им подобные, без каких-либо ухищрений.

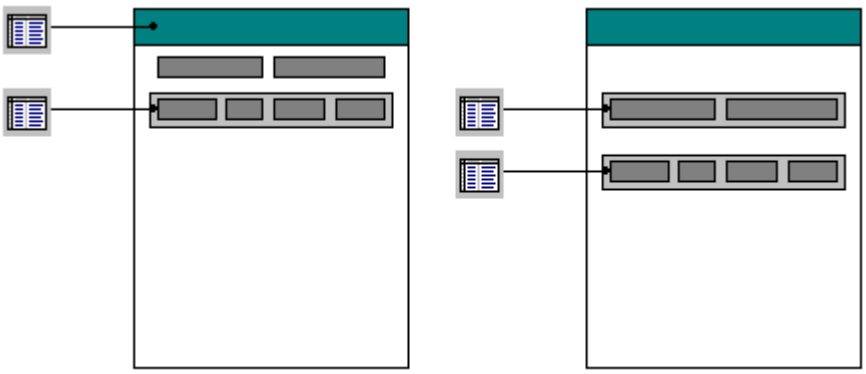
Для иллюстрации вышесказанного приведем примеры построения некоторых отчетов в FastReport и ReportBuilder.

Пример 1. Отчет с одним уровнем данных.



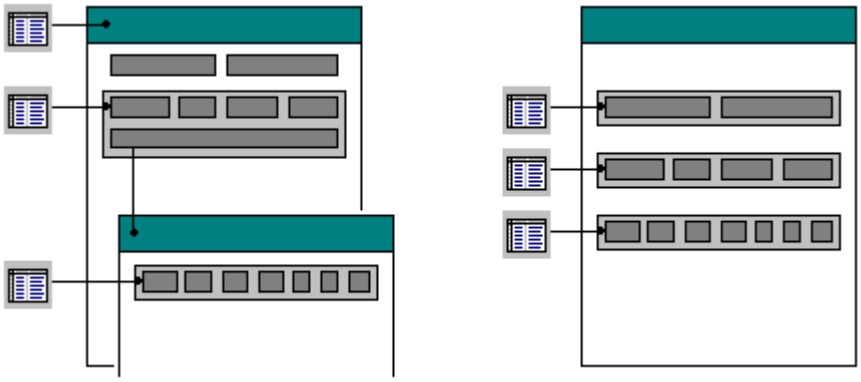
ReportBuilder FastReport

Пример 2. Отчет с двумя уровнями данных.



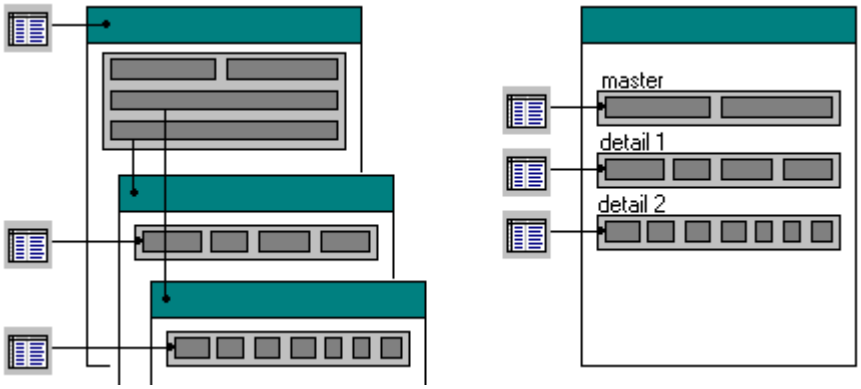
ReportBuilder FastReport

Пример 3. Отчет с тремя уровнями данных.



ReportBuilder FastReport

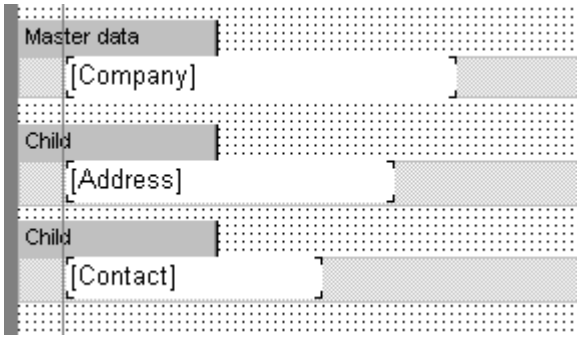
Пример 4. Отчет типа master-detail-detail.



ReportBuilder FastReport

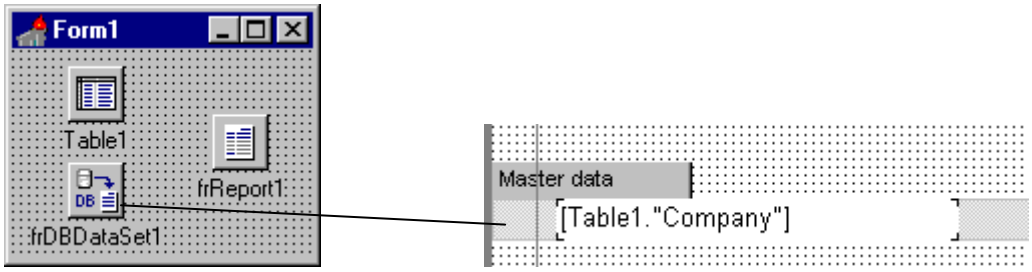
Как видим, примененная в FastReport схема дает большую гибкость при построении отчетов. Кроме того, FastReport также позволяет назначать объекту "отчет" источник данных. Для этого у объекта TfrReport имеются соответствующие свойства: ReportType (rtSimple, rtMultiple) и DataSet: TfrDataSet. Если свойство ReportType = rtSimple (по умолчанию), то бэнду master data можно не назначать источник данных – он будет взят из свойства DataSet объекта TfrReport. В противном случае (ReportType = rtMultiple) отчет будет последовательно построен столько раз, сколько записей в наборе данных, подключенных к свойству DataSet. Это удобно использовать для печати нескольких копий отчета, если надо каким-либо образом помечать копии (например, на первой копии вывести надпись "Основная", на остальных – "Для ознакомления").

Кроме основных бэндов, в FastReport имеется два вспомогательных бэнда, не входящих в "классическую" схему - Overlay и Child. Эти бэнды позаимствованы из QuickReport. Бэнд Overlay применяется для печати фоновых рисунков (или других объектов), которые выводятся на каждой странице отчета самым нижним слоем; бэнд Child может быть прикреплен к любому бэнду, кроме Page footer, и выводиться после него. К бэнду Child также можно прикреплять свои Childs. Он предназначен в основном для печати многосекционных бэндов (т.е. бэндов, состоящих из нескольких секций). Каждая секция образована одним child-бэндом и может иметь переменную высоту в зависимости от высоты находящихся в секции объектов.


Master data
[Company]
Child
[Address]
Child
[Contact]

Отчет с одним уровнем данных (список)

Это самый простой вид отчета. Для его создания необходимо разместить на листе бэнд "master data" и поместить в него объекты со ссылками на нужные данные. Бэнд необходимо подключить к источнику данных TfrxxxDataset.



Отчет с двумя уровнями

Для построения этого отчета необходимо разместить на листе бэнды "master data" и "detail data", затем поместить в них объекты со ссылками на нужные данные. Следует отметить, что таблицы (или запросы), на которых строится такой отчет, должны быть связаны отношением master-detail.

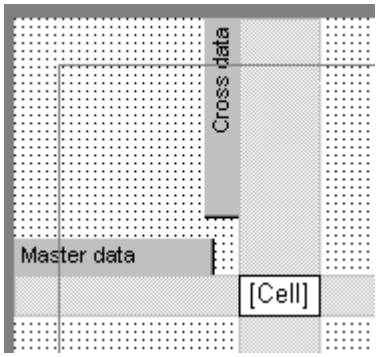
Порядок, в котором бэнды размещаются на листе, не имеет значения - сначала будет напечатана строка из master, потом соответствующие ей строки из detail. Отчет будет формироваться до тех пор, пока есть данные для master. Если надо, чтобы master с его подчиненным списком печатался с нового листа, установите для бэнда флажок "Формировать новую страницу" (FormNewPage). У отчета есть особенность - если для текущей строки из master список detail пуст, строка master печататься не будет. Если все же необходимо распечатать такие строки, для бэнда "master data" нужно установить флажок "Печатать, если detail пуст" (PrintIfSubsetEmpty).

Отчет с тремя уровнями

Для построения отчета разместите на листе бэнды "master data", "detail data" и "subdetail data". В остальном принципы построения такие же, как и для отчета с двумя уровнями детализации.

Cross-tab отчет

Этот вид отчета предназначен для печати таблицы, число столбцов которой на этапе построения отчета неизвестно. При формировании отчета столбцы, не поместившиеся на странице, будут перенесены на следующую. Для построения этого отчета разместите на странице бэнд "master data" и бэнд "cross data". Бэнды crossXXX размещаются на странице вертикально. На пересечении бэндов поместите объект, который будет отображаться в качестве ячейки таблицы. Необходимо поместить объект таким образом, чтобы он полностью помещался в границы области пересечения бэндов.



Это все, что нужно для построения базового отчета. Осталось назначить источники данных для "master data" и "cross data". Если бэнд "cross data" пересекает несколько обычных бэндов, то необходимо назначить источники данных для каждого из бэндов. Как видно, для построения cross-tab отчета необходимо как минимум два набора данных, связанных между собой отношением master-detail.

Если заглянуть в демо (отчет Cross-tab), то видно назначение бэндов "cross header", "cross footer": footer печатается последним столбцом (например, если нужно отпечатать сумму по строке), а header - первым. Кроме того, если в опциях бэнда "cross header" включен флажок "Выводить на всех страницах", то при переносе широкой таблицы на смежный лист header будет продублирован.

Отчет этого типа может иметь переменную высоту строки данных. Для этого надо включить опцию "растягиваемый" у бэнда "master data". При этом формирование кросс-таблицы будет происходить в два прохода: на первом проходе определяется максимальная высота строки, на втором выполняется вывод данных.

Текущая версия FastReport не позволяет строить cross-tab отчеты со сложными заголовками (как в ReportBuilder). Эта возможность будет доступна в одной из следующих версий FastReport.

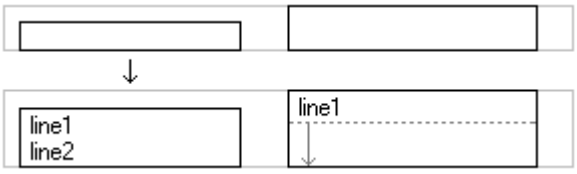
Динамические отчеты

В динамических отчетах высота бэндов зависит от высоты находящихся в них объектов. Объектами, которые могут растягиваться в зависимости от количества имеющегося в них текста, являются объекты "Текст", "Текст с тенью" и "RichText".

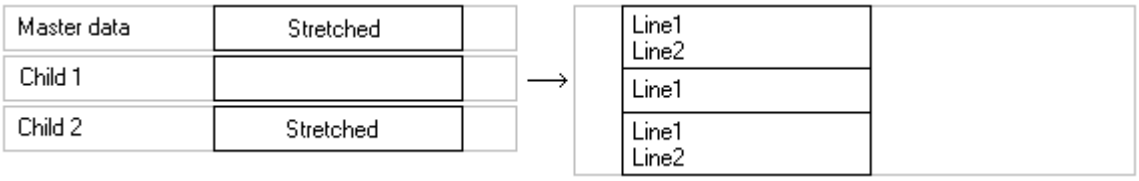
Чтобы сделать объект растягиваемым, надо включить его флажок "Растягиваемый" (Stretched). *Кроме того, этот же флажок надо включить у бэнда, на котором расположен объект.* При печати растягиваемого бэнда FastReport считает максимальную высоту объектов, находящихся на бэнде, и устанавливает высоту бэнда равной максимальной высоте объектов.

Растягиваемый объект и объект с фиксированной высотой.

Если на бэнде несколько растягиваемых объектов, расположенных рядом, то при растягивании нижние границы всех объектов (независимо от их актуальной высоты) будут совпадать с нижней границей бэнда:



А как быть, если растягиваемые объекты нужно поместить друг под другом, либо под растягиваемым объектом поместить нерастягиваемый? В ReportBuilder для этой цели служат регионы и опции объектов StretchWithParent и ShiftWithParent. В FastReport для этого применяется бэнд Child. Объекты, находящиеся друг под другом, один или несколько из которых растягиваемые, нужно разнести по разным бэндам:

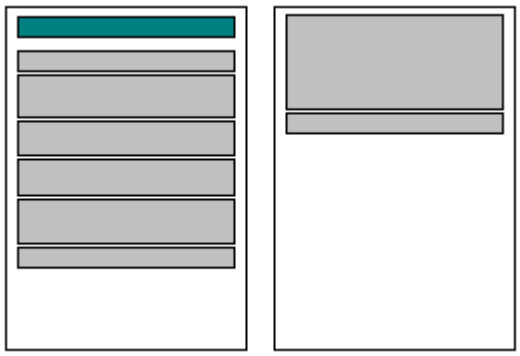


При этом всем Child - бэндам надо также установить флажок "Растягиваемый". Для связывания бэндов служит свойство "ChildBand" в инспекторе объектов. В вышеприведенном примере нужно связать Master data с Child 1 и Child 1 с Child 2.

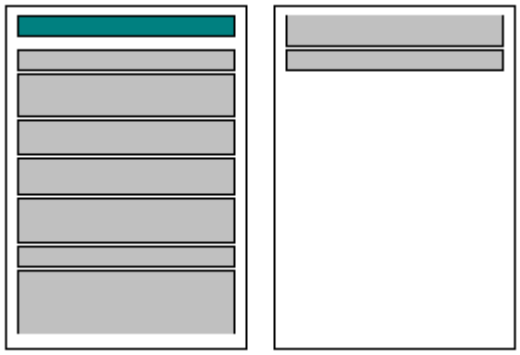
Следует отметить, что все объекты FastReport имеют рамку. С помощью рамок отчету можно легко придать табличный вид. При растягивании объекта рамка растягивается вместе с ним. Это очень удобно - так, в ReportBuilder текстовые объекты рамки не имеют и ее приходится делать с помощью объекта "Shape". Это связано с рядом трудностей - нужно ровно разместить рамку относительно объекта. Кроме того, при печати не всегда удается состыковать рамки соседних объектов. В QuickReport рамка присутствует, но при растягивании объекта не растягивается вместе с ним.

Разрываемые бэнды

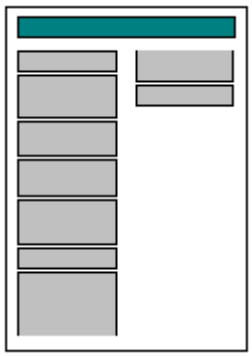
При печати бэнда FastReport проверяет наличие свободного места на странице, и, если оно меньше необходимого, формирует новую страницу и печатает бэнд на ней. При этом место на странице расходуется неэкономно, особенно в случаях, когда высота бэнда достаточно большая:



Для того, чтобы вывести на листе столько строк текста, сколько помещается, бэнд должен иметь помимо включенной опции "Растягиваемый" опцию "Разрываемый" (Breaked). При этом отчет будет выглядеть примерно так:



Если отчет многоколоночный, то содержимое объектов будет переноситься на следующую колонку:



Разрывать содержимое могут объекты "Текст", "Текст с тенью" и "RichText". Все остальные объекты будут выведены на той странице, где для этого достаточно свободного места.

Многоколоночный отчет

Если в обычном отчете при достижении конца страницы формирование продолжается с нового листа, то в многоколоночном - на том же листе, но рядом, т.е. в несколько смежнорасположенных колонок. Количество колонок задается в опциях страницы. Обычный отчет можно превратить в многоколоночный, всего лишь задав количество колонок в опциях страницы. Бэнды "column header" и "column footer" позволяют снабдить каждую колонку заголовком и нижней частью.

Кроме того, в FR2.4 количество колонок можно задавать отдельно для каждого дата-бэнда. Если установить свойство Columns > 1 у бэнда, то вывод строк данных будет происходить слева направо и сверху вниз. Можно также указать ширину колонки и промежуток между колонками. Бэнд отображает текущие установки в дизайнере штриховыми линиями.

Отчет с титульным листом

FR позволяет легко делать многостраничные отчеты (например, первая страница - титульный лист, все остальное - прочая информация). Для того, чтобы создать/удалить лист, используйте кнопки на панели инструментов либо контекстное меню, появляющееся при нажатии правой кнопки мыши на закладке страницы.

По сути, многостраничный отчет представляет собой несколько отчетов в одном. На каждой странице может содержаться собственный отчет, со своим наборов бэндов и

настройками страницы. Для построения аналогичного отчета в QR применяется композитный отчет, в ReportBuilder - компонент "вложенный отчет".

Содержимое страницы может быть выведено на свободном месте предыдущего листа, если в настройках страницы включена опция "Печать на предыдущем листе" (свойство PrintToPrevPage в инспекторе).

Вложенный отчет

Вложенный отчет - это отчет с размещенным на нем объектом SubReport , который представляет собой ссылку на другой отчет, расположенный на отдельной странице. При формировании такого отчета вместо объекта SubReport будет выведен соответствующий отчет. При вставке объекта в отчет автоматически добавляется новая страница с названием "SubReport xx".

Объекты SubReport можно располагать на листе друг под другом или рядом. В случае, если объекты надо расположить друг под другом, их необходимо разнести по разным бэндам (например, первый отчет - в бэнде "master data", второй - в бэнде "child").

На использование SubReports накладываются следующие ограничения:

- нельзя использовать колонки;
- во вложенных отчетах (не в основном!) игнорируются бэнды ReportTitle, ReportSummary, PageHeader, PageFooter, ColumnXXX;
- нельзя использовать разрываемые бэнды;
- нельзя использовать группы.

Master-Detail-Detail отчет

Для построения этого отчета разместите на листе бэнд "master data" и два бэнда "detail data". Для бэндов "detail data" назначьте соответствующие источники данных.

Разумеется, таким образом можно строить и отчеты типа Master-Master, Master-Detail-SubDetail-SubDetail. Отчеты типа Master-Detail-Master-Detail строить таким способом нельзя - нужно разнести раные master-detail по отдельным страницам (см. отчет с титульным листом). Количество одинаковых дата-бэндов на листе может быть до 32.

Композитный (составной) отчет

Композитный отчет позволяет получить отчет, состоящий из нескольких других. Для этого разместите на форме компонент TfrCompositeReport , и программным путем заполните его свойство Reports ссылками на отчеты, которые нужно объединить в один. Пример использования см. в демо. Отчеты будут формироваться по порядку, причем если страница отдельного отчета имеет включенную опцию "Печать на предыдущем листе", то она будет напечатана начиная со свободного места на последнем листе предыдущего отчета.

Отчет с BLOb-полями

Если в объекте необходимо отобразить информацию из Blob-поля (мемо, картинка, OLE и пр.), в мемо объекта вставьте переменную, которая ссылается на соответствующее поле из БД либо название самого поля, например: [Table1."BlobFld"]. Эту же операцию можно выполнить в инспекторе объектов, заполнив свойство объекта "DataField".

Если нужно отобразить аналогичную информацию, но из другого источника, то эту операцию следует выполнять программно в обработчике события OnBeforePrint.

Отчет без бэндов

Часто бывает необходимо отобразить данные только из одной записи БД, разместив их на бланке. Пример - платежное поручение, программку для печати которого писал, наверное, каждый. В этом случае отчет может вообще не содержать бэндов. Объекты, расположенные непосредственно на листе, при печати выведутся на своих местах.

Отчет с группами

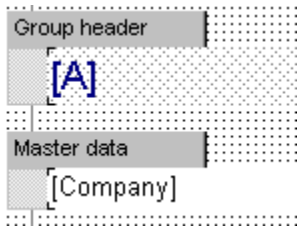
Группы применяются для группировки строк данных с использованием некоторого критерия. В качестве критерия обычно выступает выражение, основанное на полях из набора данных. При формировании отчета FR следит за значением критерия, и, как только оно изменится, формирует новую группу.

Для построения отчета разместите на листе бэнды "group header" и "master data". Назначьте источник данных для "master data", затем в редакторе бэнда "group header" укажите условие группировки. Например, чтобы распечатать список сотрудников в группированном виде (группировка по первой букве фамилии), надо указать примерно такое условие группировки: Copy([Фамилия], 1, 1). При изменении этого условия будет сформирована новая группа.

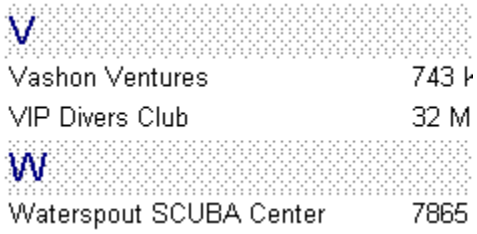
На использование групп накладываются следующие ограничения:

- нельзя применять группы во вложенных отчетах;
- группа может выступать только в роли списка 1-го уровня, т.е. сначала идет группа, затем - master, detail и т.п.

Исходный отчет



Сформированный отчет



Примечание. Для правильной работы групп исходный набор данных должен быть уже отсортирован по условию группировки. Сортировку можно осуществить, применяя запрос (Query) с выражением ORDER BY.

Отчет с диаграммами

Предположим, имеется отчет вида "Клиент - сумма заказа":

Action Club	$1000
Action Diver Supply	$12000

Adventure Undersea $5000

Для построения диаграммы по этим данным необходимо разместить объект "Диаграмма" в бэнде "report summary" (или любом другом, который печатается после вывода всех необходимых данных). В редакторе объекта нужно выбрать тип диаграммы (один из шести возможных - линии, точки, столбцы горизонтальные и вертикальные, области и круговые диаграммы). Для связывания диаграммы с данными необходимо указать имена объектов, которые содержат эти данные (в нашем случае это объект, содержащий строку наименования клиента и объект с суммой). Имя объекта можно увидеть в Object inspector.

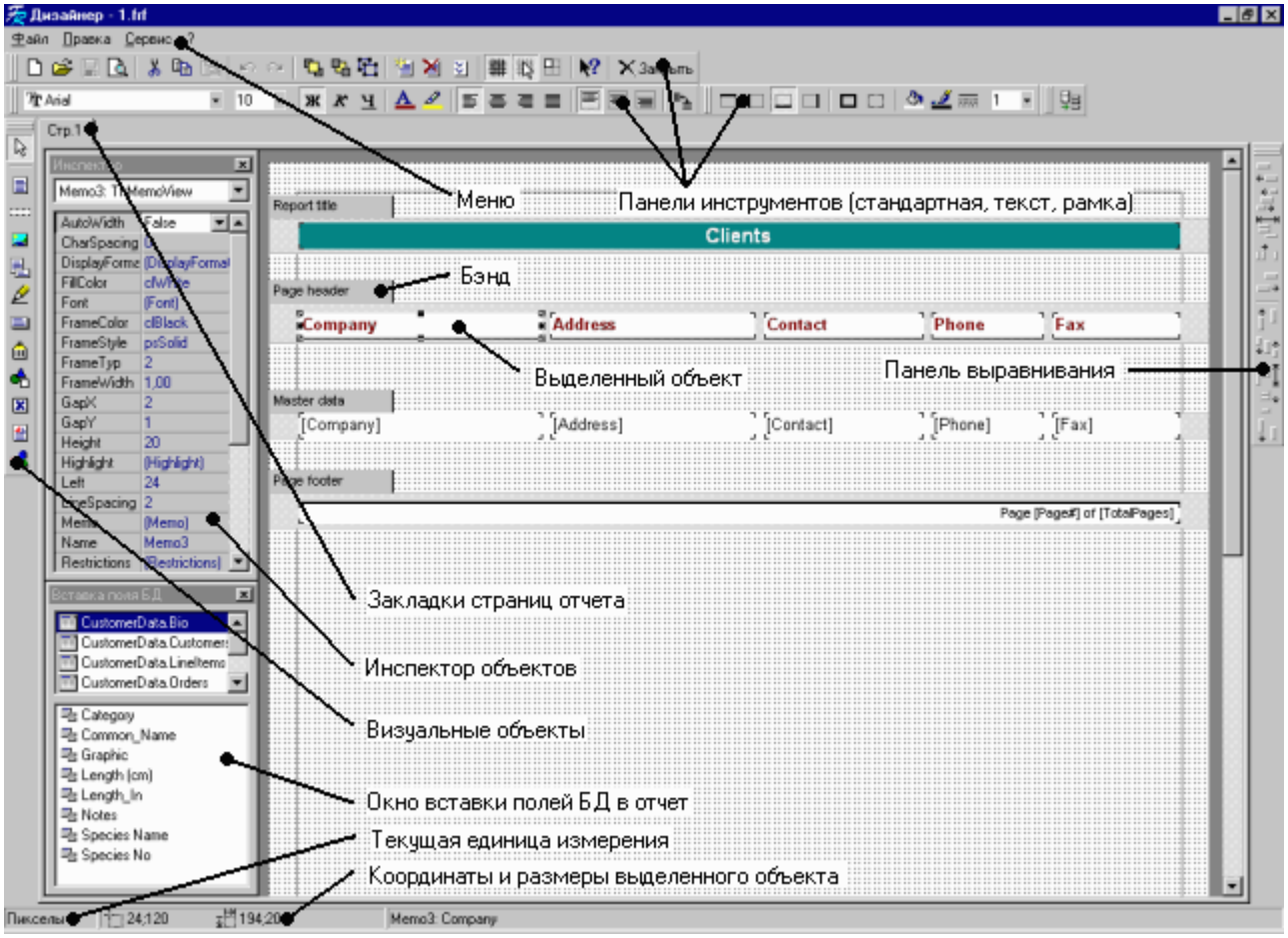
Дизайнер

Дизайнер

Дизайнер

Компонент снабжен встроенным дизайнером, который можно вызвать в design-time двойным щелчком мыши на компоненте TfrReport. Дизайнер предоставляет пользователю удобные средства для разработки внешнего вида отчета и позволяет сразу выполнить предварительный просмотр. Интерфейс дизайнера выполнен на современном уровне с использованием панелей инструментов (toolbars), расположение которых можно изменять по своему вкусу. Информация о расположении панелей запоминается в реестре, и при следующем запуске восстанавливается. Также в реестре запоминаются остальные настройки дизайнера.

Дизайнер доступен из среды Delphi в design-time. Для использования дизайнера в скомпилированном проекте необходимо использовать компонент TfrDesigner из палитры компонентов FastReport, либо включить в список `uses` модуль FR_Desgn. Использование дизайнера в run-time дает возможность пользователю настраивать вид отчета, а также редактировать готовый отчет.



Клавиши управления

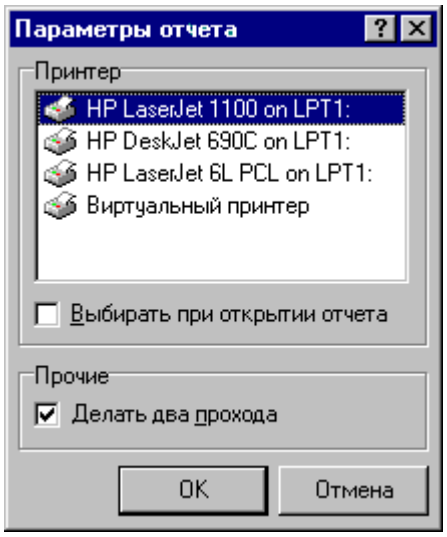
- Стрелки - перемещение по объектам.
- Ctrl + Стрелки - плавное перемещение выделенного объекта (объектов).
- Shift + Стрелки - уменьшение/увеличение размеров объекта.
- Alt + Стрелки - склеивание объекта с ближайшим в выбранном направлении.
- Enter - вызов редактора объекта.
- Delete - удаление объекта.
- Insert - показывает диалог вставки поля БД в отчет;
- Ctrl + Enter - вызов редактора мемо.
- Ctrl + 1..9 - установка толщины рамки объекта.
- Ctrl + Z - отменить последнее действие. Число уровней отката - до 100.
- Ctrl + Y - повторить отмененное действие.
- Ctrl + G - показать сетку.
- Ctrl + B, Ctrl + I, Ctrl + U - управление стилем шрифта - аналоги кнопок **Ж, К, Ч** на панели текста.
- Ctrl + F - обрамить выделенный объект (объекты).
- Ctrl + D - снять обрамление с объекта.
- Ctrl + X - вырезать в буфер.
- Ctrl + V - вставить из буфера.
- Ctrl + C - копировать в буфер.
- Ctrl + A - выделить все объекты на листе.
- Ctrl + N - новый отчет.
- Ctrl + O - открыть файл с отчетом.
- Ctrl + S - сохранить отчет в файле.
- Ctrl + P - предварительный просмотр отчета.

Управление мышью

- Левая кнопка - выбор объекта; вставка нового объекта; перемещение и изменение размеров объекта (объектов).
- Правая кнопка - контекстное меню объекта (объектов), над которым находится указатель мыши.
- Двойной щелчок левой кнопкой - вызов редактора объекта. Если двойной щелчок сделать на пустом месте листа, то вызывается диалоговое окно «Параметры страницы».
- Shift + левая кнопка - если объект уже выделен, то снимает с него выделение, иначе объект выделяется. Выделение остальных объектов не изменяется.
- Ctrl + левая кнопка - при нажатии и перемещении мыши рисуется рамка; после отпускания кнопки мыши выделяются все объекты, попавшие в рамку. Такого же эффекта можно добиться, если щелкнуть левой кнопкой мыши на пустом месте листа, и, не отпуская кнопки, потянуть мышь до нужной позиции.
- При перемещении бэнда его объекты перемещаются вместе с ним. Если это нежелательно, то перемещение следует выполнять при нажатой клавише Alt.
- Изменить масштаб выделенных объектов можно, потянув мышкой за красный квадратик на нижнем правом углу группы выделенных объектов.
- При приближении указателя мыши к границе двух соседних объектов вид указателя изменяется. Если при этом нажать левую кнопку и переместить указатель мыши, граница объектов также перемещается (горизонтальный сплиттинг).

Параметры отчета

Окно с параметрами отчета доступно из меню "Файл|Параметры отчета...".



В списке принтеров имеется "Виртуальный принтер". Он необходим в случае, если в системе нет установленных принтеров. При этом дизайнер и предварительный просмотр отчета функционируют нормально. Вы можете выбрать любой из форматов бумаги, поддерживаемой Windows (вплоть до формата A2), устанавливать ориентацию листа бумаги. Т.е. "виртуальный" принтер эмулирует функции реального принтера (разумеется, на нем нельзя печатать ;)

Другой случай, когда необходимо использовать виртуальный принтер - Вы разрабатываете форму отчета для принтера с форматом бумаги A3, а самого принтера у Вас нет (он есть у заказчика). В этом случае выбор виртуального принтера снимает проблему.

Флажок "Выбирать при открытии отчета" позволяет привязать отчет к одному из принтеров, установленных в системе. Это значит, что при открытии такого отчета принтером по умолчанию станет выбранный принтер. Это полезно в случае, если в системе имеется несколько разных принтеров - текстовые документы можно привязать к монохромному принтеру, документы с графикой - к цветному.

Если флажок "Делать два прохода" установлен, формирование отчета будет осуществляться в два этапа. На первом проходе отчет формируется, осуществляется его разбивка на страницы, но результат нигде не сохраняется. На втором проходе происходит обычное формирование отчета с сохранением результата в потоке.

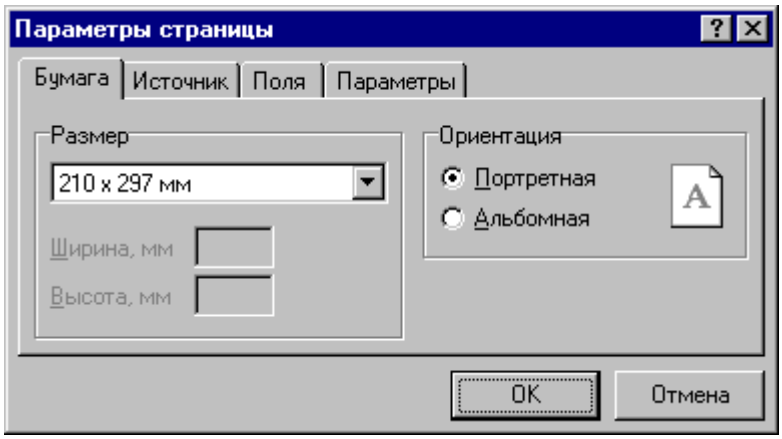
Для чего же нужно делать два прохода? Наиболее часто эта опция применяется в случае, если в отчете имеется упоминание об общем количестве страниц в нем, т.е. информация вида "Страница 1 из 15". Общее количество страниц подсчитывается на первом проходе и доступно через системную переменную TOTALPAGES. Наиболее частая ошибка - попытка применить эту переменную в однопроходном отчете, в этом случае она возвращает 0.

Другая область применения - выполнение каких-либо вычислений на первом проходе и отображение результатов на втором. Например, в случае, когда необходимо отобразить в заголовке группы сумму, которая обычно подсчитывается и отображается в подвале группы. Такого рода вычисления связаны с использованием встроенного языка FR.

Параметры страницы

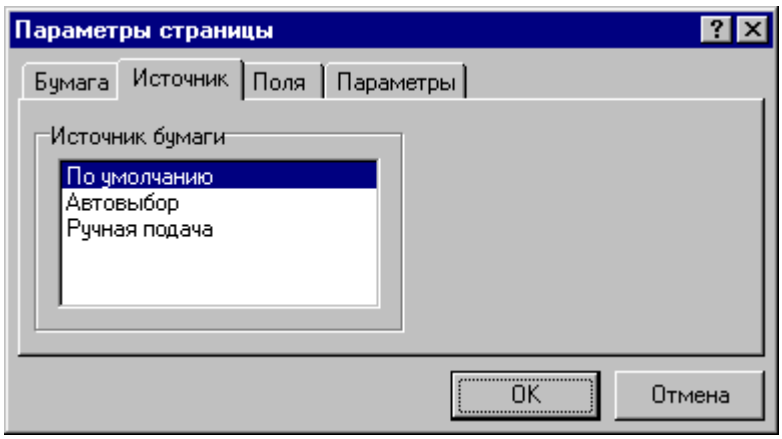
Окно с параметрами страницы доступно из меню "Файл|Параметры страницы..." либо при двойном щелчке левой кнопкой мыши на свободном месте листа отчета. Окно представляет собой блокнот с четырьмя закладками: "Бумага", "Источник", "Поля" и "Параметры".

Закладка "Бумага"

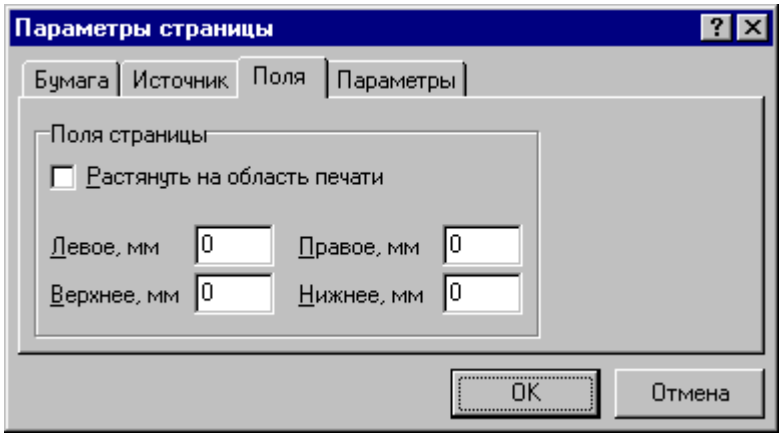


Поля ввода ширины и высоты становятся доступными, если из списка "Размер" выбран пункт "Размеры, определяемые пользователем". Следует отметить, что не все драйверы принтеров поддерживают пользовательский размер бумаги. Кроме того, не все принтеры поддерживают произвольные пользовательские размеры (например, драйвер принтера HP LaserJet 6L не дает устанавливать размеры листа меньше чем 76 * 127мм).

Закладка "Источник"



Закладка "Поля"



Здесь можно задать размеры отступа от краев листа, а также задать опцию «Растянуть на область печати». На этом моменте следует остановиться более подробно.

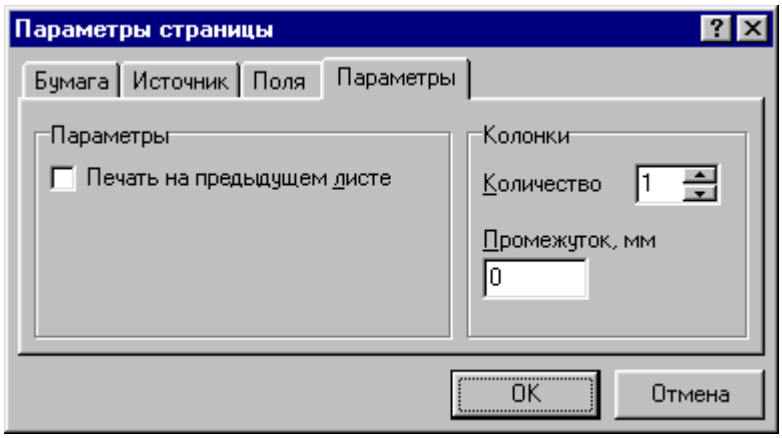
Возможны два варианта страницы в дизайнере: с полями печати и без них. Если поля печати включены (по умолчанию), то они отображаются в дизайнере и все, что находится вне полей, на печать не выводится. Это позволяет добиться максимального соответствия между видом страницы в дизайнере и на бумаге. Координаты объектов исчисляются от верхнего левого угла листа.

Но при переходе на другой принтер из-за изменившихся границ печати некоторые объекты могут вылезти за пределы полей и не будут напечатаны. Если включить флажок "Растянуть на область печати", то предполагается, что страница в дизайнере соответствует области печати, т.е. все, что находится на странице, будет корректно напечатано на любом принтере. Координаты объектов в этом случае исчисляются от верхнего левого угла области печати. Правда, при этом вид страницы в дизайнере будет немного отличаться от распечатанного документа.

Если поля страницы нулевые, то при построении отчета будут использована информация о полях, предоставляемая драйвером принтера. Если Вы используете матричный принтер, будьте внимательны. Дело в том, что во многих матричных принтерах одно или несколько полей имеет нулевой размер, в связи с чем при формировании отчета и при его распечатке может возникнуть ситуация, когда текст выходит за границы листа. Чтобы избежать этого, явно задавайте поля станицы либо устанавливайте флажок " Растянуть на область печати".

Следует отметить, что на формирование отчета при включенных полях страницы оказывает непосредственное влияние только установка значений для верхнего и нижнего полей. Левое же и правое поля действуют только для Cross-tab отчетов.

Закладка "Параметры"

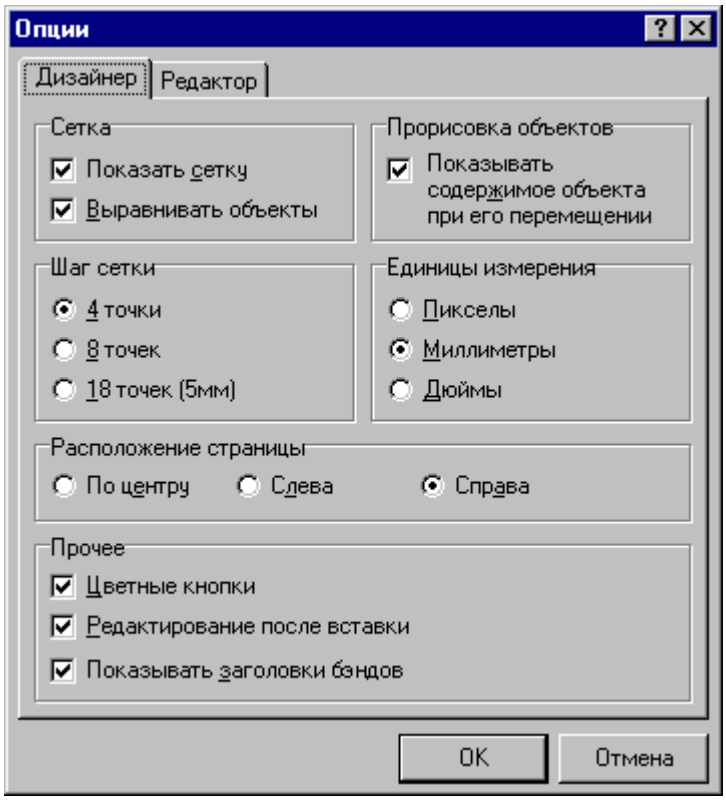


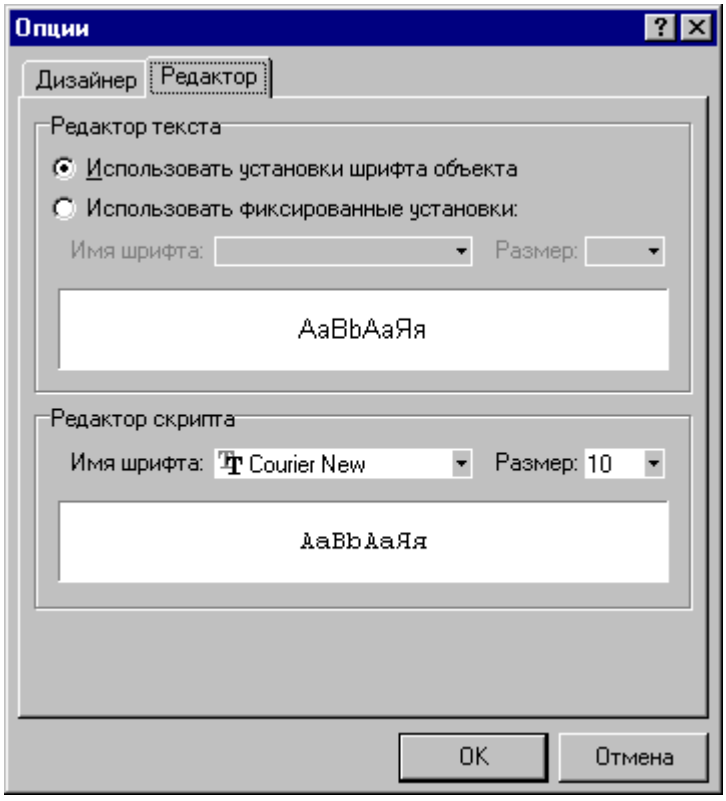
На этой закладке можно указать количество колонок для печати многоколоночных отчетов. Текущие установки отображаются в дизайнере.

Флажок «Печатать на предыдущем листе» позволяет начать печать страницы, начиная со свободного места на предыдущем листе. Эта опция может применяться в случае, если шаблон отчета состоит из нескольких листов либо при печати составных (composite) отчетов.

Опции дизайнера

Чтобы установить опции дизайнера, воспользуйтесь командой меню "Сервис|Опции...".





Здесь можно задать используемые единицы измерения (пикселы, миллиметры, дюймы), указать шаг сетки в точках (шаг сетки 18 точек точно соответствует 5мм. Это особо полезно в случаях, когда надо сделать бланк, размеры которого кратны 5мм. Например, платежное поручение делается таким образом за 10 минут).

Также можно указать способ прорисовки объектов при их перемещении (или изменения размеров).

Опция "Расположение страницы" позволяет выбрать расположение страницы. Это необходимо, если на экране присутствует инспектор объектов: таким образом можно задать взаимное расположение страницы и инспектора.

Если опция "Цветные кнопки" отключена, то все кнопки становятся черно-белыми; при наведении указателя мыши на кнопку она показывается в цвете.

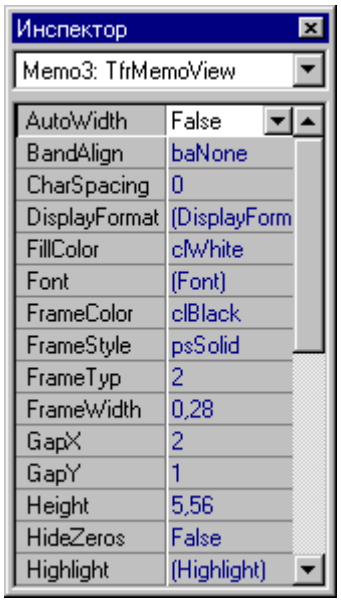
Опция "Показывать заголовки бэндов" позволяет отключать заголовки у бэндов в целях экономии свободного места на странице.

Наконец, опция "Редактирование после вставки" управляет процессом вставки новых объектов. Если опция включена, каждый раз при вставке объекта будет показываться его редактор. При вставке большого количества пустых объектов опцию лучше отключать.

На закладке "Редактор" можно настроить шрифт для окна редактора текста. При включенной опции "Использовать установки шрифта объекта" шрифт в окне редактора текста будет соответствовать шрифту редактируемого объекта.

Инспектор объектов

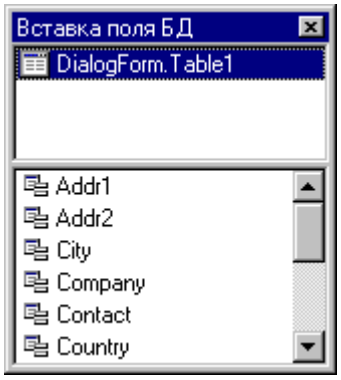
Инспектор позволяет манипулировать всеми свойствами выбранных объектов.



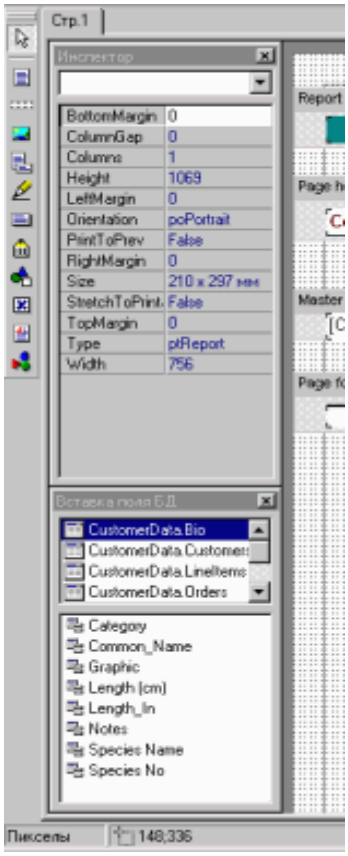
Окно инспектора похоже на Object Inspector в Delphi и функционирует аналогичным образом. Так же, как и остальные панели инструментов, его можно показать или скрыть соответствующей командой меню "Сервис|Панели инструментов". Показать окно можно также при нажатии на клавишу F11. Двойной щелчок на заголовке окна инспектора сворачивает окно в полоску, еще один двойной щелчок - восстанавливает исходные размеры.

Диалог вставки полей БД в отчет

Этот диалог позволяет быстро вставить поле БД в отчет. Он вызывается из меню "Сервис|Панели инструментов|Вставка поля БД" или при нажатии клавиши Insert.



Окно содержит два списка: список таблиц (запросов) в верхней части окна и список полей внизу. Вставка поля осуществляется методом drag&drop (перетащил и бросил). Для этого надо выделить мышкой нужное поле из списка внизу и, не отпуская кнопки мыши, перенести его на страницу отчета. При отпускании кнопки мыши на странице появится новый объект "Текст" со ссылкой на выбранное поле БД.

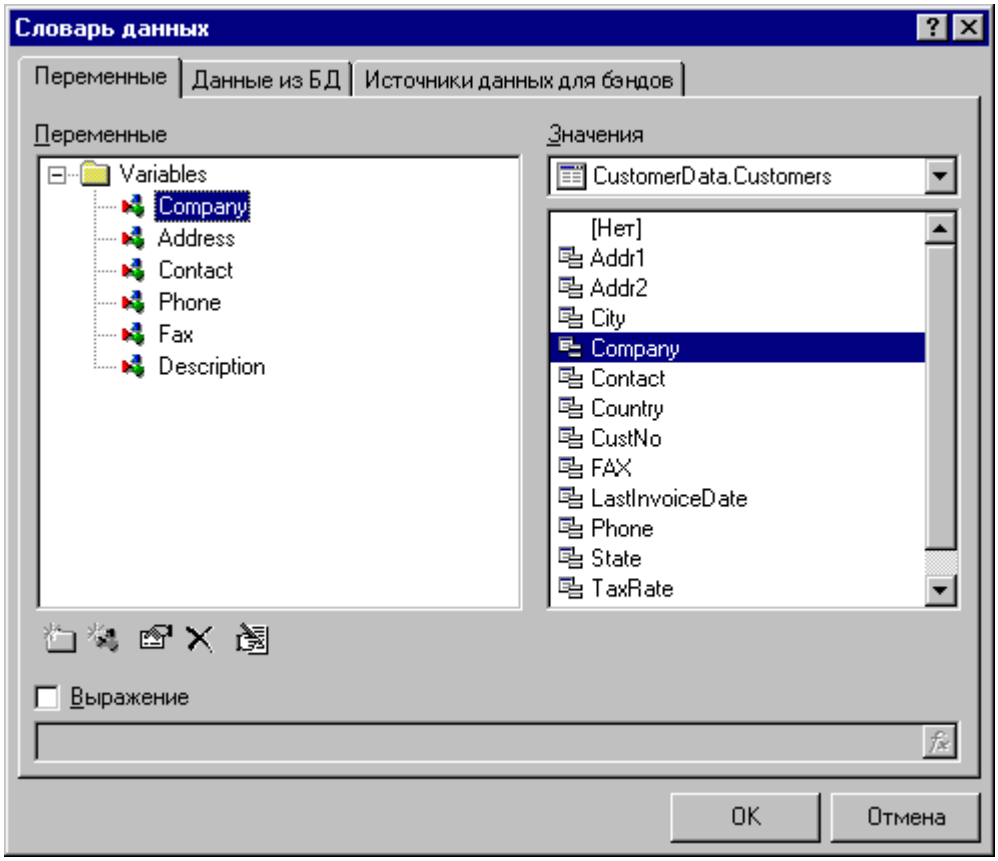


Если диалог используется часто, его удобно разместить под инспектором объектов или над ним. В этом случае двойной щелчок на заголовке окна диалога свернет окно в полоску и развернет окно инспектора; еще один двойной щелчок восстановит исходные размеры окна и уменьшит окно инспектора.

Словарь данных

Окно словаря данных (Data dictionary) доступно из меню "Файл|Словарь данных...". Окно представляет собой блокнот с тремя закладками: "Переменные", "Данные из БД" и "Источники данных для бэндов". Словарь данных хранится в файле вместе с формой отчета, но есть возможность хранить его в отдельном файле с расширением FRD и считывать при необходимости в существующий отчет. Для этого воспользуйтесь командами меню дизайнера "Файл|Открыть" и "Файл|Сохранить как..." и с появившихся стандартных диалогах открытия или сохранения файла выберите тип файла - "Словарь данных FastReport".

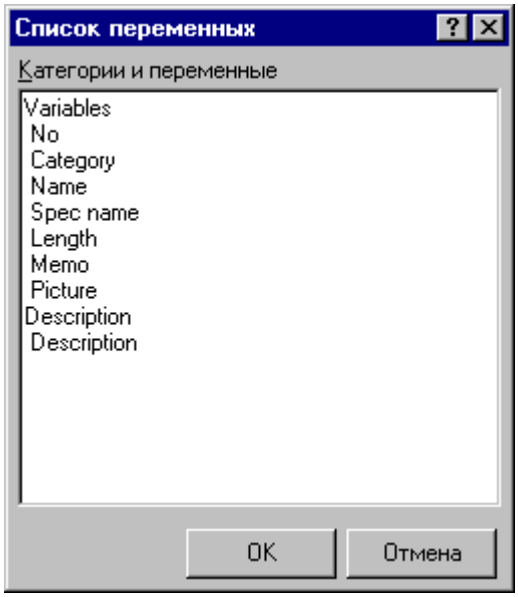
Закладка "Переменные"



Эта закладка предназначена для работы со списком переменных.

Список переменных расположен в левой части окна. Как видно из рисунка, структура списка двухуровневая: он состоит из категорий, в каждой категории может быть одна или несколько переменных. Категории нужны только для визуальной группировки переменных и в отчет не вставляются.

При создании нового отчета список пуст, и чтобы заполнить его, воспользуйтесь кнопками под списком: - добавляет новую категорию и - добавляет новую переменную к текущей категории. Эти действия можно также выполнить с клавиатуры: клавиша Insert добавляет новую переменную, Ctrl+Insert добавляет новую категорию. Назначение остальных кнопок следующее: кнопка редактирует название переменной или категории (клавиатурный аналог - клавиша Enter), кнопка удаляет переменную или категорию (клавиатурный аналог - клавиша Delete). Кнопка вызывает редактор списка переменных, где они представлены в виде списка строк. Здесь можно делать массовую вставку переменных из буфера обмена, переносить переменные из одной категории в другую.



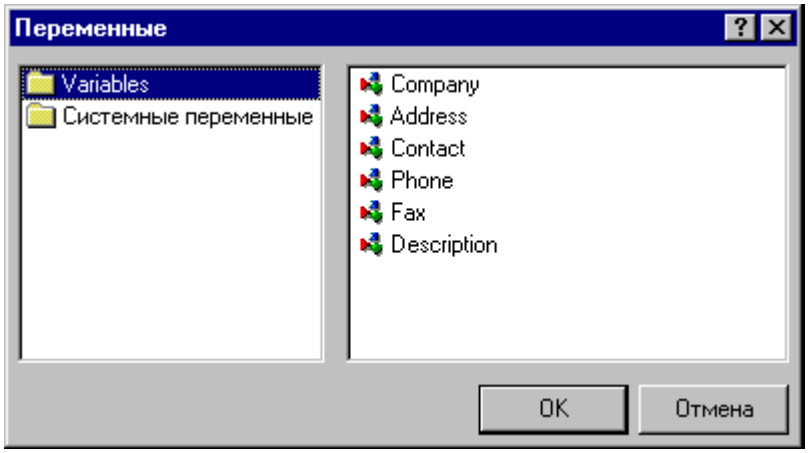
После того как список переменных определен, каждой переменной необходимо назначить значение. Для этого нужно выбрать переменную в списке слева и выбрать соответствующее значение в списке справа, пользуясь мышью.

В списке справа показаны все источники данных (т.е. невизуальные dbaware-компоненты, наследники TDataSet), которые доступны в данный момент, и имена их полей. В design-time будут доступны все датасеты, находящиеся в открытых модулях (units). В run-time будут доступны все датасеты, находящиеся в созданных формах или модулях данных (TDataModule). Кроме того, если выбрать из выпадающего списка справа значение «Системные переменные», то появится возможность сопоставить переменной одно из следующих значений:

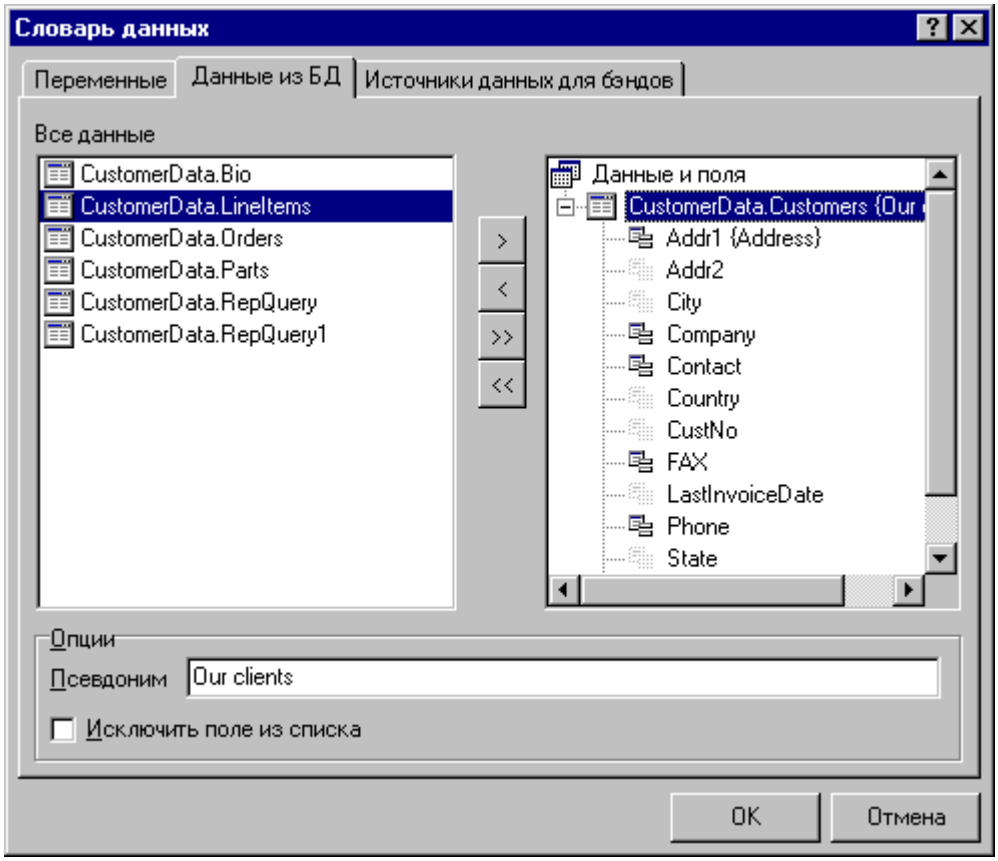
- страница - номер текущей страницы, соответствует функции Page# (см. описание встроенных функций);
- дата - дата начала формирования отчета, соответствует Date;
- время - время начала формирования отчета, соответствует Time;
- Строка, LineThrough#, Колонка, CurrentLine#, Всего страниц - см. описание встроенных функций.

Переменной можно также сопоставить выражение - для этого следует включить выбрать переменную и включить флажок "Выражение" в нижней части окна. При этом строка ввода выражения становится активной. Для визуального построения выражения нажмите кнопку fx.

Окно вставки переменных в отчет для вышеприведенного примера будет выглядеть следующим образом:



Закладка "Данные из БД"



На этой закладке в левой части представлен список всех доступных наборов данных - таблиц, запросов, имеющихся на всех формах проекта.

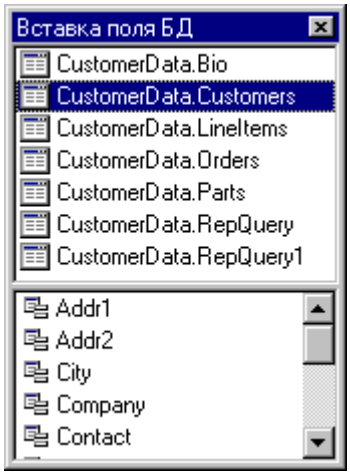
Назначение этого диалога простое: исключить ненужные данные из списка и дать им более полные названия (псевдонимы). Это необходимо потому, что отчет, как правило, содержит данные всего лишь из одного-двух запросов или таблиц. FastReport же по умолчанию предлагает вставлять поля из всех источников данных, которые он обнаружил на всех формах проекта. В больших проектах это десятки (а то и сотни) таблиц и запросов.

Чтобы использовать набор данных, надо перенести его из левого списка в правый. Сделать это можно либо с помощью оперции drag&drop, либо двойным щелчком мыши на нужном наборе данных, либо воспользовавшись кнопками в средней части окна.

Для исключения поля из списка выберите необходимый набор данных и его поле и нажмите на флажок "Исключить поле из списка". Повторное нажатие восстанавливает вхождение поля в список. Команда также доступна при нажатии на клавишу "Пробел".

Для задания псевдонима выберите необходимый элемент и в строке "Псевдоним" укажите новое название. Если псевдоним не нужен, сотрите содержимое строки. Команда также доступна при нажатии клавиши Enter.

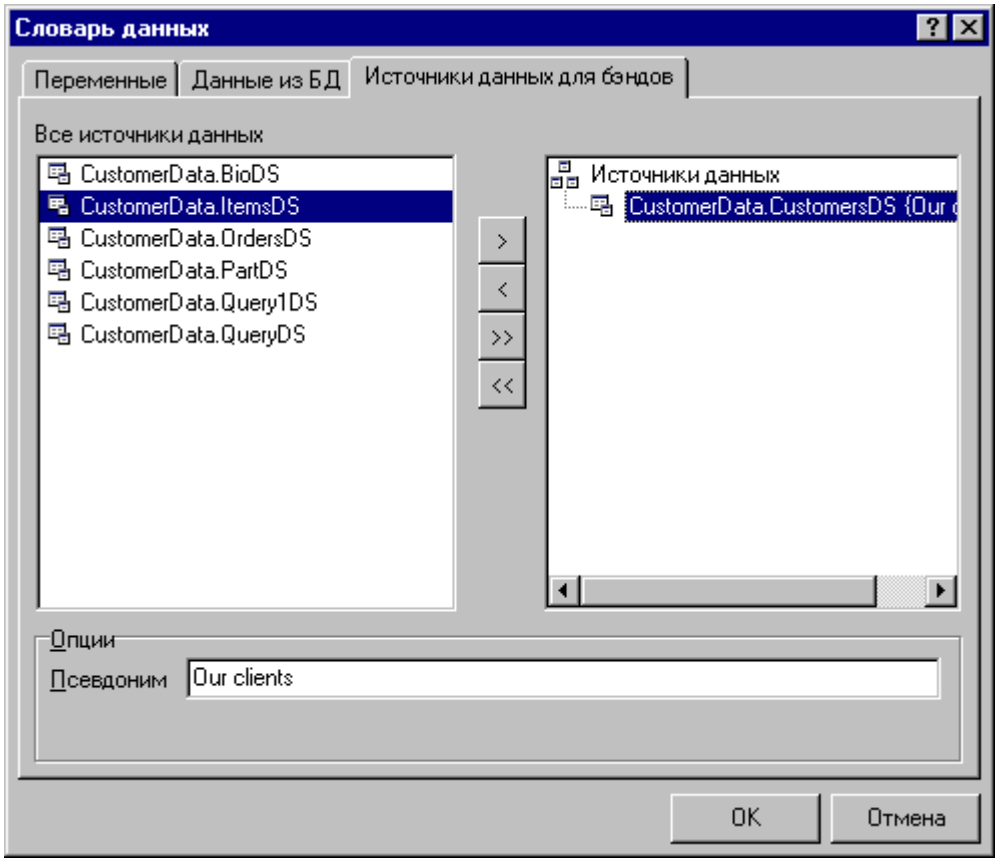
Как видно из рисунка, назначение псевдонимов источникам данных и их полям облегчает их использование при построении отчета. Сравните диалог вставки полей в отчет до применения псевдонимов и после:



Без псевдонимов

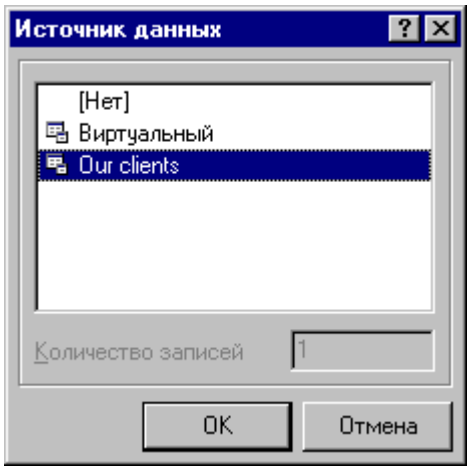
С использованием псевдонимов

Закладка "Источники данных для бэндов"



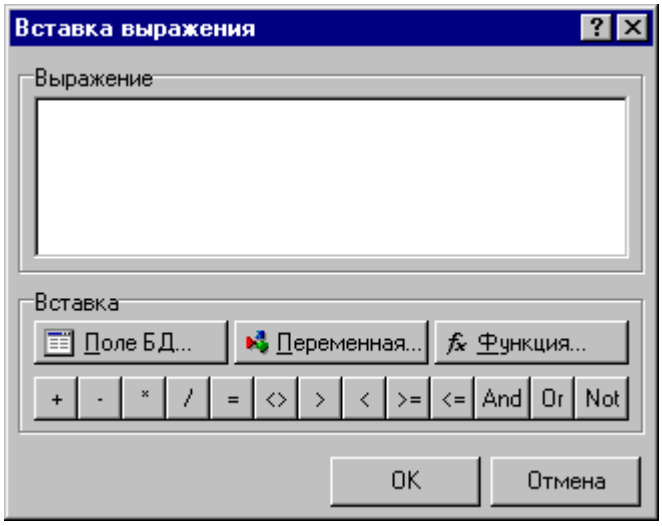
На этой закладке представлен список всех доступных источников данных для бэндов - компонентов TfrDBDataSet, TfrUserDataSet, имеющихся на всех формах проекта. Так же, как и

в предыдущем диалоге, здесь можно давать элементам более осмысленные имена (псевдонимы). Результат можно увидеть в диалоге выбора источника данных для дата-бэнда:



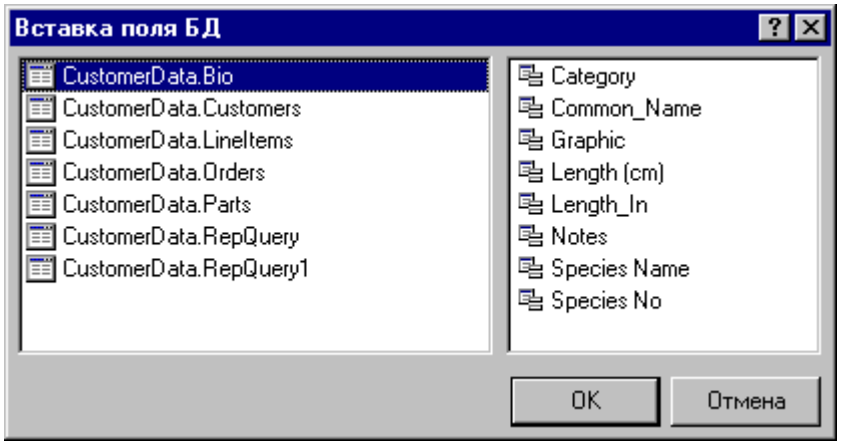
Конструктор выражений

Окно конструктора выражений можно вызвать из редактора текста, нажав кнопку на панели инструментов. Оно также доступно из некторых диалоговых окон, в которых требуется задать некоторое выражение (например, в редакторе условия группировки бэнда group header).



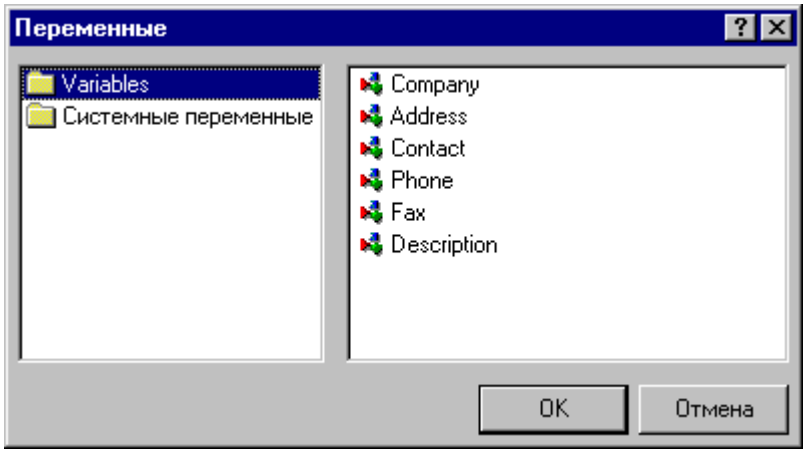
Окно содержит поле ввода выражения и кнопки вызова диалогов вставки переменных, полей БД, функций, а также кнопки быстрой вставки знаков арифметических и логических операций.

Диалог вставки поля БД



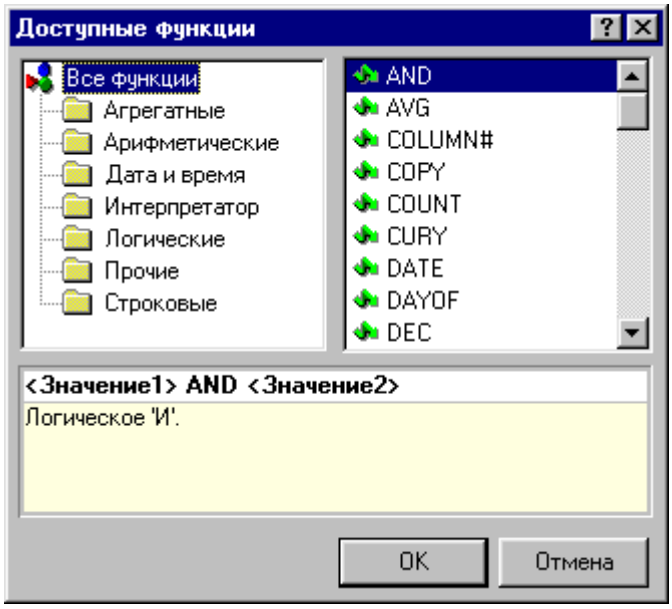
Диалог позволяет выбрать поле БД для вставки в выражение. Слева размещен список всех доступных таблиц БД, справа – список полей в выбранной таблице. Для вставки поля в выражение надо выбрать поле и нажать кнопку "ОК", либо сделать на поле двойной щелчок мышью.

Диалог вставки переменной

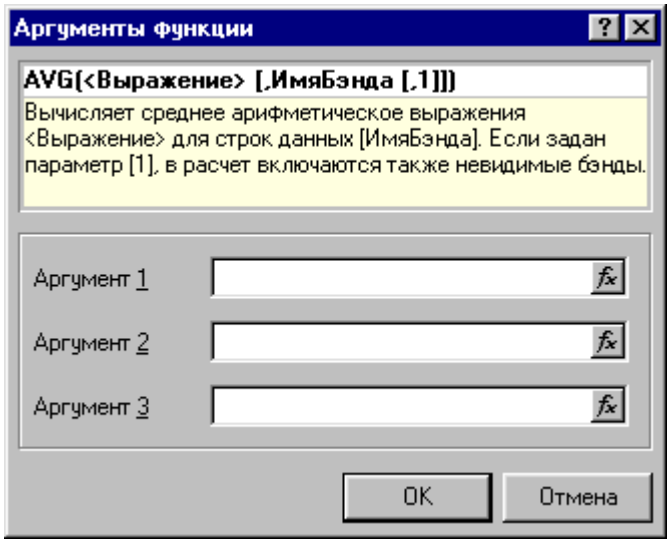


Диалог позволяет выбрать переменную для вставки в выражение. Слева размещен список категорий, справа – список переменных в выбранной категории. Для вставки переменной в выражение надо выбрать переменную и нажать кнопку "OK", либо сделать на ней двойной щелчок мышью.

Диалог вставки функции

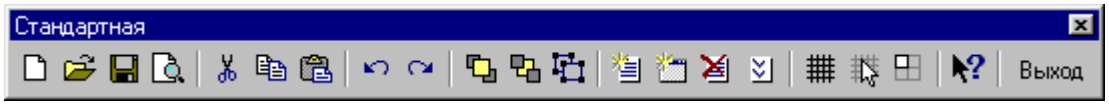


В этом диалоге можно выбрать функцию для вставки в выражение. При выборе функции в нижней части окна отображается краткое описание функции и списка ее аргументов. Если функция имеет аргуметны, то при при нажатии на кнопку "OK" будет предложено заполнить их:


Аргументы функции
AVG(<Выражение> [,ИмяБэнда [,1]])
Вычисляет среднее арифметическое выражения <Выражение> для строк данных [ИмяБэнда]. Если задан параметр [1], в расчет включаются также невидимые бэнды.
Аргумент 1
Аргумент 2
Аргумент 3
OK
Отмена

Панели инструментов

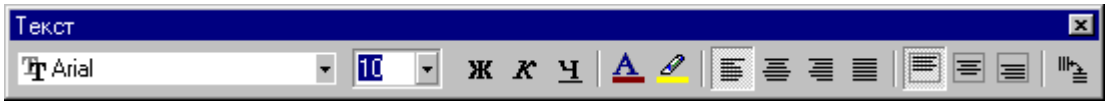
Панель инструментов "Стандартная"



Иконка	Название	Описание
	Новый отчет	Создает новый пустой отчет.
	Открыть отчет	Открывает существующий отчет из файла с расширением FRF. Клавиатурный аналог - Ctrl+O.
	Сохранить отчет	Сохраняет отчет в файл с расширением FRF. Клавиатурный аналог - Ctrl+S.
	Предвари-тельный просмотр	Выполняет построение отчета и его предварительный просмотр. Клавиатурный аналог - Ctrl+P.
	Вырезать	Вырезает выделенные объекты в буфер обмена. Клавиатурный аналог - Ctrl+X.
	Копировать	Копирует выделенные объекты в буфер обмена. Клавиатурный аналог - Ctrl+C.
	Вставить	Вставляет объекты из буфера обмена. Клавиатурный аналог - Ctrl+V.
	Отменить	Отменяет последнюю операцию. Число уровней отката - до 100. Клавиатурный аналог - Ctrl+Z.
	Повторить	Повторяет последнюю отмененную операцию. Клавиатурный аналог - Ctrl+Y.
	На передний план	Помещает выделенные объекты на передний план.
	На задний план	Помещает выделенные объекты на задний план.
	Выделить все	Выделяет все объекты на странице. Клавиатурный аналог - Ctrl+A.
	Новая страница	Создает новую страницу.
	Удалить страницу	Удаляет текущую страницу.

	Свойства страницы	Вызывает диалог со свойствами страницы.
	Показать сетку	Показывает точечную сетку на странице. Шаг сетки можно задать в опциях дизайнера. Клавиатурный аналог - Ctrl+G.
	Выравнивать объекты по сетке	При перемещении или изменении размеров объектов координаты/размеры будут изменяться скачкообразно в соответствии с шагом сетки.
	Расположить в узлах сетки	Изменяет размеры/расположение выделенных объектов таким образом, чтобы они размещались в узлах сетки.
	Справка	Показывает справку по элементам управления.
Выход	Выход	Закрывает окно дизайнера.

Панель инструментов "Текст"



Иконка	Название	Описание
Arial	Шрифт	Позволяет выбрать название шрифта из выпадающего списка. Помнит пять последних использованных шрифтов. Двойной щелчок мышью вызывает стандартный диалог выбора шрифта.
10	Размер шрифта	Позволяет выбрать размер шрифта из выпадающего списка. Размер можно также ввести вручную.
Ж	Утолщение	Устанавливает/снимает утолщение шрифта. Клавиатурный аналог - Ctrl+B.
К	Наклон	Устанавливает/снимает наклон шрифта. Клавиатурный аналог - Ctrl+I.
Ч	Подчеркивание	Устанавливает/снимает подчеркивание шрифта. Клавиатурный аналог - Ctrl+U.
	Цвет шрифта	Выбирает цвет шрифта из выпадающего списка.
	Условное выделение	Показывает диалог с атрибутами выделения для выбранного объекта "Текст".
	Выравнивание влево	Устанавливает выравнивание текста влево.

	Выравнивание по центру	Устанавливает выравнивание текста по центру.
	Выравнивание вправо	Устанавливает выравнивание текста вправо.
	Выравнивание по ширине	Устанавливает выравнивание текста равномерно по ширине.
	Выравнивание по верхнему краю	Устанавливает выравнивание текста по верхнему краю.
	Выравнивание по высоте	Устанавливает выравнивание текста по высоте.
	Выравнивание по нижнему краю	Устанавливает выравнивание текста по нижнему краю.
	Вертикальный текст	Переключает номальный/вертикальный шрифт.

Панель инструментов "Прямоугольник"

Иконка	Название	Описание
	Верхняя линия	Включает/выключает верхнюю линию рамки.
	Левая линия	Включает/выключает левую линию рамки.
	Нижняя линия	Включает/выключает нижнюю линию рамки.
	Правая линия	Включает/выключает правую линию рамки.
	Все линии	Включает все линии рамки.
	Нет линий	Выключает все линии рамки.
	Цвет фона	Выбирает цвет фона из выпадающего списка.
	Цвет линии	Выбирает цвет линии из выпадающего списка.
	Стиль линии	Выбирает стиль линии из выпадающего списка.

	Толщина линии	Выбирает толщину линии из выпадающего списка.

Панель инструментов "Выравнивание"

Иконка	Описание
	Выровнять левые края.
	Центрировать по горизонтали.
	Центрировать по горизонтали в окне.
	Расположить равномерно по ширине.
	Выровнять правые края
	Выровнять верхние края.
	Центрировать по вертикали.
	Центрировать по вертикали в окне.
	Расположить равномерно по высоте.
	Выровнять нижние края.

Построение автономных отчетов

Введение

Диалоговые формы

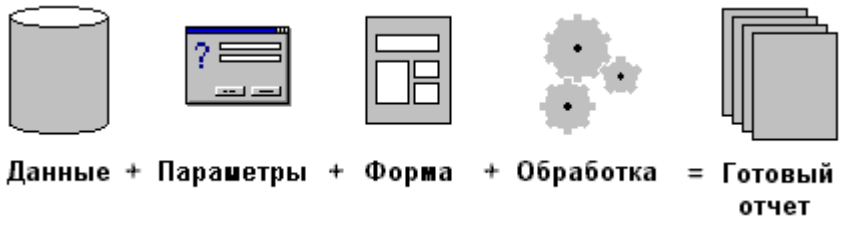
Компоненты доступа к данным

Компонент TfrDataStorage

Языковые средства

Введение

Как уже говорилось в главе "Введение", основные этапы построения отчета можно свести к следующим:



В отчетах, встроенных в проект, все эти этапы легко реализуются средствами Delphi. Так, доступ к данным осуществляется с использованием dbaware-компонент TTable, TQuery и подобным им (как правило, имеется общая на весь проект база данных и доступ возможен только к ее таблицам); для запроса параметров используются диалоговые формы, созданные в процессе разработки проекта; наконец, этап обработки осуществляется с использованием обработчиков событий (event handlers).

Несмотря на эффективность, такой подход (применяемый в большинстве генераторов отчетов), не является универсальным. Малейшее изменение в любом из вышеописанных этапов приводит к необходимости изменения и перекомпиляции всего проекта. Кроме того, в информационных системах с большим количеством отчетов это приводит к загромождению проекта диалоговыми формами. И наконец, такой подход совершенно непригоден в проектах с большой "текучкой" отчетов (т.е. там, где постоянно появляются новые виды отчетов и модернизируются старые).

Эволюция встроенных генераторов отчетов (таких, как QuickReport, ReportBuilder, FastReport) идет по пути постепенного отрыва от среды проекта. Так, сначала появилась возможность модифицировать отчеты во время выполнения программы (в run-time). Потом конечному пользователю дали возможность самому определять источники данных, на основе которых будет построен отчет. ReportBuilder версии 4.0 и позже, помимо этого, позволяет выполнять обработку данных во время построения отчета с использованием встроенного интерпретатора Pascal-подобного языка. FastReport замыкает логическую цепочку - в его составе есть компоненты для построения диалоговых форм.

Таким образом, среда FastReport позволяет конечному пользователю:

- создавать новые и модифицировать существующие отчеты;
- создавать источники данных, на основе которых будет построен отчет;
- создавать диалоговые формы для запроса параметров отчета у пользователя;
- используя встроенный Pascal-подобный язык, выполнять обработку данных в отчете и управлять логикой работы диалоговых окон.

Как видно, это позволяет разрабатывать полностью независимые от проекта отчеты. Компонентная идеология, используемая при создании диалогов и источников данных, основана на применяемой в Delphi.

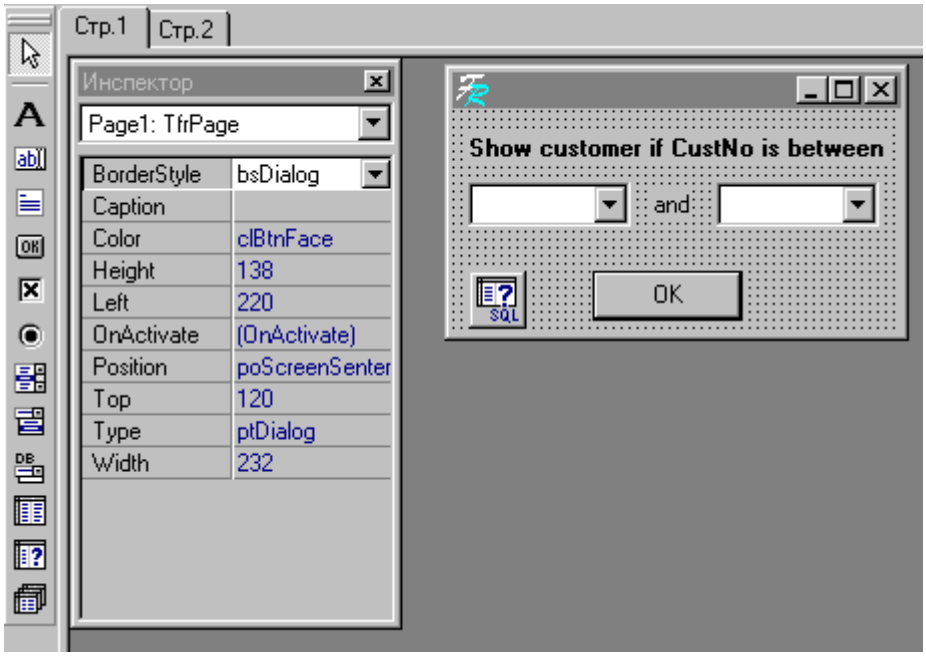
Диалоговые формы

Для разработки диалоговых форм используется тот же дизайнер, что и для страниц отчета. Для создания новой формы служит кнопка на панели инструментов дизайнера - она добавляет в отчет новую страницу, при переключении на которую меняется содержимое панели "Объекты":

Рис. Панель инструментов "Объекты" при активной странице "Форма отчета".

Рис. Панель инструментов "Объекты" при активной странице "Диалоговая форма".

При переключении на страницу с формой диалога рабочее поле дизайнера изменяется - теперь это форма, на которой можно размещать объекты - элементы управления:



Не правда ли, напоминает среду Delphi? В отчете может быть несколько диалоговых форм, они будут выполняться по порядку перед построением отчета.

Элементы управления

Элементы управления диалоговых форм подключаются при использовании в проекте компонента TfrDialogControls из палитры компонентов FastReport. Для этого достаточно положить компонент на любую форму в Вашем проекте или добавить в список `uses` `FR_DCtrl`. Это приводит к подключению следующих элементов управления:

Иконка	Название	Описание
	Label	Строка текста.
	Edit	Предназначен для ввода строки текста с клавиатуры.
	Memo	Предназначен для ввода нескольких строк текста.
	Button	Кнопка.
	CheckBox	Флажок.
	RadioButton	Круглая кнопка - аналог переключателя с зависимой фиксацией.
	ListBox	Список строк предназначен для выбора одного из значений.
	ComboBox	Выпадающий список строк.

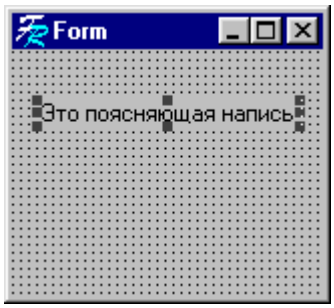
Надо отметить, что все элементы управления являются потомками класса TfrStdControl и имеют общий набор свойств и методов:

Свойство	Значение по умолчанию	Описание
Color	clBtnFace	Цвет фона объекта.
Enabled	True	Разрешает или запрещает работу объекта.
Font	-	Установки шрифта, которым будет выведен текст объекта.
Height	-	Высота объекта.
Left	-	Координата X левого верхнего угла объекта.
Name	-	Имя объекта.
OnClick	-	Реакция на действие над объектом - аналог Delphi OnClick. Здесь может быть код на FastReport Pascal.
Restrictions	-	Набор флагов, запрещающих те или иные действия над объектом (перемещение, удаление, редактирование и пр.).
Top	-	Координата Y левого верхнего угла объекта.
Visible	True	Видимость объекта.
Width	-	Ширина объекта.

Рассмотрим каждый элемент управления.

Label

Назначение этого элемента управления - вывод поясняющей надписи на диалоговой форме:

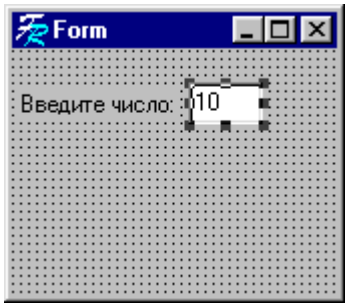


Элемент имеет следующие свойства (помимо вышеперечисленных общих свойств):

Свойство	Значение по умолчанию	Описание
Alignment	taLeftJustify	Выравнивание строки текста внутри рамок объекта.
AutoSize	True	Установка ширины объекта равной максимальной ширине имеющегося в нем текста.
Caption	-	Собственно строка текста.
WordWrap	False	Перенос слов внутри объекта. При этом свойство AutoSize должно быть False.

Edit

Этот элемент управления предназначен для ввода строки текста с клавиатуры.

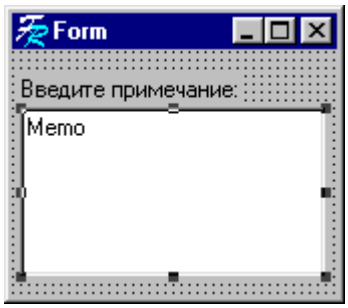


Элемент имеет следующие свойства:

Свойство	Значение по умолчанию	Описание
ReadOnly	False	Запрещает модифицировать строку текста.
Text	-	Текст, который будет показан в элементе управления по умолчанию.

Memo

Этот элемент управления предназначен для ввода нескольких строк текста с клавиатуры.

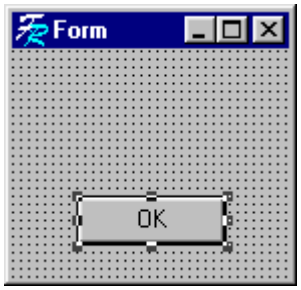


Элемент имеет следующие свойства:

Свойство	Значение по умолчанию	Описание
Lines	-	Строки текста, который будет выведен по умолчанию.
ReadOnly	False	Запрещает модифицировать текст.

Button

Этот элемент управления представляет собой кнопку.



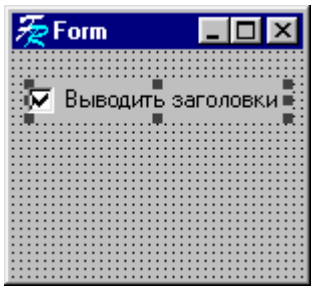
Элемент имеет следующие свойства:

Свойство	Значение по умолчанию	Описание
Caption	-	Заголовок кнопки.
ModalResult	mrNone	Определяет, надо ли завершать работу диалогового окна при нажатии на кнопку и какой результат при этом возвращать.

Свойство ModalResult предназначено для автоматического закрытия окна диалога при нажатии на кнопку. Обычно кнопка "OK" имеет ModalResult = mrOk, кнопка "Отмена" - mrCancel. Кроме того, аналогично свойство имеется у самой формы диалога. Если присвоить ему что-либо, отличное от mrNone, диалог завершит работу.

CheckBox

Этот элемент управления представляет собой флажок, который может быть в двух состояниях: включенном и выключенном. Около флажка выводится поясняющая надпись.

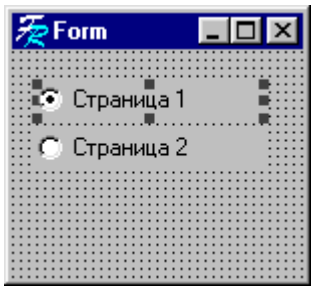


Элемент имеет следующие свойства:

Свойство	Значение по умолчанию	Описание
Alignment	taRighJustify	Позиция надписи по отношению к флажку.
Caption	-	Поясняющая надпись.
Checked	False	Состояние флажка (включен/выключен).

RadioButton

Этот элемент управления представляет собой аналог переключателя с зависимой фиксацией. По этой причине в одиночку не применяется.

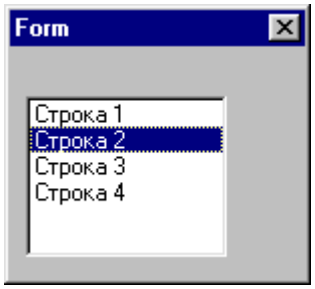


Элемент имеет следующие свойства:

Свойство	Значение по умолчанию	Описание
Alignment	taRighJustify	Позиция надписи по отношению к переключателю.
Caption	-	Поясняющая надпись.
Checked	False	Состояние переключателя (включен/выключен).

ListBox

Этот элемент управления представляет собой список строк с возможностью выбора одной из них.

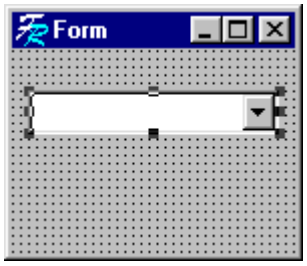


Элемент имеет следующие свойства:

Свойство	Значение по умолчанию	Описание
Items	-	Строки текста.

ComboBox

Этот элемент управления представляет собой выпадающий список строк с возможностью выбора одной из них.



Элемент имеет следующие свойства:

Свойство	Значение по умолчанию	Описание
Items	-	Строки текста.
Style	csDropDown	Тип элемента.
Text	-	Выбранное значение.

Свойство элемента Style определяет тип элемента. Возможны 3 значения:

- csDropDown - элемент позволяет не только выбирать значение из списка, но и набирать нужное в строке редактирования;
- csDropDownList - элемент позволяет вводить только значения из списка;
- csLookup - строка в списке представлена в виде Строка1;Строка2. При этом в списке показывается только часть строки Строка1, а при выборе этого значения в программу передается часть Строка2. Например, таким образом легко организовать выбор месяца, заполнив элемент строками типа Январь;1. Выбранное значение доступно через свойство Text.

Диалоговая форма

Сама диалоговая форма также имеет набор свойств. Для того, чтобы увидеть свойства формы в инспекторе объектов, нужно щелкнуть мышкой на пустом месте формы, не занятом элементами управления.

Свойство	Значение по умолчанию	Описание
BorderStyle	bsDialog	Тип диалоговой формы - с фиксированными размерами либо с возможностью менять размеры окна с помощью мыши.
Caption	-	Заголовок окна.
Color	clBtnFace	Цвет окна.
Height	-	Высота окна.
Left	-	Позиция X окна.
OnActivate	-	Обработчик события, вызывающийся после инициализации всех элементов окна.
Position	poScreenCenter	Позиция окна - по центру экрана либо по заранее заданным координатам Left, Top.
Top	-	Позиция Y окна.
Type	ptDialog	Тип страницы - диалоговая форма либо страница отчета.
Width	-	Ширина окна.

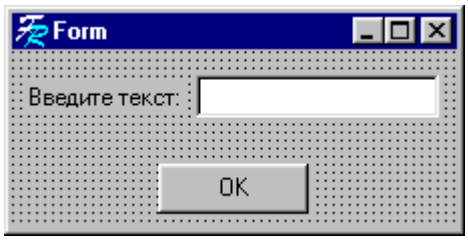
Обработчик OnActivate соответствует Delphi OnActivate. Он вызывается, когда все элементы управления уже размещены на форме диалога и форма готова к выводу на экран. Этот обработчик удобно использовать для занесения каких-либо начальных значений в элементы управления, например, заполнения списка ListBox и пр.

Передача информации в отчет

Значения, введенные в элементы управления, часто требуется тем или иным образом передать в отчет. Существует два способа это сделать.

С использованием переменных

Для передачи информации от элемента управления в отчет используются переменные. Например, необходимо текст из элемента управления Edit отобразить в заголовке отчета.



Это можно сделать следующим образом: в обработчике события OnClick кнопки OK написать:

```
begin
  TitleText := Edit1.Text;
end
```

В залоговок отчета при этом надо поместить объект "Текст" с содержимым [TitleText]:

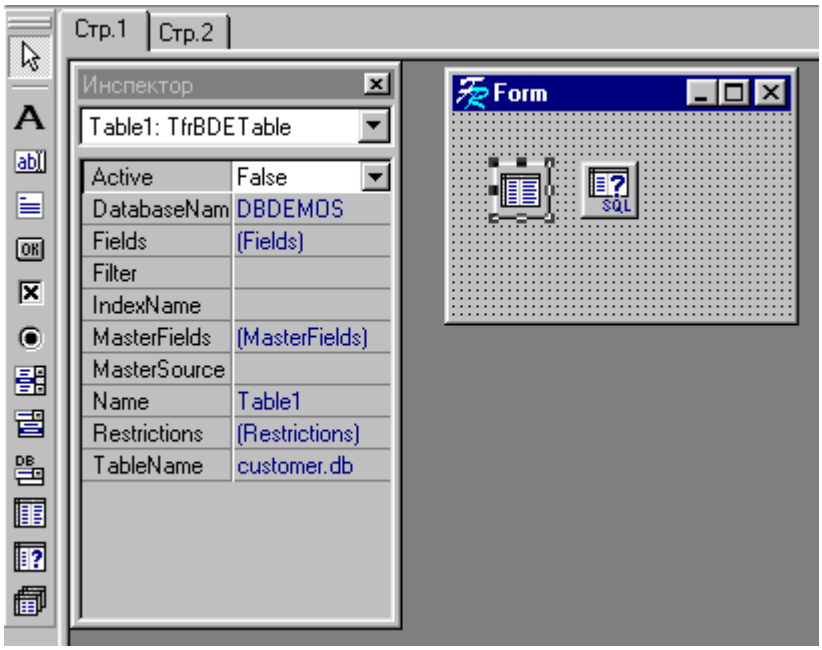
Прямое обращение

Во многих случаях проще использовать для передачи введенного значения в отчет прямое обращение к элементу управления. При этом промежуточных переменных заводить не требуется:

Компоненты доступа к данным

Большинство отчетов, как правило, основано на данных из БД. Для доступа к таким данным Delphi предоставляет эффективные механизмы, которые и используются в FastReport. Речь идет о компонентах TTable и TQuery, которые могут выступать в качестве источников данных для отчета. Вообще, можно использовать с этой целью любые компоненты - наследники TDataSet.

Кроме доступа к данным, определенным в проекте, FastReport позволяет создавать новые компоненты в run-time. В FastReport принципы создания компонентов доступа к данным максимально приближены к тем, что используются в среде Delphi. Так же, как и в Delphi, на форму кладется компонент и в инспекторе объектов настраиваются его свойства. Компонентная идеология очень гибкая: можно легко создавать новые компоненты для поддержки разных движков доступа к данным.



Описание компонентов

Рассмотрим использование компонентов для доступа к данным с помощью BDE. Они подключаются при использовании компонента TfrBDEComponents из палитры FastReport в проекте. При этом в панели объектов дизайнера появляются следующие объекты: TfrBDELookupComboBox, TfrBDETable, TfrBDEQuery, TfrBDEDataBase.

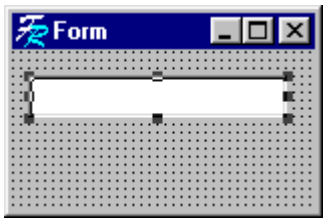
Эти компоненты по своему назначению аналогичны соответствующим компонентам Delphi (TDBLookupComboBox, TTable, TQuery, TDataBase).

Иконка	Название	Описание
	TfrBDELookupComboBox	Элемент управления, предназначенный для выбора значения из справочника.
	TfrBDETable	Предназначен для доступа к таблице БД.
	TfrBDEQuery	Предназначен для выполнения SQL-запроса.
	TfrBDEDataBase	Предназначен для соединения с БД.

Рассмотрим каждый компонент.

TfrBDELookupComboBox

Этот элемент управления предназначен для выбора значения из таблицы-справочника. При этом вместо выбираемого значения подставляется его идентификатор в справочнике.



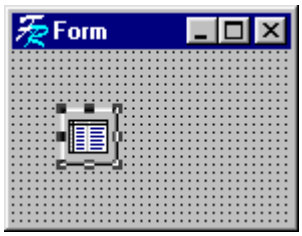
Элемент имеет следующие свойства:

Свойство	Значение по умолчанию	Описание
KeyField	-	Поле-индентификатор выбираемого значения.
ListField	-	Поле, значения из которого будут выбираться в списке.
ListSource	-	Источник данных - таблица-справочник.
Text	-	Выбранное значение.

Для подключения элемента управления к справочнику необходимо заполнить значения трех свойств: KeyField, ListField и ListSource. Выбранное значение доступно через свойство Text, с помощью него же можно установить начальную позицию указателя в списке.

TfrBDETable

Компонент предназначен для организации доступа к таблице БД.



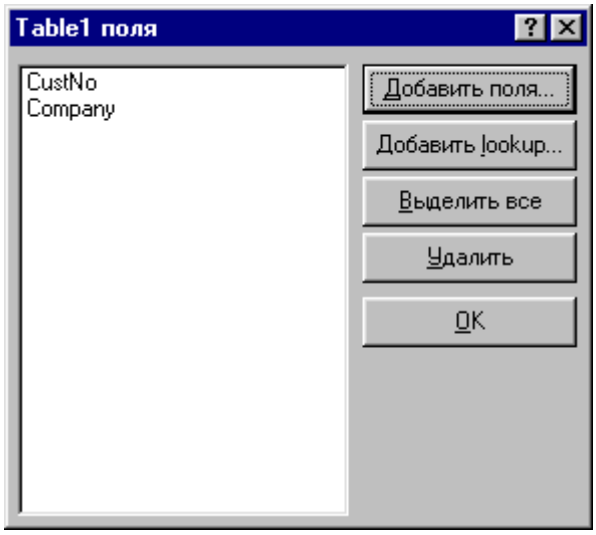
Компонент имеет следующие свойства:

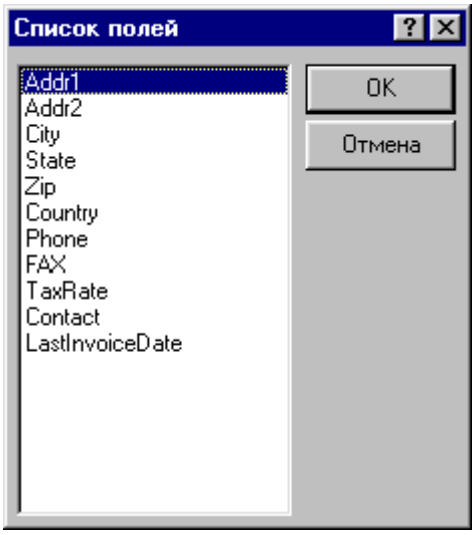
Свойство	Значение по умолчанию	Описание
Active	False	Определяет, активна ли таблица.
DatabaseName	-	Имя псевдонима (alias) БД.
Fields	-	Список доступных полей.
Filter	-	Выражение для фильтрации записей.
IndexName	-	Имя вторичного индекса.
MasterFields	-	Поля, связанные с master-набором данных.
MasterSource	-	Мастер-набор данных.
TableName	-	Имя таблицы БД.

Назначения свойств компонента аналогичны свойствам Delphi TTable. Для подключения компонента к таблице БД достаточно заполнить свойства DatabaseName и TableName. Открытие таблицы осуществляется с помощью установки Active := True.

Свойства Fields и MasterFields настраиваются с помощью редакторов. Для этого нужно нажать на кнопку ... в инспекторе объектов.

Редактор свойства Fields позволяет выбрать поля, которые будут доступны при обращении к таблице.





При первом вызове редактора список полей пуст, т.е. доступны все поля. Для того, чтобы заполнить список нужными полями, нужно нажать кнопку "Добавить поля...". При этом вызывается диалоговое окно со списком полей, которые могут выбраны. После того, как нужные поля выбраны, следует нажать кнопку OK.

Кроме этого, редактор позволяет создавать lookup-поля. Для чего нужны lookup-поля, поясним на примере.

Допустим, имеется две таблицы БД: таблица "Заказы" с полями *N*, *Date*, *ClientID*, *Amount* и таблица "Клиенты" с полями *ID*, *Name*, *Address*. В таблице "Заказы" содержится информация о заказах (номер, дата, идентификатор клиента, сделавшего заказ, и сумма заказа). В таблице "Клиенты" содержится информация о клиентах (идентификатор клиента, полное имя, адрес).

Чтобы сформировать простейший отчет по таблице "Заказы" вида Номер заказа – Дата – Имя клиента – Сумма, потребуется связать обе таблицы по полям ClientID -> ID. Эта задача решается путем создания lookup-поля, которое добавляется к таблице "Заказы" и представляет собой ссылку на поле *Name* таблицы "Клиенты".

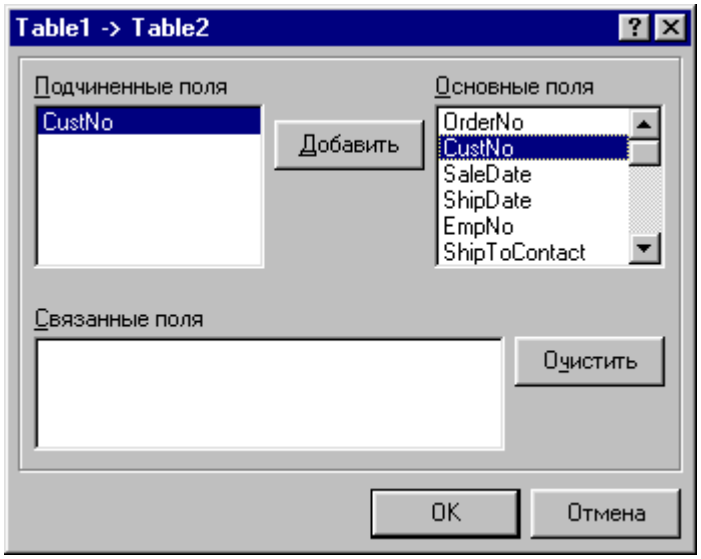
Диалог создания lookup-поля доступен из редактора полей. Диалог аналогичен тому, что используется в среде Delphi.

Для создания lookup-поля надо указать его имя и тип, а также размер (в случае, если выбран строковый тип). Далее необходимо заполнить следующие поля:

- Ключевое поле – поле в исходном наборе данных, которое следует рассматривать как ссылку на поле из lookup-набора. В нашем примере это будет поле *ClientID*.
- Источник данных – lookup-набор данных.
- Lookup ключ – поле в lookup-наборе, являющееся ключевым. В нашем примере это поле *ID*.
- Результирующее поле – поле lookup-набора, которое надо подставить в исходный набор данных. В нашем примере это поле *Name*.

После этого в таблице "Заказы" создается фиктивное поле с заданным именем, которое содержит расшифровку поля *ClientID*. К нему можно обращаться как к обычному полю, но только для чтения.

Редактор свойства MasterFields используется для создания master-detail связей между двумя таблицами. Для связывания двух таблиц отношением master-detail у подчиненной таблицы надо указать в свойстве MasterSource основную таблицу и вызвать редактор свойства MasterFields для подчиненной таблицы. Диалог аналогичен тому, что используется в среде Delphi.



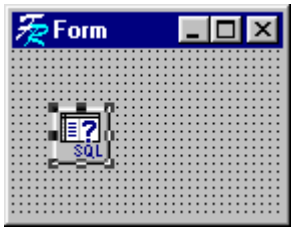
Здесь можно визуально связать поля master и detail наборов данных. Когда наборы связаны друг с другом отношением Master-Detail, то при перемещении по master набору

содержимое detail набора фильтруется таким образом, чтобы в нем содержались только записи, имеющие отношение к текущей записи master набора.

Для связи полей наборов выделите поле из списка слева (detail набор), затем поле из списка справа (master набор), и нажмите кнопку "Добавить". При этом связка полей переместится в нижний список. Чтобы очистить нижний список, воспользуйтесь кнопкой "Очистить". Связываемые поля должны иметь одинаковый тип и быть ключевыми.

TfrBDEQuery

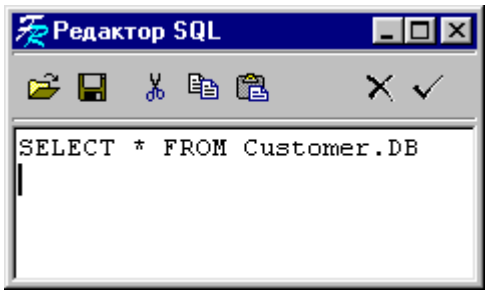
Компонент предназначен для выполнения SQL-запросов к БД.



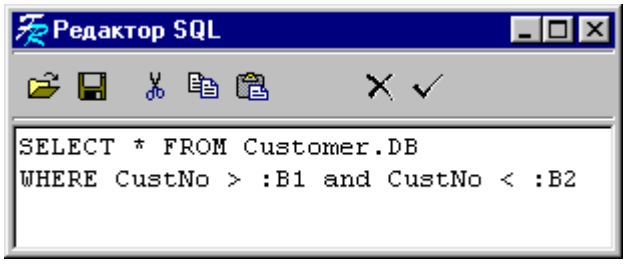
Компонент имеет следующие свойства:

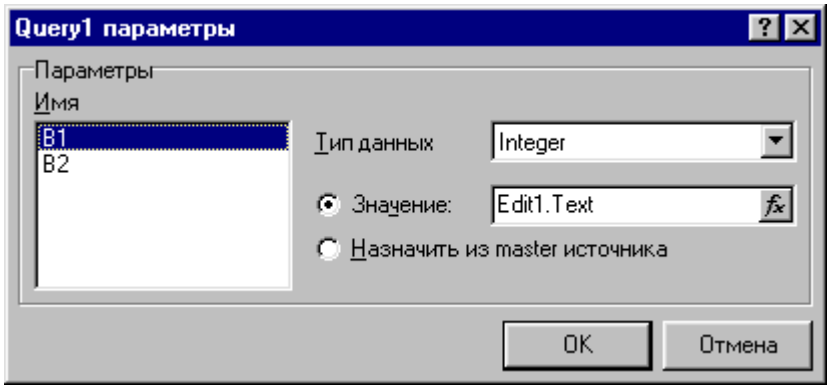
Свойство	Значение по умолчанию	Описание
Active	False	Определяет, активен ли запрос.
DatabaseName	-	Имя псевдонима (alias) БД.
DataSource	-	Мастер-набор данных.
Fields	-	Список доступных полей.
Filter	-	Выражение для фильтрации записей.
Params	-	Список параметров запроса.
SQL	-	Текст запроса.

Свойства Active, DatabaseName, Fields и Filter аналогичны вышеописанным свойствам компонента TfrBDETable. Свойство SQL имеет свой редактор для заполнения SQL-запроса.



Свойство Params также имеет свой редактор. Оно доступно, если текст запроса содержит параметры.

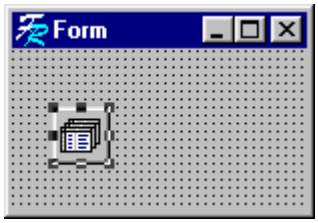




Как видно, редактор параметров аналогичен используемому в Delphi. Параметр может быть двух типов: назначаемый из master-источника либо имеющий конкретное значение (причем в качестве значения может выступать как константа, так и ссылка на переменную или свойство объекта, как показано на рисунке). В случае, когда параметр берется из master-набора данных, набор данных должен содержать поле с именем, совпадающим с именем параметра. Кроме того, это поле должно быть в списке доступных полей. При этом указывать тип параметра необязательно.

TfrBDEDataBase

Этот компонент служит для подключения к базе данных. Его назначение аналогично компоненту Delphi TDataBase.

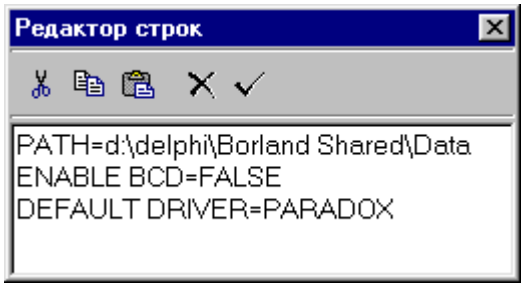


Компонент имеет следующие свойства:

Свойство	Значение по умолчанию	Описание
AliasName	-	Псевдоним, на основе настроек которого будет выполнено подключение к БД.
Connected	False	Если True, активизирует подключение.
DatabaseName	-	Имя, которое будет добавлено в список псевдонимов.
DriverName	-	Имя драйвера, обеспечивающего подключение к БД.
LoginPrompt	True	Определяет, надо ли запрашивать у пользователя пароль при подключении к БД.
Params	-	Параметры подключения.

Компонент осуществляет подключение к базе данных (обычно его используют для подключения к серверной СУБД). Настройки для подключения берутся либо из соответствующего алиаса (свойство AliasName), либо прописываются вручную (для этого необходимо указать имя драйвера DriverName). Компонент должен иметь заполненное свойство DatabaseName - это значение попадет в список псевдонимов.

Для установки параметров подключения нужно вызвать редактор свойства Params. При этом в редакторе будут показаны установки по умолчанию для выбранного подключения. При желании эти установки можно откорректировать и сохранить.



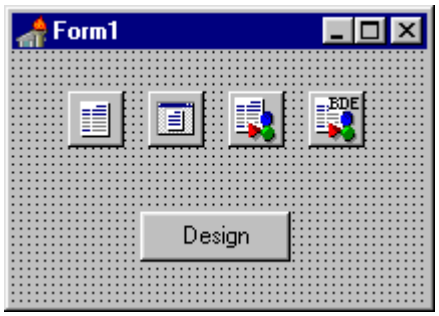
Свойство LoginPrompt определяет, надо ли спрашивать у пользователя пароль при подключении к БД. Если LoginPromt = False, то имя пользователя и пароль должны быть указаны в параметрах подключения, например:

```
SERVER NAME=Путь_к_файлу_*.gdb
USER NAME=SYSDBA
PASSWORD=masterkey
```

Построение отчетов

Рассмотрим построение простого отчета, содержащего компоненты доступа к данным. В качестве данных будем использовать демонстрационные таблицы из поставки Delphi - DBDEMOS.

Для начала создадим проект, с помощью которого будем проводить эксперименты. Для этого создайте новый проект в Delphi и разместите на форме компоненты TfrReport, TfrDesigner, TfrDialogControls, TfrBDEComponents.



Для кнопки "Design" определите следующий обработчик:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  frReport1.DesignReport;
end;
```

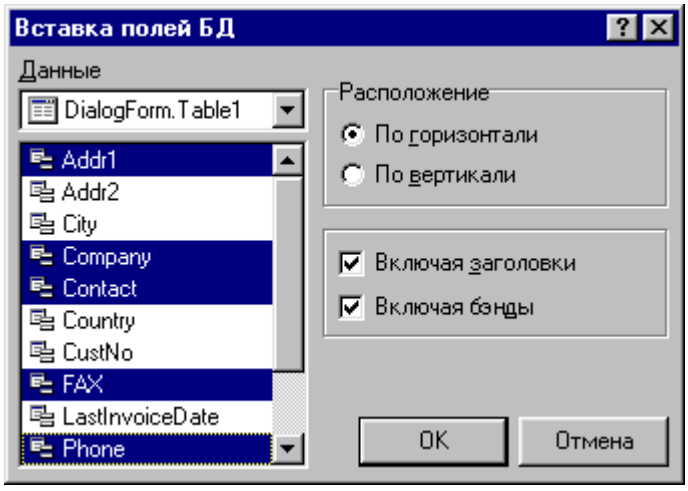
После этого скомпилируйте и запустите проект. Это все, что требуется для создания end-user дизайнера отчетов.

При нажатии на кнопку Design открывается дизайнер, содержащий пустой отчет. Рассмотрим построение простых отчетов в этой среде.

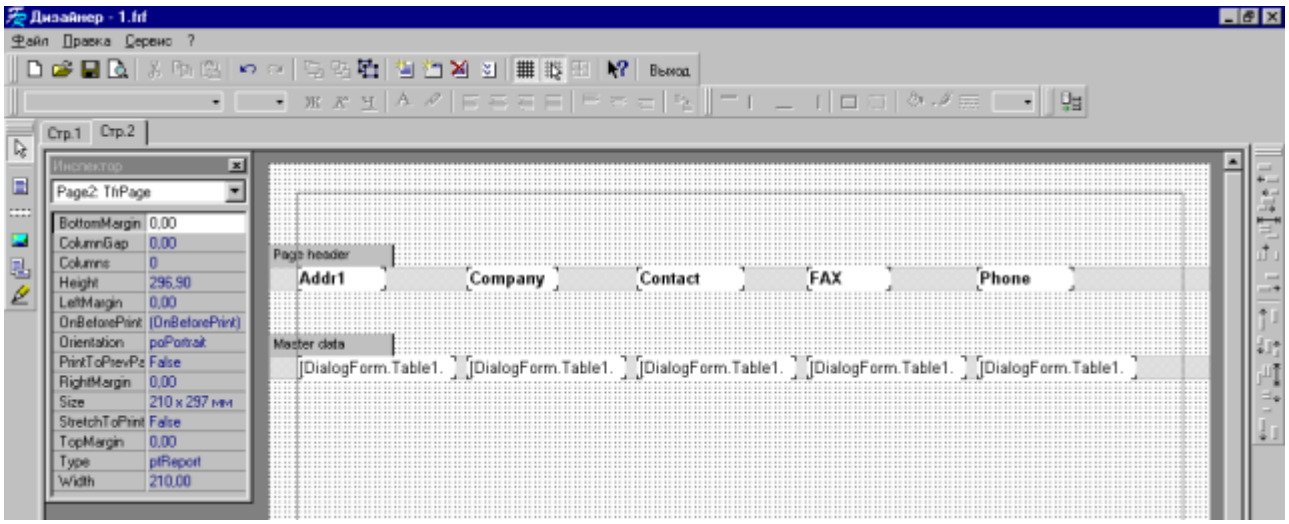
Простой отчет типа "Список"

Этот отчет будет содержать данные из одной таблицы БД. Для построения отчета проделайте следующие шаги:

- Добавьте в отчет диалоговую форму. Эта форма будет использована для размещения компонента доступа к таблице БД.
- На форму положите компонент TfrBDETable. Настройте его свойства: DatabaseName = 'DBDEMOS', TableName = 'Customer.db'.
- Перейдите на страницу с формой отчета. Щелкните в дизайнере мышью по кнопке “Вставка полей БД в отчет” на панели инструментов "Мастера". В открывшемся диалоговом окне выберите поля, которые нужно включить в отчет и нажмите кнопку "OK".



После этого форма отчета будет выглядеть примерно так:

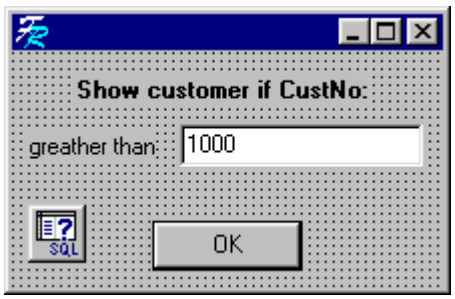


Для просмотра полученного отчета надо нажать на кнопку "Предварительный просмотр" на панели инструментов.

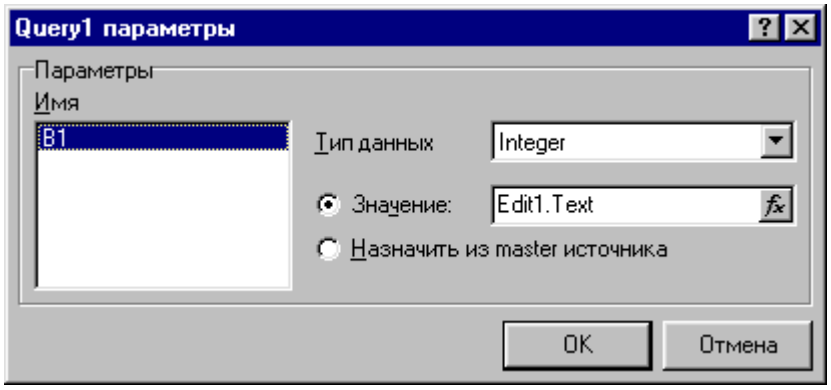
Отчет с запросом параметров

Рассмотрим построение более сложного отчета, где перед построением отчета у пользователя запрашиваются параметры в диалоговом окне. Для этого проделайте следующие действия:

- Добавьте в отчет диалоговую форму.
- Положите на форму отчета компоненты Query, Label, Edit, Button:

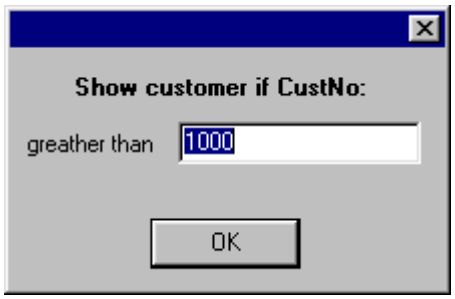


- Настройте свойства компонента Query: DatabaseName = 'DBDEMOS', SQL = 'Select * from Customer.db where CustNo > :B1'.
- Откройте редактор свойства Params компонента Query и настройте данные параметра:



- Перейдите на страницу с формой отчета. Щелкните в дизайнере мышью по кнопке “Вставка полей БД в отчет” на панели инструментов "Мастера". В открывшемся диалоговом окне выберите поля, которые нужно включить в отчет и нажмите кнопку "ОК".

При построении отчета на экран будет выведен диалог, в котором предлагается ввести номер покупателя. После ввода нужного значения и нажатия кнопки ОК построение отчета будет выполнено. На печать выведутся покупатели с номерами, большими чем введенный.



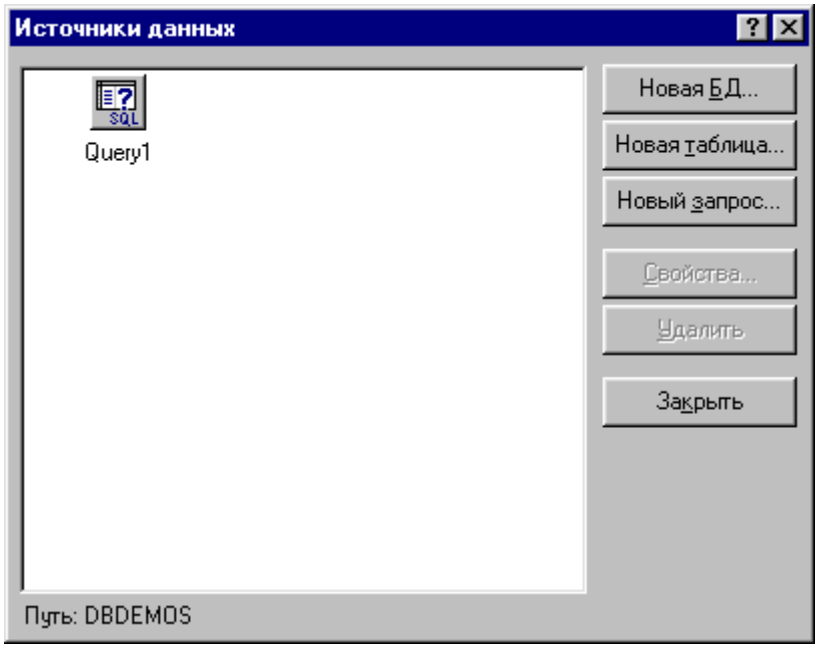
Компонент TfrDataStorage

Компонент TfrDataStorage ("хранилище данных") предназначен для создания в run-time баз данных, таблиц и запросов, настройки списка доступных полей, создания lookup-полей, связей типа Master-Detail. Одним словом, компонент позволяет выполнять ту же работу, что и в среде Delphi. Вся информация сохраняется вместе с формой отчета.

В настоящее время компонент устарел – автор рекомендует использовать связку TfrDialogControls + TfrBDEComponents, описанную в предыдущем разделе.

При подключении компонента в списке мастеров дизайнера появляются два мастера: "Менеджер данных" и "Диалог параметров".

Первый позволяет работать с данными - определять базы данных, таблицы и запросы, редактировать их. Второй мастер предназначен для редактирования формы диалога ввода значений параметров запроса.



Менеджер данных может работать со следующими движками БД: BDE, Interbase Express (IBX), ActiveX Data Objects (ADO). Для активизации нужного движка снимите комментарий с нужной строки в файле FR.INC.

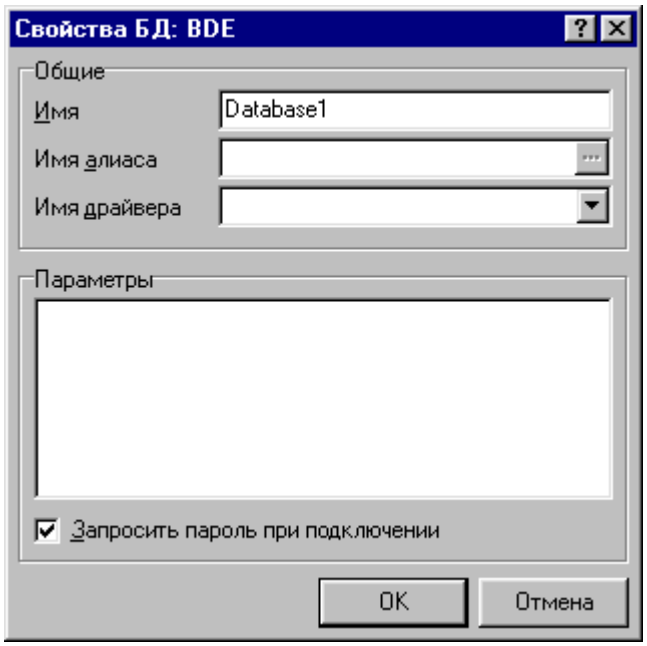
Все компоненты БД, созданные с помощью этого мастера, находятся в модуле данных (datamodule) "ReportData", который доступен в run-time. Для каждой таблицы или запроса создается источник данных для FastReport с таким же именем, как и у таблицы, но со знаком подчеркивания впереди.

Создание базы данных

Создание компонента БД необходимо в случае, если вы работаете с компонентами IBX или ADO. Перед созданием таблиц или запросов необходимо определить как минимум один объект БД.

Для создания новой БД запустите мастер "Менеджер данных". В появившемся окне нажмите кнопку "Новая база данных". В появившемся диалоге "Свойства БД" необходимо задать следующие параметры:

- если вы работаете с ADO или IBX, необходимо указать имя файла, содержащего базу данных;
- если вы работаете с BDE, необходимо указать название алиаса, которое добавится к списку существующих алиасов, и имя драйвера БД. Для локальных таблиц Paradox, DBase следует использовать драйвер "STANDARD".



В поле редактирования параметров удобно указать имя пользователя и пароль, чтобы подключение к базе данных происходило в автоматическом режиме. Например, для подключения к SQL-серверу Interbase через драйвер BDE INTERBASE, используйте следующие параметры:

```
SERVER NAME=Путь к файлу *.gdb
USER NAME=SYSDBA
PASSWORD=masterkey
```

Для подключения с помощью IBX параметры несколько иные:

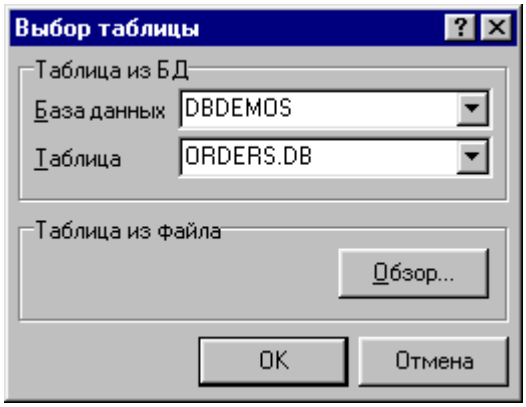
```
user_name=SYSDBA
password=masterkey
```

Флажок "Запросить пароль при подключении" выводит диалоговое окно для ввода имени пользователя и пароля при подключении к SQL серверу. Если вы уже задали эти значения в параметрах БД, флажок можно отключить.

Создание таблицы

Для создания новой таблицы БД запустите мастер "Менеджер данных". В появившемся окне нажмите кнопку "Новая таблица". В появившемся диалоге "Выбор таблицы" необходимо

выбрать таблицу из любой доступной базы данных или, в случае использования локальных таблиц Paradox или DBase, выбрать таблицу прямо из файла, воспользовавшись кнопкой "Обзор".



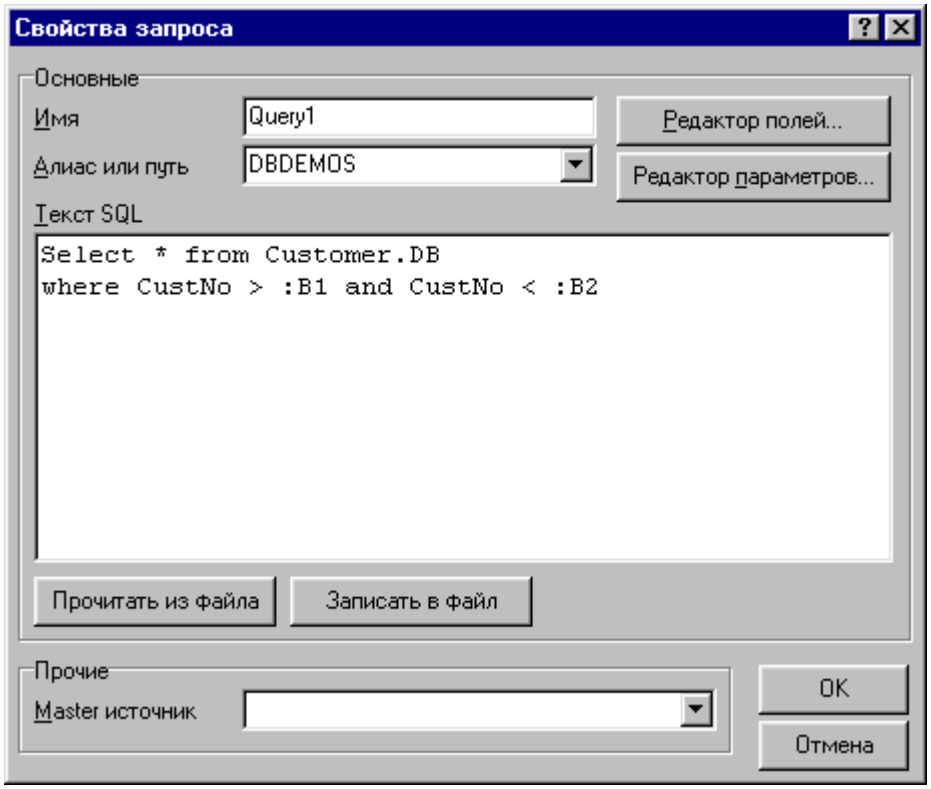
После того, как файл с таблицей выбран, откроется диалоговое окно "Свойства таблицы". Здесь можно задать имя таблицы (по умолчанию предлагается "Table" + первый свободный номер). Вместе с таблицей создаются источники данных TDataSource и TfrDBDataSet. Источнику данных TfrDBDataSet присваивается имя как у таблицы, но со знаком подчеркивания впереди.

Также в этом окне можно вызвать редактор полей, выбрать индекс для таблицы (если у таблицы есть вторичные индексы), задать выражение для фильтрации записей таблицы.

Для указания Master-набора данных для таблицы необходимо выбрать в списке "Master" нужный набор данных, а в строке "Master поля" указать поля из Master-набора (либо, нажав кнопку в правой части строки, вызвать диалог для визуального связывания полей).

Создание запроса

Для создания нового запроса запустите мастер "Менеджер данных". В появившемся окне намите кнопку "Новый запрос". После этого откроется диалоговое окно "Свойства запроса". Здесь можно задать имя запроса (по умолчанию предлагается "Query" + первый свободный номер). Вместе с запросом создаются источники данных TDataSource и TfrDBDataSet. Источнику данных TfrDBDataSet присваивается имя как у запроса, но со знаком подчеркивания впереди.



Выпадающий список "Алиас" содержит местонахождение таблиц БД, на основе которых построен запрос. Здесь может быть значение из выпадающего списка либо путь к каталогу с таблицами. Это поле можно оставить пустым, в этом случае алиас или путь надо указывать в тексте запроса.

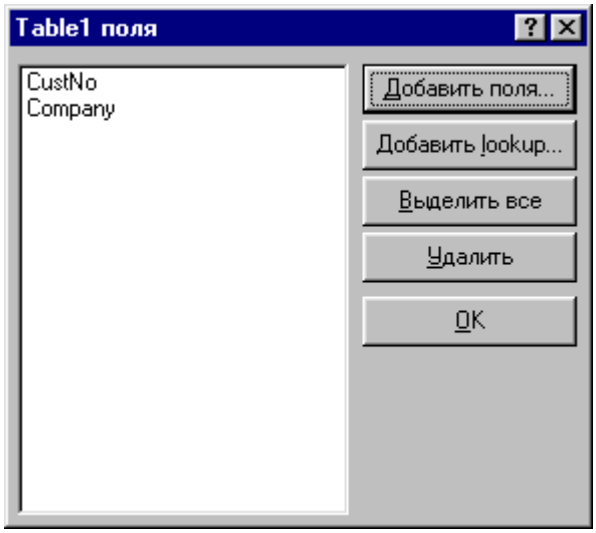
После этого необходимо набрать текст запроса в поле "Текст SQL". Если в тексте запроса имеются параметры (т.е. идентификаторы со знаком ":" в начале имени), то каждый параметр необходимо описать в редакторе параметров.

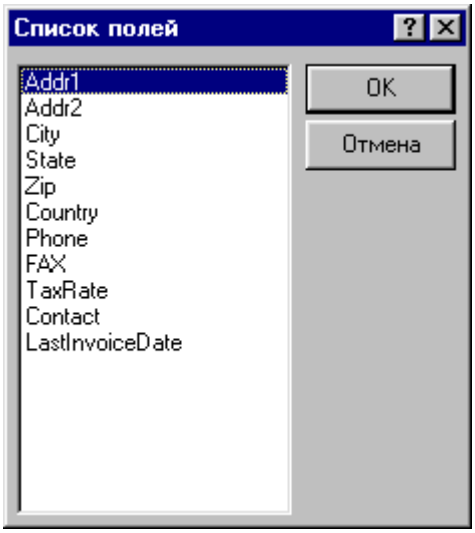
Также в этом окне можно вызвать редактор полей и указать Master-набор данных. Для указания Master-набора данных для запроса необходимо выбрать в списке "Master" нужный набор данных. При этом в тексте запроса должны содержаться параметры - имена полей из Master-источника. Кроме того, каждый параметр должен иметь опцию "Назначить из Master-источника".

При нажатии на кнопку OK проверяется правильность запроса. Если в тексте запроса или в описании параметров есть ошибка, то будет выдано соответствующее предупреждение и редактирование свойств запроса продолжится.

Редактор полей

В этом редакторе можно заполнить список доступных полей таблицы или запроса. Если список пуст, то доступными будут все поля.





Для добавления полей в список нажмите кнопку "Добавить поля". В появившемся окне выберите добавляемые поля и нажмите кнопку OK. Для выбора группы полей щелкните мышью на первом поле группы, затем при нажатой клавише Shift - на последнем.

Чтобы добавить lookup-поле, воспользуйтесь кнопкой "Добавить lookup". См. создание lookup-поля.

Для удаления выделенных полей из списка предназначена кнопка "Удалить".

Создание lookup-поля

Для чего нужны lookup-поля, поясним на примере. Допустим, имеется две таблицы БД: таблица "Заказы" с полями N, Date, ClientID, Amount и таблица "Клиенты" с полями ID, Name, Address. В таблице "Заказы" содержится информация о заказах (номер, дата, идентификатор клиента, сделавшего заказ, и сумма заказа). В таблице "Клиенты" содержится информация о клиентах (идентификатор клиента, полное наименование, адрес).

Чтобы сформировать простейший отчет по таблице "Заказы" вида Номер заказа - Дата - Наименование клиента - Сумма, потребуется связать обе таблицы по полям ClientID -> ID. Эта задача решается путем создания lookup-поля, которое добавляется к таблице "Заказы" и представляет собой ссылку на поле Name таблицы "Клиенты".

Диалог создания lookup-поля доступен из редактора полей.

Для создания lookup-поля надо указать его имя и тип, а также размер (в случае, если

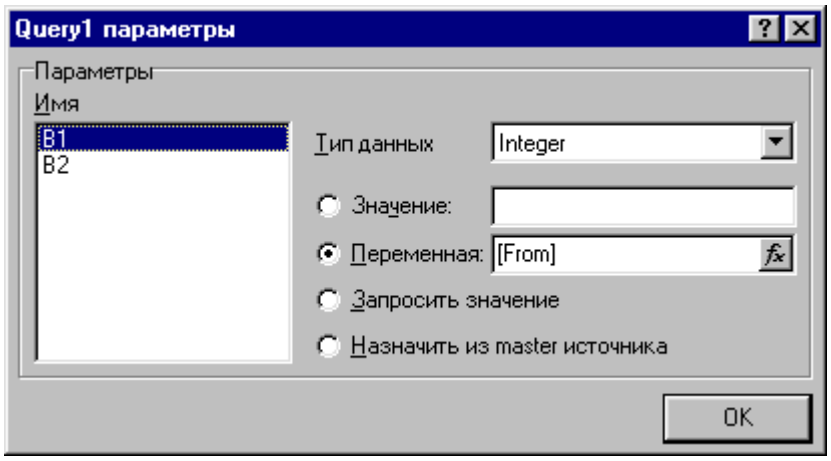
выбран строковый тип). Далее необходимо заполнить следующие поля:

- Ключевое поле - поле в исходном наборе данных, которое следует рассматривать как ссылку на поле из lookup-набора. В нашем примере это будет поле ClientID.
- Источник данных - lookup-набор данных.
- Lookup ключ - поле в lookup-наборе, являющееся ключевым. В нашем примере это поле ID.
- Результирующее поле - поле lookup-набора, которое надо подставить в исходный набор данных. В нашем примере это поле Name.

После этого в таблице "Заказы" создается фиктивное поле с заданным именем, которое содержит расшифровку поля ClientID. К нему можно обращаться как к обычному полю, но только для чтения.

Редактор параметров запроса

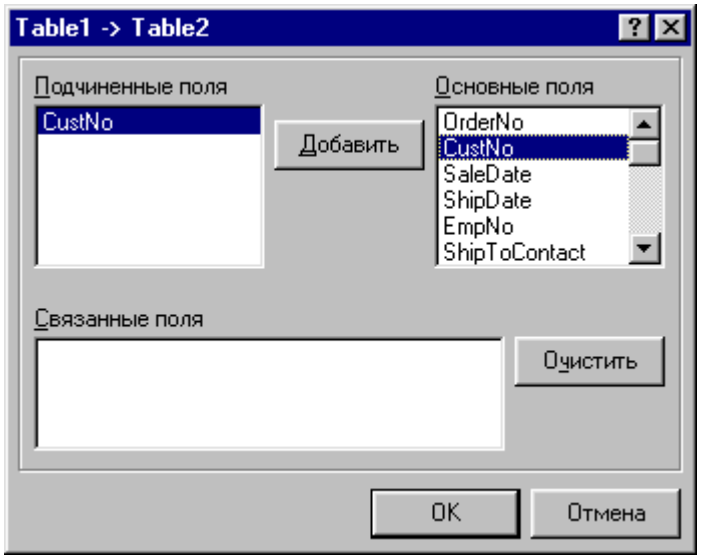
В диалоге можно назначить тип каждому параметру, а также указать, откуда брать значение параметра: из master-набора данных, из диалога ввода значений параметров, либо сразу назначить конкретное значение.



В случае, когда параметр берется из master-набора данных, набор данных должен содержать поле с именем, совпадающим с именем параметра. Кроме того, это поле должно быть в списке доступных полей (см. редактор полей). При этом указывать тип параметра необязательно.

Связывание данных

В этом диалоге можно визуально связать поля master и detail наборов данных. Когда наборы связаны друг с другом отношением Master-Detail, то при перемещении по master набору содержимое detail набора фильтруется таким образом, чтобы в нем содержались только записи, имеющие отношение к master набору.



Для связи полей наборов выделите поле из списка слева (detail набор), затем поле из списка справа (master набор), и нажмите кнопку "Добавить". При этом связка полей переместится в нижний список. Чтобы очистить нижний список, воспользуйтесь кнопкой "Очистить".

Связываемые поля должны иметь одинаковый тип и быть ключевыми.

Диалог ввода значений параметров

Если отчет содержит в качестве источника данных один или несколько запросов с параметрами, которые имеют опцию "Запросить значение" (см. редактор параметров запроса), то перед формированием отчета на экран будет выведен диалог ввода значений параметров. В диалоге будут собраны все параметры, которые встречаются во всех запросах, необходимых для построения отчета.

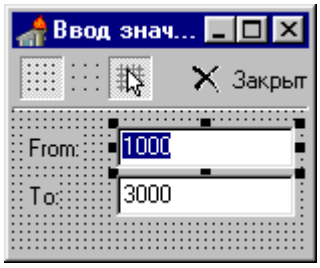
По умолчанию для ввода значения параметра используется управляющий элемент "строка ввода", который имеет поясняющую надпись - имя параметра. Если параметров несколько, то управляющие элементы выводятся друг под другом. Это неудобно - ведь в качестве имени параметра обычно используется сокращенное название на английском языке. В этом случае можно воспользоваться дизайнером диалога.

Диалог будет доступен, если отчет действительно в этом нуждается. Пустой отчет или отчет, не содержащий ссылок на запросы с параметрами, не выводит диалог.

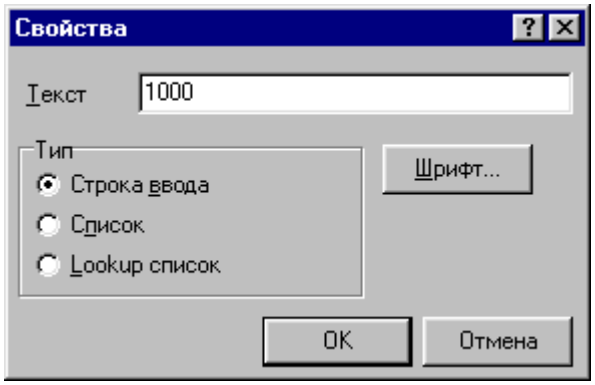
Дизайнер диалога

Дизайнер предоставляет удобные средства для изменения внешнего вида диалога ввода значений параметров. Он позволяет расположить управляющие объекты диалога в нужном порядке, поменять поясняющие надписи, в качестве которых по умолчанию используются имена параметров. Кроме того, дизайнер позволяет сменить тип управляющего элемента для ввода значения параметра - по умолчанию предлагается управляющий элемент "строка ввода".

Дизайнер не позволяет добавлять новые объекты или удалять существующие. Можно только менять их позицию и размеры, задавать свойства (например, параметры шрифта). Все операции выполняются мышью. Мышь работает точно так же, как и в дизайнере формы отчета.

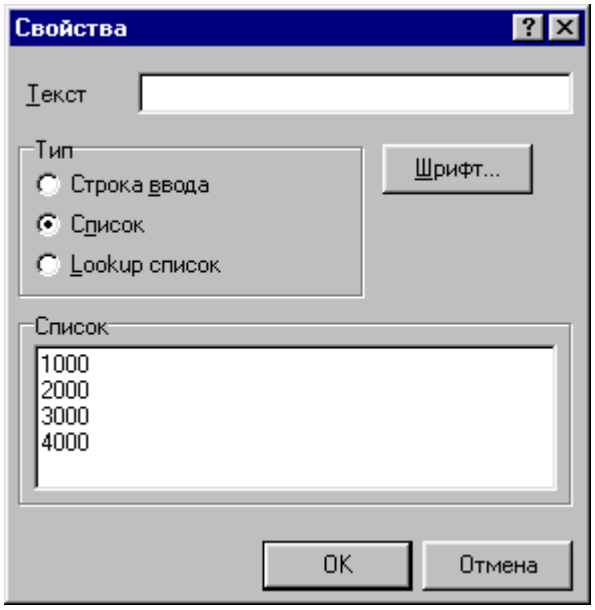


Чтобы задать свойства объекта, сделайте двойной щелчок мышью на нем. В появившемся диалоговом окне можно задать текст, который будет выведен в объекте и параметры шрифта. Диалог также можно вызвать, если выбрано несколько однотипных объектов.



Если выбранный объект - управляющий элемент, то в диалоге свойств станет доступен переключатель "Тип объекта". Можно выбрать три типа:

- Строка ввода - представляет собой строку, в которой можно набрать любой текст. Если в поле "Текст" диалога что-то набрано, то это значение будет отображаться в строке ввода при открытии диалога.
- Список - представляет собой выпадающий список со строками. Полезная вещь, если нужно предоставить пользователю выбор из нескольких значений. Если строка списка содержит символ ";", то в списке будет выведена часть строки до точки с запятой, а при выборе этого значения на самом деле в параметр подставится часть строки после точки с запятой. Таким способом удобно, например, выбирать месяц: при использовании списка со строками типа "январь;1", "февраль;2" в списке будут отображаться названия месяцев, а в параметр попадут номера.



- Lookup список - удобно использовать для выбора значения из справочника (таблицы БД с ключевым полем и полем, содержащим название). Для настройки lookup следует выбрать таблицу, ее ключевое поле, которое подставится в параметр, и поле, которое будет отображаться в выпадающем списке.

Можно менять размеры формы - следует только учитывать, что при выполнении диалога снизу к форме добавится кнопка OK.

Языковые средства

В состав FastReport входит интерпретатор Pascal-подобного языка. Данный интерпретатор является мощным средством, позволяющим в независимости от используемого языка программирования (Delphi или C++Builder) писать скрипты, выполняющиеся при построении отчетов.

В качестве языка используется диалект Pascal, со следующими возможностями:

- операторы: присваивания; условные операторы, операторы циклов и безусловного перехода: `if...then...else, while...do, repeat...until, for..to..do, goto;`
- операторные скобки `begin...end`;
- бестиповые переменные, массивы;
- обращение к свойствам и методам объектов FastReport через точечную нотацию.

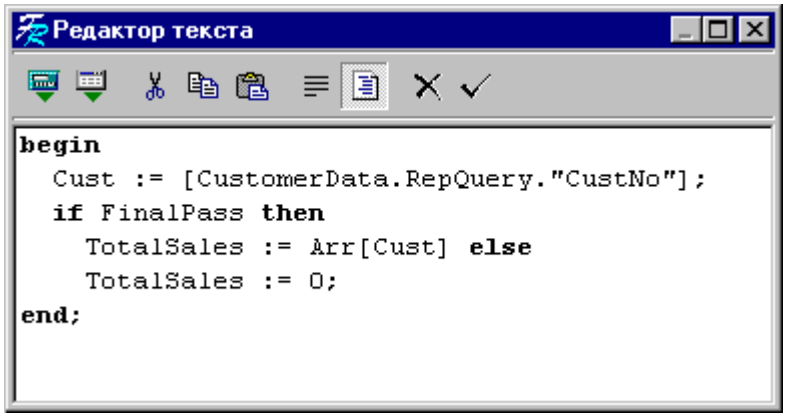
Язык по сравнению с Object Pascal сильно упрощен. Так, используются следующие упрощения:

- все переменные имеют тип Variant; отсутсвует описание переменных;
- все переменные являются глобальными, нет локальных переменных;
- отсутствуют такие типы данных, как "класс", "запись", перечисляемый тип и т.п.;
- нет возможности написания собственных процедур или функций;
- нет операторов прерывания цикла (break, continue);
- количество параметров, передаваемых в процедуру или функцию, может быть не более 3;
- в связи с тем, что переменные не имеют типа, контроль типов данных невозможен - это следует учитывать при написании логических выражений;
- массивы могут быть только одномерными.

Несмотря на значительные упрощения, интерпретатор позволяет выполнять довольно сложную обработку информации. Из скрипта доступны все свойства и методы объектов отчета, а также поля таблиц БД. В скрипте можно создавать переменные и массивы, которые будут доступны во всем отчете. О возможностях встроенного языка говорит тот факт, что такая довольно сложная задача, как печать сумм группы в ее заголовке (сама сумма считается в подвале группы) средствами языка FastReport делается элементарно.

Скрипты и объекты

В каждом объекте может быть размещен один или несколько (блок) операторов. Редактирование скрипта производится в окне редактора текста (для того, чтобы увидеть скрипт, надо нажать кнопку в верхней части окна). Скрипт выполняется каждый раз перед печатью объекта. По сути, скрипт является обработчиком события OnBeforePrint объекта.



Скрипт имеется не только у объектов. И бэнды, и страницы отчета также имеют свой скрипт. Вызвать скрипт бэнда можно, вызвав редактор свойства OnBeforePrint бэнда (из инспектора объектов, либо выделив бэнд и нажав Ctrl+Enter). Вызвать скрипт страницы отчета можно, вызвав редактор свойства OnBeforePrint страницы (для этого надо щелкнуть на пустом месте страницы и вызвать редактор из инспектора объектов). Диалоговая форма, как и страница отчета, имеет скрипт - свойство OnActivate. Скрипт остальных объектов может быть вызван через их свойство Memo либо при нажатии Ctrl+Enter.

Написание кода

В скрипте можно использовать свойства и методы объектов отчета, поля таблиц БД, различные константы. Также можно создавать переменные и массивы, которые будут доступны во всем отчете. Кроме того, можно использовать процедуры и функции.

Использование переменных

Переменную описывать не надо, она имеет тип Variant. В имени переменной можно использовать латинские буквы, цифры и знак подчеркивания. Переменную из скрипта можно использовать в объектах, и наоборот, переменную из списка переменных можно использовать в скрипте. Скриптовые переменные хранятся в объекте **TfrVariables**, к которому есть доступ через глобальную переменную **frVariables**.
Пример использования промежуточной переменной:

```
begin
  Cust := [CustomerData.RepQuery."CustNo"];
  if FinalPass then
    TotalSales := Arr[Cust] else
    TotalSales := 0;
end;
```

В этом примере создается переменная Cust, которой присваивается значение поля таблицы БД.

Кроме этого, можно обращаться к переменным, определенным в словаре данных, к системным переменным, а также пользовательским переменным. При этом переменные могут содержать в имени недопустимые с точки зрения синтаксиса символы (например, системная

переменная Page#). Обращаться к таким переменным следует с использованием квадратных скобок:

```
begin
  a := [Page#];
end
```

Обращение к полям БД

В скрипте могут быть использованы ссылки на поле из таблицы БД. Обращение к полям выполняется в следующем виде:
[Имя формы.Имя таблицы."Имя поля"]

Полная форма обращения применяется в случае, если компонент-таблица и компонент-отчет расположены на разных формах (или модулях данных). Если компоненты расположены на одной форме, обращение может быть вида *[Имя таблицы."Имя поля"]*.

Форма обращения *["Имя поля"]* использует для поиска поля таблицу, которая является основной для бэнда, в котором такое обращение произошло. Например, имея бэнд master data, к которому через источник данных подключена таблица Customer.db, мы можем в пределах всего отчета обращаться к полям таблицы с использованием краткой формы. Использование полной формы обращения не приводит к замедлению работы - в ядре FastReport имеется кэш, который хранит обращения к полям БД.

Массивы

В скрипте, помимо переменных, можно создавать массивы. Массивы могут быть только одномерными, но можно организовать доступ к элементам массива таким образом, что можно трактовать массив как двумерный или с болшим количеством измерений.

Пример использования массива:

```
begin
  MyArr[0] := 'a'; MyArr[1] := 'b'; MyArr[3] := 'd';
  MyArr[2] := MyArr[0] + MyArr[1] + 'c' + MyArr[3];
end;
```

Фактически значения элементов массива хранятся в списке frVariables в виде
Arr_имя массива_индекс. Т.е. в вышеприведенном примере мы получим содержимое frVariables:

```
Arr_MyArr_0 := 'a'
Arr_MyArr_1 := 'b'
Arr_MyArr_2 := 'abcd'
Arr_MyArr_3 := 'd'
```

Константы

В скрипте можно использовать константы. Простейший пример - использование числовых, строковых и логических констант:

```
begin
  a := 0;
  b := 'abcd';
  c := True;
  d := 'That''s all!';
end;
```

Обратите внимание на использование одинарной кавычки внутри строковой константы - как и в Pascal, ее надо дублировать: `d := 'That''s all!'.`

Кроме простых констант, в скрипте доступны константы - названия цветов, стилей шрифта и пр. Вот список констант, которые можно использовать:

- цвета: clWhite, clBlack и т.п. - все стандартные цвета + системные цвета;
- результат работы диалогового окна: mrNone, mrOk, mrCancel;
- служебные: CRLF, Null;
- стиль шрифта: fsBold, fsItalic, fsUnderline;
- рамка объекта: frftNone, frftRight, frftBottom, frftLeft, frftTop;
- выравнивание текста в объекте "Текст": frtaLeft, frtaRight, frtaCenter, frtaVertical, frtaMiddle, frtaDown;
- выравнивание по бэнду: baNone, baLeft, baRight, baCenter, baWidth, baBottom.

Кроме того, определены константы для подключаемых объектов, например, *csCheck* для объекта "CheckBoxObject". Все, что можно видеть в выпадающих списках свойств в инспекторе объектов, можно использовать как константу в скрипте.

Обращение к объектам

В скрипте можно обращаться к свойствам и методам объектов отчета. К объектам отчета относятся визуальные объекты, элементы управления, бэнды, страницы отчета, сам отчет. Для обращения к объекту применятся точечная нотация, например: *Memo1.Text*. Для обращения к собственным свойствам или методам точечная нотация необязательна.

Свойства, к которым возможно обращение, отображены в инспекторе объектов. К некоторым комплексным свойствам (например, Font) возможно обращение как к *Font.Name, Font.Size* и т.п.:

```
begin
  Memo1.Font.Name := 'Courier New';
  Memo1.Font.Size := 10;
  Memo1.Font.Color := clRed;
  Memo1.Font.Style := fsBold + fsItalic
end;
```

К свойствам типа TStrings (Memo, SQL, Items и пр.) возможен доступ по индексу:

```
if Memo1.Lines[1] = 'a' then
  Memo1.Lines[1] := 'b'
```

Также к таким свойствам могут применяться операции Add, Delete, Clear и Count:

```
if Memo1.Lines.Count > 10 then
  Memo1.Lines.Delete(10)
else
begin
  Memo1.Lines.Clear;
  Memo1.Lines.Add('a');
end;
```

Полный список свойств и методов объектов см. в разделе "Свойства и методы объектов". Следует отметить, что обращение к несуществующему методу или свойству не приведет к сообщению об ошибке. Поэтому следует быть осторожным при написании кода.

Использование процедур и функций

В скрипте могут содержаться вызовы процедур и функций. Особенность интерпретатора, а точнее, парсера, который отвечает за обработку процедур/функций, состоит в том, что процедура или функция может иметь не более трех параметров. В скрипте могут использоваться как встроенные процедуры, так и внешние, определенные в проекте. Список встроенных процедур и функций см. в разделе "Описание встроенных процедур и функций".

Примечание. При обращении к процедуре или функции нельзя ставить пробел между именем процедуры и открывающей скобкой.

Модификация объектов

В скрипте можно выполнить любые модификации объекта (либо других объектов) - например, изменение размеров, цвета, содержимого и пр. *Следует учесть, что объекты, которые уже обработаны, модифицированы быть не могут*. То есть, попытка из объекта, лежащего на бэнде report summary, изменить содержимое объекта из report title, ни к чему не приведет. Однако, сделав отчет двухпроходным, мы сможем выполнить такую модификацию.

Аналогичным образом обстоят дела в случае многостраничного отчета. Мы можем обращаться к любому объекту по его имени (имя в пределах отчета уникально). Но модифицированы могут быть только объекты, которые еще не обрабатывались. Если все же требуется модификация уже обработанного объекта, следует сделать отчет двухпроходным.

Описание встроенных процедур и функций

Агрегатные функции

Могут применяться в бэндах ReportSummary, PageFooter, MasterFooter, DetailFooter, SubdetailFooter, GroupFooter, CrossFooter.

- Sum(<expression> [,band] [,1]). Вычисляет сумму выражения expression для ряда строк данных. Если параметр band не задан, то сумма считается по строке данных, соответствующей данному бэнду (по бэндам MasterData, DetailData, SubdetailData); иначе сумма считается только по бэнду с именем band. Если в расчет надо включить невидимые бэнды, надо указывать третий параметр (1). Пример использования:

Sum([Part total], Band1);
Sum([[Part total] + [Part price]]);
Sum([Part total], Band1, 1).

- Avg, Min, Max. Синтаксис аналогичен функции Sum. Функция Avg вычисляет среднее арифметическое, функция Min возвращает минимальное, функция Max - максимальное значение из ряда.
- Count(<band>). Возвращает количество строк данных. Пример использования:
  Count(Band1).

Функции работы со строками

- Str(<value>). Преобразует число в строку.
- Copy(<string>, <from>, <count>). Возвращает подстроку (аналогично одноименной функции в Delphi).
- If(<expression>, <string1>, <string2>). Возвращает строку string1, если выражение expression истинно; иначе возвращает строку string2.
- FormatFloat(<formatstr>, <value>). Преобразует числовое значение value в строку, используя форматирование formatstr. Возможные значения formatstr описаны в документации к Delphi, раздел «Formatting strings».
- FormatDateTime(<formatstr>, <value>). Преобразует дату/время value в строку, используя форматирование formatstr. Возможные значения formatstr описаны в документации к Delphi, раздел «Formatting strings».
- StrToDate(<value>). Преобразует строку в дату.
- StrToTime(<value>). Преобразует строку во время.
- UpperCase(<value>). Преобразует символы строки в верхний регистр.
- LowerCase(<value>). Преобразует символы строки в нижний регистр.
- NameCase(<value>). Преобразует символы строки в нижний регистр, а первый символ - в верхний.
- Length(<string>). Возвращает длину строки.
- Trim(<string>). Отбрасывает пробелы в начале и в конце строки и возвращает результат.
- Pos(<substring>, <string>). Возвращает позицию подстроки <substring> в строке <string>.

Арифметические функции

- Int(<value>). Возвращает целую часть числа value.
- Frac(<value>). Возвращает дробную часть числа.
- Round(<value>). Возвращает округленное значение.
- value1 Mod value2. Возвращает остаток от деления value1 на value2.
- MinNum(<value1>, <value2>). Возвращает меньшее из двух чисел.
- MaxNum(<value1>, <value2>). Возвращает большее из двух чисел.

Прочие функции

- Input(<caption> [,<default>]). Выводит на экран диалоговое окно с заголовком caption и строкой ввода. Если задан параметр default, то это значение появится в строке ввода. После ввода пользователя возвращает введенную строку.
- Date. Возвращает дату начала формирования отчета.
- Time. Возвращает время начала формирования отчета.
- Line#. Возвращает номер строки данных; нумерация начинается с начала группы. Например:

  Master data
  1. Detail data
  2. Detail data
  3. Detail data

  Master data
  1. Detail data
  2. Detail data
- LineThrough#. Возвращает номер строки данных; нумерация строк начинается с начала отчета, например:

  Master data
  1. Detail data
  2. Detail data
  3. Detail data

  Master data
  4. Detail data
  5. Detail data
- Column#. Возвращает номер колонки в cross-tab отчете.
- Page#. Возвращает номер текущей страницы.
- TotalPages. Возвращает общее количество страниц в сформированном отчете. Для использования этой функции отчет должен быть двухпроходным.
- DayOf(<date>). Возвращает день даты.
- MonthOf(<date>). Возвращает месяц даты.
- YearOf(<date>). Возвращает год даты.
- MessageBox(<text>, <caption>, <buttons_and_icons>). Выводит стандартное окно диалога с текстом <text> и заголовком <caption>. Набор кнопок и иконка окна задается в параметре <buttons_and_icons>. Возвращает значение, соответствующее нажатой кнопке (mrOk, mrCancel, mrYes, mrNo). В качестве <buttons_and_icons> можно использовать значения:

Кнопки	**Иконка**
mb_Ok	mb_IconError
mb_OkCancel	mb_IconQuestion
mb_YesNo	mb_IconInformation
mb_YesNoCancel	mb_IconWarning

Процедуры и функции для манипуляции построением отчета

- CurY. Возвращает текущую позицию по Y (в пикселах), с которой будет выведен очередной бэнд. Этому свойству можно также присвоить нужное значение для смены позиции. Это позволяет строить довольно экзотические отчеты.
- FreeSpace. Возвращает высоту оставшегося свободного места на странице в пикселах.

- FinalPass. Возвращает истину, если отчет двухпроходный и выполняется последний проход.
- PageHeight. Возвращает высоту страницы в пикселах с учетом высоты бэнда Page footer.
- PageWidth. Возвращает ширину страницы в пикселах.
- StopReport. Останавливает построение отчета.
- NewPage. Начинает формирование отчета с новой страницы.
- NewColumn. Начинает формирование отчета с новой колонки, если отчет многоколоночный.
- ShowBand(<band>). Показывает бэнд <band>.

Свойства и методы объектов

Все визуальные объекты отчета являются потомками класса TfrView. Он имеет следующие свойства и методы, к которым можно обращаться из скрипта:

Свойство	Тип	Описание
BandAlign	Integer	Выравнивание внутри бэнда. Одна из констант baNone, baLeft, baRight, baCenter, baWidth, baBottom.
Enabled	Boolean	Разрешает или запрещает работу объекта.
FillColor	Integer	Цвет фона объекта. Может быть задан константой clXXX.
FrameColor	Integer	Цвет рамки объекта.
FrameStyle	Integer	Стиль линии рамки. Одна из констант psSolid, psDash, psDot, psDashDot, psDashDotDot, psDouble.
FrameTyp	Integer	Тип рамки объекта - набор констант frftTop, frftBottom, frftLeft, frftRight.
FrameWidth	Double	Толщина рамки.
Height	Integer	Высота объекта.
Left	Integer	Координата X левого верхнего угла объекта.
Memo	String	Текст, находящийся в мемо объекта. К этому свойству можно также обращаться с использованием индекса, например: Memo[1].
Memo.Count	Integer	Возвращает количество строк в мемо.
Name	String	Имя объекта.
Stretched	Boolean	Является ли объект растягиваемым.
Top	Integer	Координата Y левого верхнего угла объекта.
Visible	Boolean	Видимость объекта.
Width	Integer	Ширина объекта.

Методы:

Метод	Параметры	Описание
Hide	-	Аналогично установке Visible := False.
Memo.Add	String	Добавляет строку в мемо.
Memo.Clear	-	Очищает мемо.
Memo.Delete	Integer	Удаляет строку с данным индексом из мемо.
Show	-	Аналогично установке Visible := True.

Стандартные объекты

Объект "Текст" (TfrMemoView)

Помимо общих свойств и методов, которые были рассмотрены выше, объект имеет собственные свойства:

Свойство	Тип	Описание
Alignment	Integer	Выравнивание текста внутри объекта. Набор значений frtaLeft, frtaRight, frtaCenter, frtaVertical, frtaMiddle, frtaDown.
AutoWidth	Boolean	Если True, ширина объекта устанавливается равной ширине имеющегося в объекте текста.
CharSpacing	Integer	Рассстояние между символами.
Font.Name	String	Имя шрифта, которым будет выведен текст объекта.
Font.Size	Integer	Размер шрифта.
Font.Style	Integer	Стиль шрифта. Набор констант fsBold, fsItalic, fsUnderline.
Font.Color	Integer	Цвет шрифта.
GapX	Integer	Промежуток между рамкой объекта и текстом.
GapY	Integer	Промежуток между рамкой объекта и текстом.
HideZeros	Boolean	Если True, не выводить нулевые значения переменных.
LineSpacing	Integer	Расстояние между строками текста.
Suppress	Boolean	Подавлять повторяющиеся значения.
TextOnly	Boolean	Если True, обработка переменных не производится.
WordBreak	Boolean	Переносить слова по слогам (только русские слова).
WordWrap	Boolean	Переносить длинные слова на следующую строку.

Методы: нет.

Объект "Бэнд" (TfrBandView)

Свойство	Тип	Описание
Breaked	Boolean	Бэнд является разрываемым.
ChildBand	String	Дочерний бэнд.
ColumnGap	Integer	Промежуток между колонками.
Columns	Integer	Количество колонок у дата-бэнда.
ColumnWidth	Integer	Ширина колонки.
Condition	String	Условие группировки для бэнда group header.
DataSource	String	Источник данных для дата-бэнда.
EOF	Boolean	True, если достигнут конец набора записей, связанного с бэндом.
FormNewPage	Boolean	Формировать новую страницу после печати бэнда и всех его детальных бэндов.
Master	String	Бэнд, по которому выполняется группировка.
OnFirstPage	Boolean	Печатать на первой странице.
OnLastPage	Boolean	Печатать на последней странице.
PrintChildIfInvisible	Boolean	Печатать дочерние (child) бэнды, если родительский бэнд невидимый.

PrintIfSubsetEmpty	Boolean	Печатать бэнд, если его детальный список пуст.
RepeatHeader	Boolean	Повторять бэнд на новой странице.

Методы:

Метод	Параметры	Описание
First	-	Устанавливает источник данных, связанный с бэндом, на первую запись.
Next	-	Устанавливает источник на следующую запись.
Prior	-	Устанавливает источник на предыдущую запись.

Объект "Рисунок" (TfrPictureView)

Свойство	Тип	Описание
BlobType	Integer	Тип рисунка, содержащегося в BLOb-поле. Одна из констант btBMP, btJPG, btICO, btWMF.
Center	Boolean	Центрировать рисунок внутри объекта.
DataField	String	BLOb-поле таблицы, содержащее рисунок.
KeepAspect	Boolean	Сохранять пропорции рисунка при растягивании.

Методы: нет.

Подключаемые объекты

Объект "Штрихкод" (TfrBarcodeView)

Свойство	Тип	Описание
DataField	String	Поле таблицы, содержащее данные.

Методы: нет.

Объект "CheckBox" (TfrCheckBoxView)

Свойство	Тип	Описание
CheckColor	Integer	Цвет крестика.
CheckStyle	Integer	Стиль (крестик/галочка). Константы csCross, csCheck.
DataField	String	Поле таблицы, содержащее данные.

Методы: нет.

Объекты "RichText", "RichText 2.0" (TfrRichView, TfrRXRichView)

Свойство	Тип	Описание
GapX	Integer	Промежуток между рамкой объекта и текстом.
GapY	Integer	Промежуток между рамкой объекта и текстом.
TextOnly	Boolean	Если True, обработка переменных не производится.
DataField	String	BLOb-поле таблицы, содержащее данные.

Методы: нет.

Объект "Прямоугольник с тенью" (TfrRoundRectView)

Этот объект является потомком объекта "Текст". Следовательно, он имеет тот же набор свойств и методов, что и родитель, и кое-что свое:

Свойство	Тип	Описание
BeginColor	Integer	Начальный цвет градиентной заливки.
EndColor	Integer	Конечный цвет градиентной заливки.
Gradient	Boolean	Использовать градиентную заливку.
RoundRect	Boolean	Использовать закругление.
RoundSize	Integer	Размер закругения.
ShadowColor	Integer	Цвет тени.
ShadowWidth	Integer	Ширина тени.
Style	Integer	Стиль градиентной заливки. Одна из констант gsVertical, gsHorizontal, gsElliptic, gsRectangle, gsHorizCenter, gsVertCenter.

Методы: нет.

Объект "Фигура" (TfrShapeView)

Свойство	Тип	Описание
Shape	Integer	Тип фигуры. Одно из значений skRectangle, skRoundRectangle, skEllipse, skTriangle, skDiagonal1, skDiagonal2.

Методы: нет.

Элементы управления

Все элементы управления являются потомками класса TfrStdControl и имеют общий набор свойств и методов:

Свойство	Тип	Описание
Color	Integer	Цвет фона объекта.
Enabled	Boolean	Разрешает или запрещает работу объекта.
Font.Name	String	Имя шрифта, которым будет выведен текст объекта.
Font.Size	Integer	Размер шрифта.
Font.Style	Integer	Стиль шрифта. Набор констант fsBold, fsItalic, fsUnderline.
Font.Color	Integer	Цвет шрифта.
Height	Integer	Высота объекта.
Left	Integer	Координата X левого верхнего угла объекта.
Name	String	Имя объекта.
Top	Integer	Координата Y левого верхнего угла объекта.
Visible	Boolean	Видимость объекта.
Width	Integer	Ширина объекта.

Методы:

Метод	Параметры	Описание
Hide	-	Аналогично установке Visible := False.
SetFocus	-	Передает фокус ввода элементу управления.
Show	-	Аналогично установке Visible := True.

Объект "Label"

Свойство	Тип	Описание
Alignment	Integer	Выравнивание строки текста внутри рамок объекта. Одно из значений taLeftJustify, taRightJustify, taCenter.
AutoSize	Boolean	Установка ширины объекта равной максимальной ширине имеющегося в нем текста.
Caption	String	Собственно строка текста.
WordWrap	Boolean	Перенос слов внутри объекта. При этом свойство AutoSize должно быть False.

Методы: нет.

Объект "Edit"

Свойство	Тип	Описание
ReadOnly	Boolean	Запрещает модифицировать строку текста.
Text	String	Текст, который будет показан в элементе управления по умолчанию.

Методы: нет.

Объект "Memo"

Свойство	Тип	Описание
Lines	String	Строки текста, который будет выведен по умолчанию. К свойству можно обращаться по индексу: Memo1.Lines[0].
Lines.Count	Integer	Количество строк текста.
ReadOnly	Boolean	Запрещает модифицировать текст.
Text	String	Строки текста, представленные в виде одной строки (с символами CR+LF).

Методы:

Метод	Параметры	Описание
Lines.Add	String	Добавляет строку.
Lines.Clear	-	Очищает строки.
Lines.Delete	Integer	Удаляет строку с данным индексом.

Объект "Button"

Свойство	Тип	Описание
Caption	String	Заголовок кнопки.
ModalResult	Integer	Определяет, надо ли завершать работу диалогового окна при нажатии на кнопку и какой результат при этом возвращать. Одна из констант mrNone, mrOk, mrCancel.

Методы: нет.

Объект "CheckBox"

Свойство	Тип	Описание
Alignment	Integer	Позиция надписи по отношению к флажку. Одна из констант taLeftJustify, taRightJustify.
Caption	String	Поясняющая надпись.
Checked	Boolean	Состояние флажка (включен/выключен).

Методы: нет.

Объект "RadioButton"

Свойство	Тип	Описание

Alignment	Integer	Позиция надписи по отношению к переключателю. Одна из констант taLeftJustify, taRightJustify.
Caption	String	Поясняющая надпись.
Checked	Boolean	Состояние переключателя (включен/выключен).

Методы: нет.

Объект "ListBox"

Свойство	Тип	Описание
Items	String	Строки текста. К свойству можно обращаться по индексу: ListBox1.Items[0].
ItemIndex	Integer	Номер выбранной строки.
Items.Count	Integer	Количество строк.

Методы:

Метод	Параметры	Описание
Items.Add	String	Добавляет строку.
Items.Clear	-	Очищает строки.
Items.Delete	Integer	Удаляет строку с данным индексом.

Объект "ComboBox"

Свойство	Тип	Описание
Items	String	Строки текста. К свойству можно обращаться по индексу: ComboBox1.Items[0].
ItemIndex	Integer	Номер выбранной строки.
Items.Count	Integer	Количество строк.
Style	Integer	Тип элемента. Одно из значений csDropDown, csDropDownList, csLookup.
Text	String	Выбранное значение.

Методы:

Метод	Параметры	Описание
Items.Add	String	Добавляет строку.
Items.Clear	-	Очищает строки.
Items.Delete	Integer	Удаляет строку с данным индексом.

Объект "BDELookupComboBox"

Объект является наследником TfrStdControl и имеет тот же базовый набор свойств и методов плюс свои свойства:

Свойство	Тип	Описание
KeyField	String	Поле-индентификатор выбираемого значения.
ListField	String	Поле, значения из которого будут выбираться в списке.
ListSource	String	Источник данных - таблица-справочник.
Text	String	Выбранное значение.

Методы: нет.

Объект "BDETable"

Свойство	Тип	Описание
Active	Boolean	Определяет, активна ли таблица.
DatabaseName	String	Имя псевдонима (alias) БД.
Fields	Variant	Список доступных полей. К свойству можно обращаться по индексу - имени поля: a := Table1.Fields['Customer'].
FieldCount	Integer	Количество полей в наборе данных.
Filter	String	Выражение для фильтрации записей.
IndexName	String	Имя вторичного индекса.
MasterFields	String	Поля, связанные с master-набором данных.
MasterSource	String	Мастер-набор данных.
TableName	String	Имя таблицы БД.
EOF	Boolean	Истина, если достигнут конец набора записей.
RecordCount	Integer	Количество записей в таблице.

Методы:

Метод	Параметры	Описание
Open	-	Открывает таблицу. Аналогичен установке Active := True.
Close	-	Закрывает таблицу. Аналогичен установке Active := False.
First	-	Устанавливает указатель на первую запись в таблице.
Last	-	Устанавливает указатель на последнюю запись в таблице.
Next	-	Устанавливает указатель на следующую запись в таблице.
Prior	-	Устанавливает указатель на предыдущую запись в таблице.

Объект "BDEQuery"

Свойство	Тип	Описание

Active	Boolean	Определяет, активен ли запрос.
DatabaseName	String	Имя псевдонима (alias) БД.
DataSource	String	Мастер-набор данных.
Fields	Variant	Список доступных полей. К свойству можно обращаться по индексу - имени поля: a := Query1.Fields['Customer'].
FieldCount	Integer	Количество полей в наборе данных.
Filter	String	Выражение для фильтрации записей.
SQL	String	Текст запроса. К свойству можно обращаться по индексу: Query1.SQL[0].
SQL.Count	Integer	Количество строк в тексте запроса.
EOF	Boolean	Истина, если достигнут конец набора записей.
RecordCount	Integer	Количество записей в таблице.

Методы: те же, что и у компонента BDETable и кое-что еще:

Метод	Параметры	Описание
SQL.Add	String	Добавляет строку.
SQL.Clear	-	Очищает строки.
SQL.Delete	Integer	Удаляет строку с данным индексом.

Объект "BDEDataBase"

Свойство	Тип	Описание
AliasName	String	Псевдоним, на основе настроек которого будет выполнено подключение к БД.
Connected	Boolean	Если True, активизирует подключение.
DatabaseName	String	Имя, которое будет добавлено в список псевдонимов.
DriverName	String	Имя драйвера, обеспечивающего подключение к БД.
LoginPrompt	Boolean	Определяет, надо ли запрашивать у пользователя пароль при подключении к БД.
Params	String	Параметры подключения. К свойству можно обращаться по индексу: DataBase1.Params[0].
Params.Count	Integer	Количество строк в параметрах.

Методы:

Метод	Параметры	Описание
Params.Add	String	Добавляет строку.
Params.Clear	-	Очищает строки.
Params.Delete	Integer	Удаляет строку с данным индексом.

Применение

Вот некоторые простые примеры использования интерпретатора:

1. Необходимо вывести сумму заказа: белым фоном, если сумма меньше 2000; зеленым, если сумма от 2000 до 10000; красным, если сумма больше 10000. Для этого в объекте с суммой необходимо набрать следующий скрипт:

```
if [Сумма] < 2000 then
  FillColor := clTransparent
else if [Сумма] < 10000 then
  FillColor := clGreen
else
  FillColor := clRed
```

Цвет можно задать числом, например:

```
FillColor := 128 + 128*256 + 128*65536 //(серый цвет)
```

2. Необходимо вывести те записи, для которых сумма заказа больше 2000. Для этого в бэнде необходимо набрать следующий скрипт (редактор можно вызвать из инспектора объектов):

```
if [Сумма] > 2000 then
  Visible := 1 else
  Visible := 0
```

3. Пользуясь функцией FreeSpace, которая возвращает размер свободного места на листе, и процедурой NewPage, которая начинает печать с новой страницы (или NewColumn – с новой колонки), можно управлять процессом формирования отчета. Например: если места осталось меньше 30мм, начать печать со следующего листа. Для этого в скрипте нужного бэнда необходимо набрать следующее:

```
if FreeSpace * 5/18 < 30 then NewPage
```

4. Для того, чтобы вывести бэнд Report summary внизу страницы, используйте следующий код в скрипте этого бэнда:

```
CurY := PageHeight - Height
```

Программирование

Обработчики событий

Переменные

Расширение функциональных возможностей

Обработчики событий

Часто данные нужно извлекать из других источников, не имеющих отношения к базам данных (например, файл, массив и т.п.). Для этих целей служит компонент TfrUserDataset, генерирующий события OnFirst, OnNext, OnCheckEOF. Кроме того, потребуется написать обработчики событий OnGetValue, OnBeforePrint компонента TfrReport.

Обработчик события OnGetValue вызывается каждый раз, когда в тексте объекта встречается переменная и нужно получить ее значение. Если переменной сопоставлено значение, то она обрабатывается внутренним обработчиком компонента. Иначе нужно подключать внешний обработчик, например:

```
procedure TForm1.Doc1GetValue(const ParName: string; var ParValue: Variant);
begin
  if ParName = 'Var1' then
    ParValue := '1'
  else if ParName = 'Var2' then
    ParValue := 2
end;
```

Обработчик события OnBeforePrint вызывается перед прорисовкой любого объекта. Обычно он используется для того, чтобы загрузить в объект содержимое Memo-поля или картинки из БД. Пример обработчика:

```
procedure TForm1.Doc1BeforePrint(Memo: TStringList; View: TView);
begin
  if Memo.Count > 0 then
    if Memo[0] = '[Memo]' then
      Memo.Assign(Table1Memo)
    else if (Memo[0] = '[Picture]') and (View is TPictureView) then
     (View as TPictureView).Picture.Assign(Table1Picture);
end;
```

Обработчик события OnUserFunction вызывается, когда в тексте объекта обнаружена переменная или выражение, содержащее вызов функции. Функция может иметь до 3-х параметров любого типа. Пример обработчика:

```
procedure TForm1.Doc1UserFunction(const name: string; p1, p2, p3: Variant;
  var val: Variant);
var
  d: Double;
begin
  if name <> 'MONEYTOSTR' then Exit;
  d := frParser.Calc(p1);
  val := MyMoneyToStr(d);
end;
```

Для навигации по источникам данных, не имеющим отношения к БД (например, массивам), используется компонент TfrUserDataset, расположенный на палитре компонентов FastReport. Он генерирует события OnFirst, OnNext, OnCheckEOF, назначение которых, думается, понятно.

Прочие события объекта TfrReport

Кроме описанных в предыдущем разделе событий FastReport содержит еще ряд обработчиков событий:

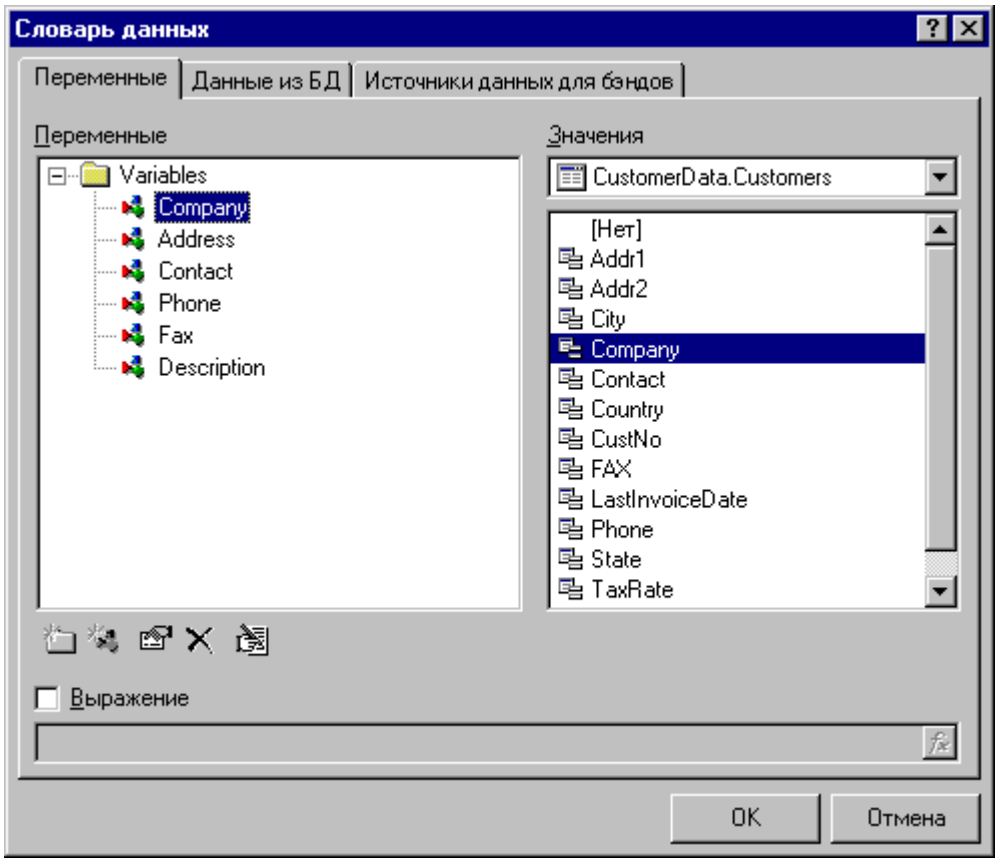
- OnBeginBand – обработчик события вызывается перед началом формирования данных секции. Ссылка на секцию, данные которой должны быть сформированы передается обработчику в качестве параметра.
- OnBeginColumn – обработчик события вызывается при формировании отчета с переменным количеством столбцов. Параметр Band показывает, для какой секции в данный момент строится многоколоночная часть. Обработчик может быть полезен для переключения источника данных (т.к. многоколоночная часть может иметь заголовок, данные первого уровня, второго и пр., для которых используются разные источники данных).
- OnBeginDoc – обработчик события вызывается перед началом формирования отчета.
- OnBeginPage – обработчик события вызывается перед началом формирования страницы.
- OnEndBand – события генерируется сразу после окончания формирования данных секции.
- OnEndDoc – обработчик события вызывается после окончания формирования отчета.
- OnEndPage – обработчик события вызывается после окончания формирования страницы.
- OnManualBuild – обработчик этого события позволяет вручную (т.е. с помощью кода) построить отчет. Если обработчик не назначен, то отчет строится как обычно. При построении отчета вручную можно управлять порядком вывода секций.
- OnPrintColumn – событие вызывается при формировании столбцов cross-tab отчета. Обработчику события передается номер столбца и его ширина по умолчанию. Меняя ширину, можно получить cross-tab отчет с различной шириной столбцов. При этом надо также изменять ширину объекта(ов), находящегося в ячейке таблицы.
- OnProgress – обработчик события вызывается для индикации прогресса выполнения текущей работы (построение отчета, печать, экспорт). На этот обработчик можно повесить свою процедуру отображения прогресса.

Переменные

В объектах и выражениях FastReport можно использовать переменные. Что это такое и для чего это нужно?

Предположим, имеется таблица сотрудников, в которой есть поля N, Name1, Name2, Name3. Вспоминать, что скрывается за такими названиями полей, иногда тяжело даже разработчику базы. Поэтому можно сопоставить полям БД нормально названные переменные (Таб.номер, Фамилия, Имя, Отчество), и в объекты вставлять их. Кроме того, переменной можно сопоставить не только поле из базы, но и математическое выражение, например, сумму значений полей БД, округленное значение, дату, время и пр. Пример использования переменных см. в любом из отчетов демо.

Для работы со списком переменных служит диалоговое окно, вызываемое по команде меню "Файл|Словарь данных...".



Список переменных расположен в левой части окна. Как видно из рисунка, структура списка двухуровневая: он состоит из категорий, в каждой категории может быть одна или несколько переменных. Категории нужны только для визуальной группировки переменных и в отчет не вставляются. Подробнее о приемах работы со списком переменных см. в разделе "Дизайнер".

Однако FastReport умеет работать не только с переменными, описанными в словаре данных. При обнаружении в выражении неизвестного идентификатора FastReport последовательно проверяет, что:

- это переменная из словаря данных;
- это поле БД;
- это одно из специальных значений типа Page#, Date, Time и т.п.;
- это переменная из списка frVariables;
- это свойство объекта;

- это константа из списка frConsts.

Кроме того, если компонент имеет подключенный обработчик события OnGetValue, то он вызывается перед всеми проверками. Если обработчик вернул какое-либо значение, то считается, что переменная инициализирована и дальнейшие проверки не производятся.

Таким образом, это дает большую свободу действий при выборе способа передачи какого-либо значения в отчет. Автор рекомендует:

- при передаче статической информации, которая не меняется от записи к записи, использовать передачу значений через глобальный объект frVariables, например, так:

```
frVariables['Отчетный период'] := 'январь';
frReport1.ShowReport;
```

- при передаче информации, которая меняется от записи к записи, использовать либо словарь данных, либо обработчик события TfrReport.OnGetValue, например, так:

```
procedure TForm1.frReport1GetValue(const ParName: String; var ParValue: Variant);
begin
  if AnsiCompareText(ParName, 'Отчетный период') = 0 then
    ParValue := Table1OtchPeriod.Value;
end;
```

- можно также программно заполнить список переменных:

```
with frReport1.Dictionary do
begin
  Variables['Номер'] := 1;
  Variables['Сумма'] := '0.2 * Table1."Summa"';
  Variables['Дата'] := '''' + 'январь' + '''';
end;
```

Дополнительная пара кавычек в этом примере необходима при присвоении строковых констант. Дело в том, что FastReport воспринимает строковые значения, присвоенные переменным из словаря данных, как подлежащие вычислению выражения. Если переменная не используется в словаре данных, то она работает как обычно.

Следует отметить, что при использовании переменной из списка frVariables или через обработчик OnGetValue, ее не надо помещать в словарь данных.

Расширение функциональных возможностей FastReport

Генератор отчетов FastReport является открытым, то есть его функциональность может быть наращена путем написания собственных библиотек функций, новых визуальных компонент, фильтров экспорта и мастеров. В данном разделе содержится информация о способах разработки всех перечисленных видов расширений.

Создание собственных окон предварительного просмотра

Иногда возникает необходимость заменить стандартное окно предварительного просмотра своим. Это может быть вызвано необходимостью выполнения дополнительных действий перед выводом (например записью в таблицу БД даты печати документа) или программисту может просто не нравится стиль встроенного окна.

В любом случае FastReport предоставляет разработчику возможность построения собственных окон предварительного просмотра. Для этого служит компонент TfrPreview , расположенный на палитре компонентов FastReport.

Данный компонент служит для отображения страницы в том виде, в котором она будет выведена на печать и содержит ряд методов для навигации по сформированному отчету, его печати и правки, сохранения готового отчета в любом из поддерживаемых форматов и загрузки отчета из файла *.FRP, а также изменения масштаба изображения в окне предварительного просмотра.

Ключевыми методами данного компонента являются:

- First, Prev, Next, Last – осуществляют переход к первой, предыдущей, следующей и последней странице отчета соответственно;
- SaveToFile – сохраняет подготовленный отчет в файл любого из поддерживаемых форматов;
- LoadFromFile – загружает ранее сохраненный отчет из файла формата *.FRP;
- Print – выводит подготовленный отчет на принтер;
- OnePage – устанавливает масштаб изображения так, чтобы в окне отображалась одна страница целиком;
- PageWidth – устанавливает масштаб изображения для отображения одной страница по ее ширине.

Кроме перечисленных методов данный компонент имеет свойство Zoom, которое управляет масштабом отображения страницы. Для изменения масштаба этому свойству нужно присвоить требуемое значение в процентах от натуральной величины.

Следует отметить, что помимо изображения страницы компонентом TfrPreview будут выведены полосы прокрутки и строка состояния с информацией о текущем номере страницы и их общем количестве.

Для подключения подготовленного окна предварительного просмотра к генератору отчетов FastReport ссылку на него нужно присвоить свойству Preview соответствующего компонента TfrReport.

Пример использования пользовательского окна предварительного просмотра можно найти в подкаталоге REPORTS каталога примеров FastReport.

Расширение списка функций

Ранее в данном руководстве уже был описан метод, при помощи которого FastReport может быть дополнен новыми функциями. Этот метод основан на использовании события OnUserFunction объекта TfrReport. Тем не менее данный способ обладает одним недостатком: в каждом объекте TfrReport, использующем пользовательскую функцию, нужно описать обработчик события OnUserFunction (т.е. пользовательские функции описанные таким образом являются локальными по отношению к объекту TfrReport). Кроме того является весьма привлекательным решение, заключающееся в оформлении целого ряда функций в виде отдельного класса (библиотеки функций).

FastReport поддерживает альтернативный способ подключения дополнительных функций к генератору отчетов, заключающийся в написании класса-наследника от TfrFunctionLibrary. Данный класс определяет базовый набор свойств и методов, которые должны быть реализованы разработчиком для возможности работы с FastReport.

Базовый класс TfrFunctionLibrary объявлен в модуле FR_Class следующим образом:

```
TfrFunctionLibrary = class(TObject)
public
  List: TStringList;
  constructor Create; virtual;
  destructor Destroy; override;
  function OnFunction(const FName: String; p1, p2, p3: Variant;
    var val: String): Boolean;
  procedure DoFunction(FNo: Integer; p1, p2, p3: Variant; var val: String);
    virtual; abstract;
end;
```

Ключевыми методами здесь являются:

- Create - в конструкторе надо заполнить список поле List списком функций, в алфавитном порядке и в верхнем регистре;
- DoFunction – при вызове метод должен вернуть в параметре Val вычисленное значение функции. Каждая функция может иметь до трех параметров. Значения этих параметров передаются методу DoFunction в качестве параметров P1, P2 и P3.

Ниже приведен простейший пример библиотеки из двух функций:

```
type
  TMyFunctionLibrary = class(TfrFunctionLibrary)
  public
    constructor Create; override;
    procedure DoFunction(FNo: Integer; p1, p2, p3: Variant;
      var val: Variant); override;
  end;

constructor TMyFunctionLibrary.Create;
begin
  inherited Create;
  with List do
  begin
    Add('DATEPROPIS');
    Add('SUMPROPIS');
  end;
end;

procedure TMyFunctionLibrary.DoFunction(FNo: Integer; p1, p2, p3: Variant;
  var val: Variant);
begin
  val := 0;
  case FNo of
    0: val := My_DateConvertion_Routine(frParser.Calc(p1));
    1: val := My_SumConvertion_Routine(frParser.Calc(p1));
  end;
```

```
end;
```

Для подключения готовой библиотеки функций к FastReport ее нужно зарегистрировать, вызвав процедуру frRegisterFunctionLibrary и передав ей в качестве параметра ссылку на класс (не на экземпляр класса!). Например так:

```
frRegisterFunctionLibrary(TMyFunctionLibrary);
```

Для подключения списка дополнительных функций к диалоговому окну «Вставка функции» дополнительно нужно сделать вызов процедуры frAddFunctionDesc, которой в качестве параметров передаются:

- имя функции;
- название категории, к которой относится функция;
- текстовое описание синтаксиса функции и выполняемых действий. Обратите внимание: символ «/» в описании функции обязателен! Он отделяет описание синтаксиса от описания самой функции.

Обратите внимание, что для каждой функции должен быть сделан свой вызов frAddFunctionDesc. Пример регистрации функции в генераторе выражений приведен ниже:

```
frAddFunctionDesc('SUMPROPIS', 'Мои функции',
  'SUMPROPIS(<Число>)/Возвращает сумму прописью.');
```

Разработка собственных фильтров экспорта

В стандартную поставку генератора отчетов FastReport входит ряд фильтров экспорта, позволяющих сохранять подготовленный отчет в форматах TXT, RTF, CSV и HTML. Поддержка всех перечисленных форматов обеспечивается подключаемыми модулями, представленными в виде невизуальных компонентов TfrTextExport, TfrRTFExport, TfrCSVExport и TfrHTMLExport. Программисту предоставляется возможность разработки дополнительных фильтров экспорта для поддержки других форматов.

Для этого служит класс TfrExportFilter, объявленный в модуле FR_Class и являющийся базовым для всех фильтров экспорта генератора отчетов FastReport. Объявление данного класса приведено ниже:

```
TfrExportFilter = class(TObject)
protected
  FileName: String;
  Stream: TStream;
  Lines: TList;
  procedure ClearLines;
public
  constructor Create(AStream: TStream); virtual;
  destructor Destroy; override;
  function ShowModal: Word; virtual;
  procedure OnBeginDoc; virtual;
  procedure OnEndDoc; virtual;
  procedure OnBeginPage; virtual;
  procedure OnEndPage; virtual;
  procedure OnData(x, y: Integer; View: TfrView); virtual;
  procedure OnText(DrawRect: TRect; x, y: Integer; const text: String;
    FrameTyp: Integer; View: TfrView); virtual;
end;
```

Ключевыми методами данного класса, подлежащими переопределению программистом являются:

- OnText – данный метод вызывается FastReport в том случае, если в отчете найден текстовый блок. Данный метод отвечает за накопление данных о расположении текста на странице отчета, начертании и прочих свойствах шрифта. В качестве хранилища для этих данных используется свойство объекта List. В качестве единицы хранения информации для одного текстового объекта может служить класс frTextRec, объявленный в модуле FR_Class;
- OnData – данный метод вызывается автоматически при нахождении нетекстового блока. Таким блоком может быть графическое изображение, диаграмма и пр. Разработчик фильтра экспорта должен обеспечить реализацию данного метода для сохранения графики.

Дополнительно фильтр экспорта может обрабатывать следующие события:

- OnBeginDoc – возникает перед началом формирования отчета. Обработчик данного события может использоваться, например, для записи стандартного заголовка в выходной файл;
- OnEndDoc – обработчик события вызывается после окончания формирования отчета. Обработчик события может записывать информацию, сигнализирующую конец информации (например, тэг </HTML>);
- OnBeginPage – событие генерируется перед началом формирования новой страницы;
- OnEndPage – обработчик данного события вызывается при окончании данных страницы и может использоваться, например, для записи символа конца страницы в текстовом файле. В стандартных фильтрах экспорта обработчик этого события используется для записи информации, накопленной методами OnText/OnData.

Перед началом операции экспорта FastReport вызывает метод ShowModal фильтра, который может использоваться для отображения окна диалога параметров экспорта. Данный

метод должен вернуть mrOK если нужно продолжить операцию экспорта данных. В противном случае FastReport будет полагать, что операция экспорта была отменена.

Вся записываемая фильтром информация должна сохраняться в поток, на который ссылается свойство Stream фильтра экспорта. Ссылка на данный объект передается в качестве параметра конструктору объекта при инициализации фильтра.

Для подключения нового фильтра к FastReport нужно вызвать процедуру frRegisterExportFilter, передав ей в качестве параметров:

- ссылку на экземпляр класса фильтра экспорта;
- текст описания формата файлов;
- маску файлов, соответствующих данному формату.

Например:

```
frRegisterExportFilter(MyExportFilter, 'Экспортированный отчет (*.exp)',
'*.exp');
```

Описания стандартных фильтров экспорта можно найти в файлах FR_E_TXT.PAS, FR_E_RTF.PAS, FR_E_CSV.PAS и FR_E_HTM.PAS.

Написание дополнительных мастеров

В FastReport могут быть встроены дополнительные мастера и инструменты. В отличие от ранее описанных расширений для мастеров FastReport не существует каких-либо стандартов. Обычно при вызове мастера на экран выводится диалоговое окно, при помощи которого пользователь может выполнить ту или иную работу.

Подключение дополнительного инструмента к дизайнеру FastReport осуществляется вызовом процедуры frRegisterTool в качестве параметром которой передаются:

- текст пункта меню, который будет добавлен в меню «Сервис/Мастера» дизайнера;
- ссылка на экземпляр объекта TBitmap, определяющий растровое изображение для кнопки, добавляемой в панель инструментов «Мастера»;
- обработчик события типа TNotifyEvent, который вызывается при нажатии кнопки на панели инструментов или выборе соответствующего пункта меню.

Ниже следует пример вызова функции frRegisterTool:

```
frRegisterTool('Мой новый мастер', MyButtonBmp, MyWizardOnClick);
```

Разработка собственных визуальных компонент

FastReport поддерживает расширения в виде дополнительных компонент, которые могут быть помещены в форму отчета. Как уже указывалось ранее, FastReport содержит ряд дополнительных компонент, таких как «Штрих-код» или «Диаграмма». Программист, использующий FastReport для построения отчетов, может разрабатывать свои собственные компоненты и подключать их к среде дизайнера так, как это делается, скажем, для библиотек функций или фильтров экспорта.

При написании собственных компонент для FastReport, как и в случае с компонентами Delphi, следует придерживаться определенного стандарта. Ниже приводятся основные концепции написания пользовательских компонент для генератора отчетов FastReport.

Класс TfrObject

Базовым объектом для FastReport является класс TfrObject. Все объекты FastReport так или иначе наследуются от него. Данный класс определяет интерфейс объекта с дизайнером, а также инспектором объектов.

Для доступа к «публикуемым» свойствам, которые будут доступны в среде дизайнера из инспектора объектов данный класс реализует следующие методы:

- AddProperty – добавляет опубликованное свойство в список свойств объекта;
- AddEnumProperty – добавляет опубликованное свойство типа "список" в список свойств объекта;
- SetPropValue – устанавливает значение опубликованного свойства;
- GetPropValue – читает значение опубликованного свойства;
- DefineProperties – метод вызывается перед отображением списка свойств в инспекторе объектов, а также при выполнении скрипта;
- ClearPropList – очищает список опубликованных свойств объекта.

Разработчик дополнительных компонент должен перекрывать реализацию этих методов в том случае, если компонент имеет собственные публикуемые свойства.

Добавление нового свойства

Добавление нового свойства осуществляется вызовом методов AddProperty и AddEnumProperty, объявленных в модуле FR_Class.pas следующим образом:

```
procedure AddProperty(PropName: String; PropType: TfrDataTypes;
  PropEditor: TNotifyEvent);
procedure AddEnumProperty(PropName: String; Enum: String;
  const EnumValues: Array of Variant);
```

Здесь параметр PropName определяет имя свойства, которое будет отображаться в инспекторе объектов и использоваться для доступа к значению этого свойства.

Параметр PropType определяет набор атрибутов нового свойства. Допустимые атрибуты определяются типом TfrDataType следующим образом:

- frdtString – свойство является строковым параметром;
- frdtInteger – свойство является целым числом;
- frdtFloat – свойство является числом с плавающей запятой;
- frdtBoolean – логическое значение;
- frdtColor – свойство определяет цветовой атрибут;
- frdtEnum – значение свойства должно быть выбрано из списка;
- frdtHasEditor – данное свойство имеет свой собственный редактор (в этом случае должен быть указан параметр PropEditor);

- frdtSize – свойство определяет позицию или размер. Указание этого атрибута позволяет дизайнеру FastReport автоматически переводить одни единицы измерения в другие (пиксели – миллиметры - дюймы);
- frdtOneObject – при выборе нескольких объектов свойства с типом frdtOneObject будут скрыты (например, таким является свойство Memo).

Последний параметр PropEditor является указателем на обработчик события TNotifyEvent. Значение данного параметра используется лишь в том случае, если в маске атрибутов свойства присутствует frdtHasEditor. В данном случае обработчик этого события вызывается при нажатии на кнопку или при двойном щелчке мыши на значении свойства в инспекторе.

Примеры определения свойств:

```
AddProperty('Name', [frdtString, frdtOneObject], nil);
AddProperty('Left', [frdtSize], nil);
AddProperty('FrameColor', [frdtColor], nil);
AddProperty('Visible', [frdtBoolean], nil);
AddProperty('Memo', [frdtOneObject, frdtHasEditor], frMemoEditor);
```

В случае необходимости определить свойство, которое имеет список возможных значений (например, Alignment имеет возможные значения taLeftJustify, taRightJustify, taCenter), следует пользоваться методом AddEnumProperty. В качестве параметров методу следует передать строку, содержащую список допустимых значений, разделенных точкой с запятой, и массив соответствующих значений. Вместо массива можно передать [Null], в этом случае значения будут соответствовать строкам, переданным в параметре Enum.

```
AddEnumProperty('FrameStyle',
  'psSolid;psDash;psDot;psDashDot;psDashDotDot;psDouble',
  [psSolid,psDash,psDot,psDashDot,psDashDotDot,psDouble]);
AddEnumProperty('AliasName', GetAliasNames, [Null]);
```

Инициализация списка свойств объекта

Для инициализации списка пукликуемых свойств объекта предусмотрен метод DefineProperties. Данный метод автоматически вызывается FastReport перед отображением списка в инспекторе объектов, а также при выполнении скрипта.

Данный метод должен, при необходимости, содержать вызов AddProperty для всех публикуемых свойств объекта:

```
procedure TfrLabelControl.DefineProperties;
begin
  inherited DefineProperties;
  AddEnumProperty('Alignment',
    'taLeftJustify;taRightJustify;taCenter',
    [taLeftJustify,taRightJustify,taCenter]);
  AddProperty('AutoSize', [frdtBoolean], nil);
  AddProperty('Caption', [frdtString], nil);
  AddProperty('WordWrap', [frdtBoolean], nil);
end;
```

Запись и чтение значения опубликованного свойства

Методы SetPropValue и GetPropValue вызываются FastReport соответственно при необходимости присвоения нового значения свойству или определения его текущего значения.

Методу SetPropValue в качестве параметров передается имя свойства, значение которого должно быть изменено и, собственно, само значение.

Метод GetPropValue принимает только один параметр – имя свойства и должен вернуть текущее значение запрошенного свойства.

В модуле FR_Class.pas эти методы объявлены следующим образом:

```
procedure SetPropValue(Index: String; Value: Variant); virtual;
function GetPropValue(Index: String): Variant; virtual;
```

Очистка списка опубликованных свойств

Иногда может возникнуть необходимость очистки списка ранее опубликованных свойств объекта. Обычно в этом нет необходимости, однако для объектов, которые могут работать в нескольких режимах (например, объект TfrPage может быть как страницей отчета, так и окном диалога) может потребоваться повторная инициализация. В этом случае для очистки списка может использоваться вызов метода ClearPropList без параметров.

Класс TfrView

Данный класс является потомком TfrObject и является базовым классом для компонентов, помещаемых в форму отчета. Компоненты, наследованные от данного класса могут не являться печатаемыми компонентами, такими как «Текст», но зато могут являться полигоном для рисования других компонент. От этого класса, например, наследуется объект «Секция».

Данный класс определяет набор базовых свойств и методов, используемых ядром FastReport для манипуляции объектами. Сюда входят информация о родительской секции, координатах и размере компонента на странице дизайнера, имени объекта и базовом имени, методы для масштабирования и отрисовки объекта, также ряд служебных методов и переменных.

Ниже приводится детальное описание всех ключевых методов и свойств.

Свойство Parent

Объявление:

```
Parent: TfrBand;
```

Данное свойство заполняется автоматически ядром FastReport и содержит ссылку на секцию, владеющую данным компонентом. Это свойство может быть использовано только во время построения отчета.

Свойства SaveX, SaveY, SaveDX, SaveDY, SaveGX, SaveGY, SaveFW

Объявления:

```
SaveX, SaveY, SaveDX, SaveDY, SaveGX, SaveGY: Integer;
SaveFW: Single;
```

Перед прорисовкой объекта FastReport записывает в эти свойства значения свойств x, y, dx, dy, gapx, gapy, определяющие положение и размеры объекта, после чего сами координаты масштабируются с учетом текущих значений ScaleX, ScaleY, OffsX, OffsY. Свойство SaveFW служит хранилищем значения FrameWidth (толщина рамки).

Свойство BaseName

Объявление:

```
BaseName: String;
```

Данное свойство определяет базовое имя для объекта, используемое в качестве основы при выборе имени для нового компонента в дизайнере. Метод CreateUniqueName дополняет значение данного свойства справа номером для получения уникального имени объекта.

Свойство Canvas

Объявление:

```
Canvas: TCanvas;
```

Данное свойство содержит ссылку на текущий холст, на котором будет осуществляться прорисовка объекта. Данное свойство автоматически заполняется при вызове метода BeginDraw.

Свойство DRect

Объявление:

```
DRect: TRect;
```

Описывает область прорисовки (т.е. то, что находится внутри рамки объекта) – с учетом линий рамки.

Свойство Memo1
Объявление:
```
Memo1: TStringList;
```

Это служебное свойство с которым осуществляются все текстовые операции, такие как подстановка значений, разбиение на строки и т.д.

Свойства FDataSet и FField
Объявления:
```
FDataSet: TfrTDataSet;
FField: String;
```

Эти свойства используются только при работе с BLOB-объектами. Перед формированием отчета поля инициализируются, если в мемо объекта первой строкой стоит что-то в квадратных скобках, и это что-то является именем поля БД или переменной, которая на него ссылается, то во время формирования отчета в методе TfrReport.OnInternalEnterRect делается проверка: является ли поле блобом, и если да, то вызывается метод TfrView.OnGetBlob.

Свойство StreamMode
Объявление:
```
StreamMode: (smFRF, smFRP);
```

Данное свойство позволяет определить куда сейчас сохраняется отчет: в файл frf или во внутренний поток (он же файл frp). Это поле используется в методах LoadFromStream и SaveToStream. Использование данного свойства позволяет избежать записи ненужной информации в файл frp (например, содержимого скрипта).

Методы ShowBackground и ShowFrame
Вызов данных методов без параметров позволяет выполнить прорисовку заднего фона и рамки объекта соответственно. Разработчик компонента может перекрыть реализацию этих методов для получения специальных эффектов.

Метод BeginDraw
Объявление:
```
procedure BeginDraw(ACanvas: TCanvas);
```

Данный метод следует обязательно вызывать перед началом прорисовки объекта в методе Draw.

Метод GetBlob
Объявление:
```
procedure GetBlob(b: TfrTField); virtual;
```

Данный метод отвечает за загрузку содержимого «большого бинарного объекта» (такого как текст в формате RTF или объекта OLE). Если объект умеет отображать данные из BLOB-полей, этот метод надо перекрывать. В качестве параметра этому методу передается объект TfrTField, который в зависимости от использованных при компиляции пакетов FastReport, является экземпляром класса TIB_Column (если определена переменная компиляции IBO) или TField (в противном случае).

Событие OnHook
Объявление:
```
procedure OnHook(View: TfrView); virtual;
```

Данный обработчик события пока используется только в ChartView. Этот метод вызывается каждый раз после обработки каждого объекта во время формирования отчета. Для того чтобы объект мог ловить события, ему надо поставить Flags := flWantHook.

Событие Loaded
Объявление:
```
procedure Loaded; virtual;
```

Данное событие вызывается автоматически после того, как форма отчета (все объекты) загружена (в настоящее время используется в dbaware-компонентах).

Свойство Name
Объявление:
```
Name: String;
```

Данное свойство определяет стандартное имя компонента Delphi.

Свойство Typ
Объявление:
```
Typ: Byte;
```

Это свойство может принимать одно из значений, определенных константами gtXXX, определяя тем самым тип объекта. Для разрабатываемых компонентов, подключаемых к среде FastReport этот параметр всегда будет равен gtAddIn.

Свойство Selected
Данное логическое свойство принимает значение «Истина» если объект в данный момент выбран (подсвечен в дизайнере) и «Ложь» в противном случае.

Свойства ScaleX, ScaleY, OffsX, OffsY
Данные свойства используются FastReport для масштабирования объектов в окне предварительного просмотра и при печати. Они не должны изменяться программистом напрямую.

Свойства x, y, dx, dy, gapx, gapy
Эти свойства определяют, соответственно, координаты верхнего левого угла объекта на странице отчета, ширину и высоту, а также расстояние области печати текста или иного содержимого от краев объекта. Значения x, y, dx и dy доступны из инспектора объектов как Left, Top, Width и Height соответственно.

Свойство Flags
Объявление:
```
Flags: Word;
```

Данное свойство содержит маску флагов определенных константами flXXX. Допустимые значения определяются типом объекта. Так для объектов «Текст» доступными являются флаги:
- flWordWrap – перенос слов;
- flWordBreak – перенос слов по слогам.

Для объектов типа «Секция» комбинация флагов может включать:
- flBandNewPageAfter – указывает на необходимость формирования новой страницы;
- flBandPrintifSubsetEmpty – разрешает печатать, если подчиненный список пуст;
- flBandPageBreak – указывает, что секция является разрываемой;
- flBandOnFirstPage – разрешает печать секции на первой странице;

- flBandOnLastPage – разрешает печать секции на последней странице;
- flBandRepeatHeader – заставляет FastReport повторять печать секции на всех страницах.

Для объектов «Картинка» специфичными являются флаги:
- flPictCenter – заставляет центрировать изображение в объекте;
- flPictRatio – заставляет FastReport сохранять пропорции при масштабировании.
- Кроме того для объектов любых типов могут быть установлены следующие флаги:
- flStretched – говорит о том, что объект растягиваемый;
- flWantHook – разрешает объекту получает управление после отрисовки любого объекта.

Свойство FrameTyp
Определяет тип рамки для объекта. Рамка задается установкой битовых полей. При этом первый бит устанавливает рамку справа, второй – снизу, третий – слева и четвертый – сверху.

Свойство FrameWidth
Данное свойство задает толщину рамки, определяемой свойством FrameTyp. Толщина рамки может быть указана дробным числом.

Свойство FrameStyle
Данное свойство определяет тип линии, используемой для прорисовки рамки объекта. Для получения сведений о допустимых значениях данного свойства обратитесь к справочной системе Delphi к разделу «TPen.Style».

Свойства FrameColor и FillColor
Эти свойства определяют цвет, используемый для рисования рамки объекта и заливки фона. Значением данных свойств может быть любой допустимый цвет или константа типа TColor.

Свойство FormatStr
Данное свойство используется только в объектах «Текст». FormatStr определяет строку формата значений переменных и полей баз данных.

Свойство Visible
Это логическое значение принимает значение «Истина» если объект должен быть видим в сформированном отчете и «Ложь» в противном случае.

Свойство Restrictions
Объявление:
```
Restrictions: Word;
```

Это свойство хранит набор флагов, запрещающих определенные действия над объектом в дизайнере. Отдельные флаги определяются группой констант frrtXXX:
- frrfDontEditMemo – запрещает редактировать свойство «Текст» объекта;
- frrfDontEditScript – запрещает редактирование скрипта;
- frrfDontEditContents – запрещает изменение содержимого объекта;
- frrfDontModify – запрещает изменять опции компонента;
- frrfDontSize – запрещает изменение размера компонента;
- frrfDontMove – запрещает перемещение объекта;
- frrfDontDelete – не позволяет удалить данный компонент из отчета.

Свойство Tag
Здесь может быть строка, содержащая переменные. При построении отчета строка обрабатывается и сохраняется в потоке. Основное назначение – поддержка «живых» отчетов,

т.е. отчетов, реагирующих на мышь в окне предварительного просмотра. Сам объект при этом может содержать число, а в тэг можно поместить идентификатор строки и/или столбца, в котором находится объект.

Свойства Memo и Script
Данные свойства являются хранилищем для свойств «Текст» и «Скрипт» объекта. Данные свойства объявлены как TStringList и могут содержать более одной строки текста.

Методы CalcGaps и RestoreCoord
Метод CalcGaps высчитывает новые координаты/размеры/печатаемую область. Для этого он умножает и смещает координаты/размеры на ScaleX, ScaleY, OffsX, OffsY и сохраняет старые в SaveX, SaveY, … После выполнения необходимых отрисовок надо вызывать RestoreCoord. Методы надо перекрывать, если нужна какая-либо дополнительная обработка, что бывает крайне редко (например компонент RoundRectView).

Метод Draw
Объявление:
```
procedure Draw(Canvas: TCanvas); virtual; abstract;
```

Данный метод отвечает за отрисовку объекта. Метод совмещенный, т.е. вызывается при рисовании объекта в дизайнере, окне предварительного просмотра и при печати. При печати свойство IsPrinting принимает значение «Истина», что позволяет выполнить дополнительную обработку. Перекрывать данный метод надо почти всегда.

Метод StreamOut
Объявление:
```
procedure StreamOut(Stream: TStream); virtual;
```

Раньше этот метод назывался Print. Метод вызывается для сохранения содержимого объекта во внутреннем потоке в процессе формирования отчета. Данный метод следует переопределять только в том случае, если нужна какая-то дополнительная обработка.

Метод ExportData
Объявление:
```
procedure ExportData; virtual;
```

Этот метод вызывается в процессе экспорта отчета с использованием фильтров экспорта. Метод следует переопределять только в том случае, если нужна какая-то дополнительная обработка.

Методы LoadFromStream и SaveToStream
Объявления:
```
procedure LoadFromStream(Stream: TStream); virtual;
procedure SaveToStream(Stream: TStream); virtual;
```

Методы вызываются при необходимости загрузки объекта из потока или сохранения в поток. Маленькая тонкость – методы вызываются при работе с файлами типа FRF и FRP. Во втором случае свойство StreamMode равно smFRP, что позволяет не хранить в FRP ненужные поля (например, содержимое скрипта).

Событие Resized
Данное событие, в данное время, используется только в OLEView. Обработчик этого события вызывается при изменении размеров объекта в дизайнере.

Событие DefinePopupMenu
Объявление:
```
procedure DefinePopupMenu(Popup: TPopupMenu); virtual;
```

Обработчик этого события вызывается при щелчке правой кнопкой мыши на объекте в дизайнере. Переопределять этот обработчик нужно в том случае, если требуется заполнение всплывающего меню своими пунктами.

Метод GetClipRgn
Объявление:
```
function GetClipRgn(rt: TfrRgnType): HRGN; virtual;
```

Метод возвращает регион, занимаемый объектом. Необходим для отрисовки объекта в дизайнере. Переопределять метод нужно в случае, если объект имеет непрямоугольную форму.

Метод SetBounds
Объявление:
```
procedure SetBounds(Left, Top, Width, Height: Integer);
```

Метод позволяет за один прием задать координаты и размеры объекта. То же самое можно сделать и присвоением значений свойствам x, y, dx, dy, но так иногда удобнее.

Метод ShowEditor
Метод вызывается FastReport при двойном щелчке мышью в дизайнере на объекте и предназначен для отображения окна редактора компонента.

Класс TfrControl

TfrControl является наследником TfrView. Данный класс является базовым для создания визуальных элементов управления FastReport - элементов диалоговых форм. Данный класс дополняет список унаследованных от своего предка свойств и методов только свойством Control. Данное свойство имеет тип TControl и должно в конструкторе объекта инициализироваться ссылкой на соответствующий компонент Delphi. Кроме того, класс переопределяет ряд унаследованных методов.

Класс TfrStretcheable

Базовый класс для визуальных растягиваемых и разрываемых объектов. Данный класс добавляет ряд свойств и методов, для обеспечения возможности частичной прорисовки компонентов.

Свойство DrawMode

Объявление:

```
TfrDrawMode = (drAll, drAfterCalcHeight, drPart);
DrawMode: TfrDrawMode;
```

Здесь ядро указывает текущий режим рисования. Само рисование, точнее, запись в поток, осуществляется в методе StreamOut. В случае, если DrawMode имеет значение drPart (объект выводится частями – часть на одной странице, часть на другой), объект должен вывести ту часть содержимого, которая поместилась в текущую высоту объекта (dy). После этого надо запомнить, на чем закончился вывод, чтобы при следующем вызове StreamOut продолжить с этого места.

Метод CalcHeight

Объявление:

```
function CalcHeight: Integer; virtual; abstract;
```

Метод вызывается перед рисованием для выяснения высоты объекта.

Метод MinHeight

Объявление:

```
function MinHeight: Integer; virtual; abstract;
```

Данный метод должен вернуть минимальную высота объекта, при которой он может что-то нарисовать (в случае объекта "Текст" - высота одной строки текста).

Метод RemainHeight

Объявление:

```
function RemainHeight: Integer; virtual; abstract;
```

Метод должен вернуть высоту оставшейся (еще не выведенной) части объекта.

Подключение новых компонент к FastReport

Подключение дополнительных компонент к дизайнеру FastReport осуществляется вызовом процедуры frRegisterObject. Эта процедура объявлена в модуле FR_Class.pas следующим образом:

```
procedure frRegisterObject(ClassRef: TClass; ButtonBmp: TBitmap;
  const ButtonHint: String);
```

Этой процедуре в качестве параметром передаются, соответственно:

- класс нового компонента (именно класс, а не его экземпляр);
- объект TBitmap размером 16x16 с изображением, которое будет отображаться на панели инструментов в дизайнере;
- строка-подсказка для кнопки компонента в дизайнере.

Примеры отчетов

Примеры отчетов

Примеры отчетов

В комплект поставки FastReport входит ряд примеров, демонстрирующих разные приемы работы и способы построения разного рода отчетов. Файлы примеров находится в подкаталоге DEMO основного каталога FastReport.

В стандартную поставку входят 9 демонстрационных проектов, демонстрирующих следующие возможности:

- включение в отчет графиков и диаграмм (каталог CHART);
- хранение шаблона отчета в форме Delphi, а не в файле FRF (каталог DFMSTORE);
- организация рабочей среды для «конечного пользователя» (каталог ENDUSER);
- использование db-aware компонент (каталог ENDUSER1);
- управление логикой построения отчета при помощи события OnManualBuild (каталог MANUAL);
- «ручное» построение отчета без использования дизайнера (каталог RUNTIME);
- печать табличного отчета с неизвестным заранее количеством столбцов (каталог PRNTBL1);
- печать табличного отчета с неизвестным заранее количеством и переменной шириной столбцов (каталог PRNTBL2);
- построение различного рода отчетов при помощи стандартного дизайнера и среды разработки Delphi (каталог REPORTS).

Все перечисленные примеры являются хорошим пособием для тех, кто хочет быстро изучить большую часть возможностей нового генератора отчетов, не перечитывая документацию. Также эти примеры позволяют получить представление о соответствии возможностей генератора отчетов потребностям конкретного пользователя.

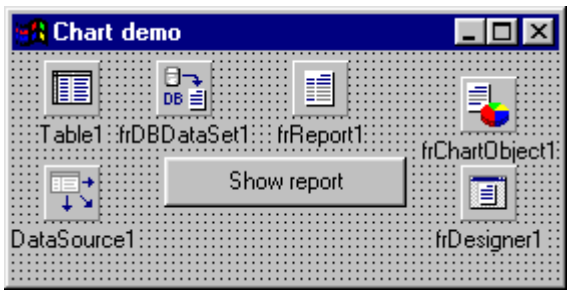
Рассмотрим некоторые из указанных примеров более подробно.

Включение в отчет графиков и диаграмм

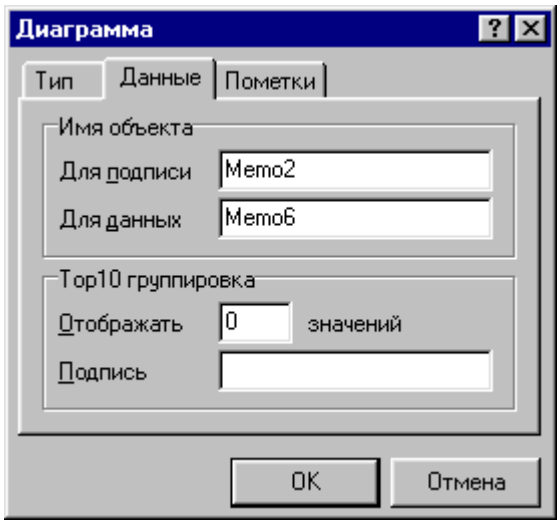
Описываемый здесь пример находится в подкаталоге CHART каталога примеров FastReport. Этот проект содержит всего одну форму, содержащую следующие компоненты:

1. пара компонентов TTable/TDataSource для извлечения данных отчета из таблицы базы данных (в качестве источника данных для данного примера используется таблица *COUNTRY.DB* из стандартных примеров Delphi);
2. компонент TfrDBDataSet для привязки источника данных БД к объекту FastReport;
3. компонент TfrReport – собственно отчет;
4. компонент TfrChartObject – подключаемый модуль «График»;
5. компонент TfrDesigner – дизайнер отчета времени выполнения.

Внешний вид формы представлен на рисунке.



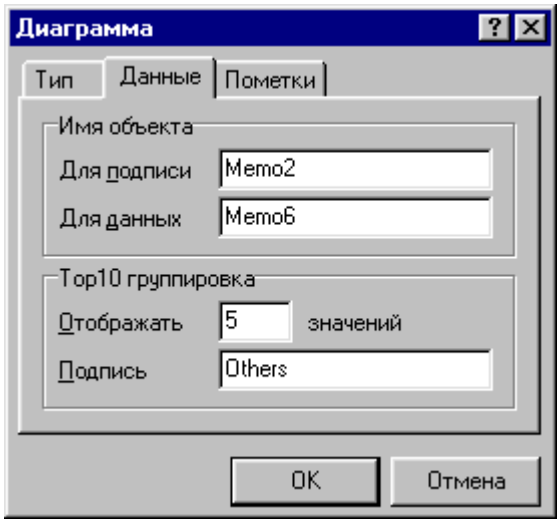
Данный отчет состоит из трех страниц, на каждой из которых находится по одному графику. Первый выводит в виде вертикальных столбцов содержимое поля *AREA* таблицы *COUNTRY*. Второй график представляет аналогичные данные, но по полю *POPULATION*. Ключевым моментом для данных двух графиков является привязка помещенного на форму отчета объекта «Диаграмма» к данным, которые должны быть в нем представлены.
Привязка осуществляется при помощи редактора компонента «Диаграмма», вызываемого двойным щелчком мыши на нужном объекте. На закладке «Данные» данного редактора нужно заполнить поля «Для подписи» и «Для данных» группы «Имя объекта».



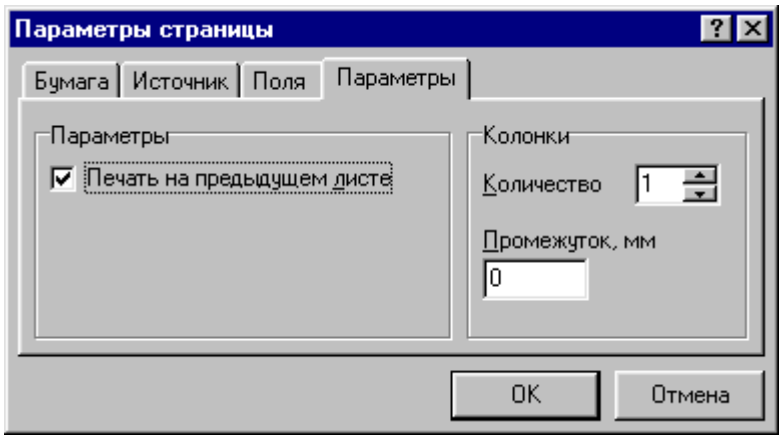
В описываемом примере в этих полях стоит значение Memo2 и Memo6 соответственно. Memo2 и Memo6 – это имена объектов «Текст», связанных с полями таблицы *COUNTRY AREA* и *POPULATION* соответственно.
Таким образом, в объектах «Диаграмма» будут накапливаться данные соответствующих полей всех записей таблицы *COUNTRY*.

Третий график демонстрирует возможность построения т.н. «TopX-графиков», в которых выводятся только X наибольших значений, а сумма остальных представляется

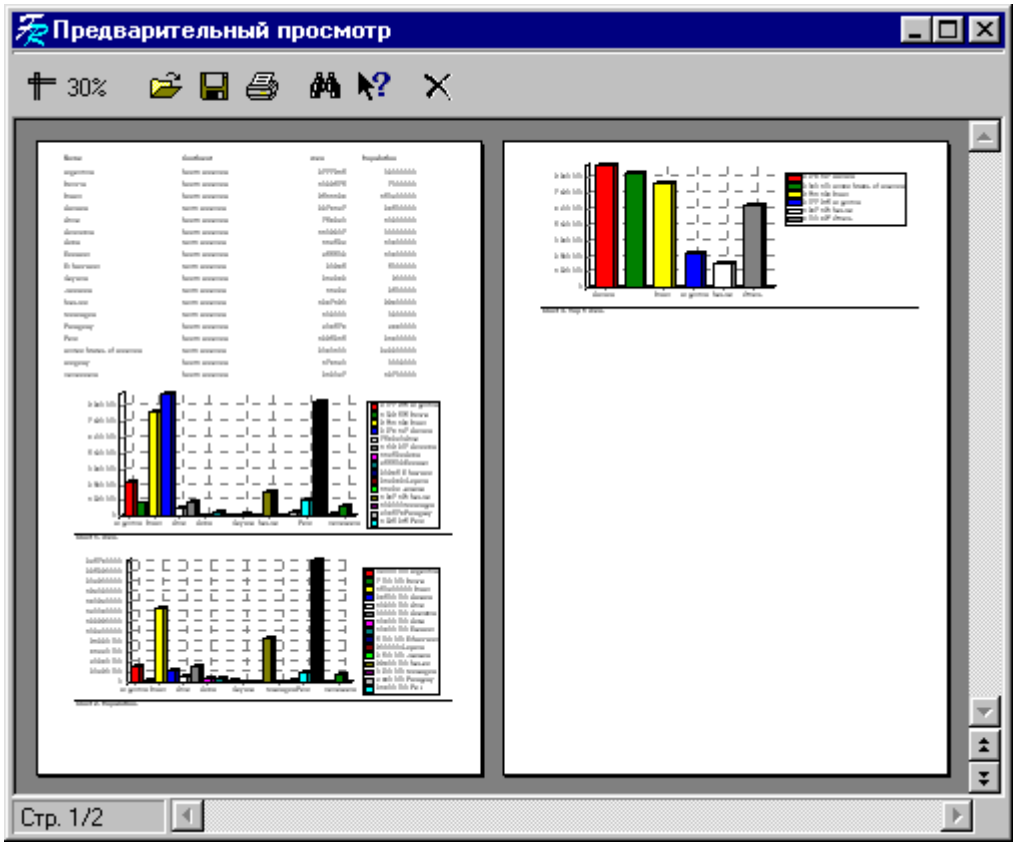
отдельным столбцом как «Другие». Для данного графика ключевым моментом является установка на закладке «Данные» редактора компонента отличного от нуля значения поля «Отображать … значений» и вводом подписи суммы оставшихся значений в поле «Подпись». Внешний вид диалогового окна редактора свойств компонента для отчета Top5 приведен на рисунке.



Еще одним моментом, используемым в данном отчете является печать вывод всех страниц формы отчета вплотную друг к другу. Это позволяет добиться более эффективного заполнения печатной копии отчета и осуществляется установкой соответствующего флажка в диалоге свойств страницы.



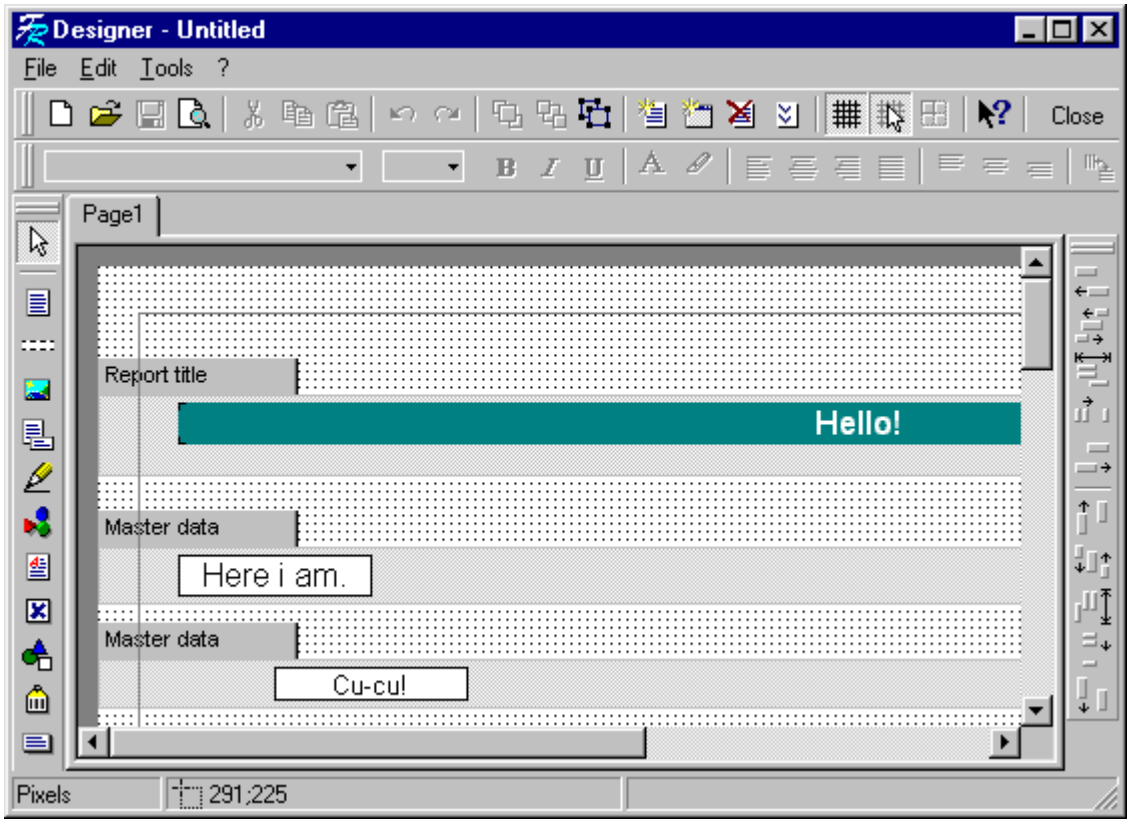
Внешний вид описанного отчета в окне предварительного просмотра представлен ниже:


Предварительный просмотр
30%
Стр. 1/2

Программное управление печатью отчетов при помощи события OnManualBuild

Исходные тексты данного примера находятся в подкаталоге MANUAL.

Единственная форма данного примера содержит только объект TfrReport и кнопку, выводящую отчет на предварительный просмотр. Форма отчета в дизайнере FastReport выглядит так, как показано на рисунке:



Если Вы сейчас нажмете в дизайнере кнопку предварительного просмотра, то увидите только строчку Hello! на зеленом фоне. Это обусловлено тем, что две секции «Master data» не связаны с источником данных и поэтому не выводятся на печать.

Однако если Вы запустите этот пример на выполнение, то получите отчет, в котором кроме указанной строки на четырех страницах напечатан текст из двух секций «Master data». Это происходит благодаря следующему обработчику события OnManualBuild объекта FastReport:

```
procedure TForm1.frReport1ManualBuild(Sender: TfrPage);
var
  i, j: Integer;
begin
  Sender.ShowBandByType(btReportTitle);
  for i := 0 to 3 do
  begin
    Sender.ShowBandByName('Band2');
    for j := 0 to 2 do
      Sender.ShowBandByName('Band3');
    if i <> 3 then
      Sender.NewPage;
  end;
end;
```

В этом обработчике события строка Sender.ShowBandByType(btReportTitle) выводит на первой странице секцию заголовка отчета со всеми расположенными на ней полями, потом в цикле формируются четыре страницы отчета методом Sender.NewPage и три раза на каждой странице печатается секция с текстом «Cu-cu!».

Таким образом при помощи обработчика события OnManualBuild можно управлять логикой процесса построения отчета, что придает отчетам большую гибкость.

Построение отчета при помощи кода

Иногда возникает необходимость печати отчета, структура которого неизвестна на стадии проектирования приложения или может меняться в процессе работы программы. В этом случае встает необходимость динамического формирования шаблона отчета. При этом в отличие от ситуации, скажем, с добавленем нового бланка для бухгалтерской программы, где можно использовать принцип т.н. end-user-отчетов, форма отчета не является фиксированной и, соответственно, не может быть определена программистом в виде отдельного файла.

FastReport предоставляет возможность динамического создания форм отчетов с помощью программного кода, как это можно делать, например, с объектами VCL. Пример программного построения формы отчета находится в подкаталоге RUNTIME каталога примеров FastReport.

В этом примере по нажатию кнопки строится отчет, позволяющий получить список компаний из таблицы *CUSTOMER.DB*. Текст обработчика события OnClick этой кнопки приведен ниже:

```
procedure TForm1.Button1Click(Sender: TObject);
var
  v: TfrView;
  b: TfrBandView;
  Page: TfrPage;
begin
  frReport1.Pages.Clear;
  frReport1.Pages.Add;              // create page
  Page := frReport1.Pages[0];

  b := TfrBandView(frCreateObject(gtBand, ''));  // create MasterData band
  b.SetBounds(0, 20, 0, 20);
  b.BandType := btMasterData;
  b.Dataset := 'frDBDataSet1';
  Page.Objects.Add(b);

  v := frCreateObject(gtMemo, '');  // create data field
  v.SetBounds(20, 20, 200, 16);
  v.Memo.Add('[Table1."Company"]');
  Page.Objects.Add(v);

  frReport1.ShowReport;
end;
```

Этот код сначала удаляет все имеющиеся в форме отчета страницы, а потом создает одну чистую:

```
frReport1.Pages.Clear;
frReport1.Pages.Add;              // create page
```

Далее на этой странице создается одна секция типа «Master data» и привязывается к источнику данных.

```
Page := frReport1.Pages[0];

b := TfrBandView(frCreateObject(gtBand, ''));  // create MasterData band
b.SetBounds(0, 20, 0, 20);
b.BandType := btMasterData;
b.Dataset := 'frDBDataSet1';
Page.Objects.Add(b);
```

Следующим шагом создается один объект «Текст» с привязкой его к полю *COMPANY* таблицы *CUSTOMER.DB*:

```
v := frCreateObject(gtMemo, '');  // create data field
v.SetBounds(20, 20, 200, 16);
v.Memo.Add('[Table1."Company"]');
Page.Objects.Add(v);
```

Последнее, что делает указанный обработчик события – это выводит подготовленный отчет на предварительный просмотр.

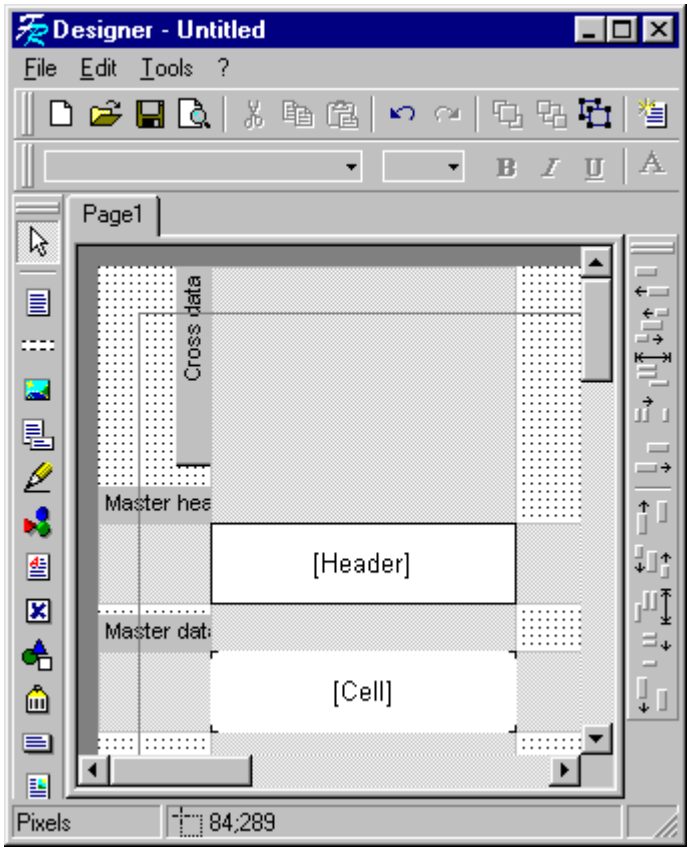
Печать табличных отчетов с переменным или неизвестным количеством столбцов

На практике часто возникает необходимость напечатать табличный отчет, в котором количество столбцов заранее неизвестно или может меняться во время работы с программой. Такая необходимость может возникнуть, например, при разработке редактора баз данных с возможностью вывода данных таблицы на печать. В этом случае можно пойти по пути программного построения формы отчета, однако FastReport предоставляет более простое решение данного вопроса.

Для этого предназначены так называемые CrossTab-отчеты. Отличительной особенностью такого отчета является то, что данные выводятся в несколько столбцов, количество которых заранее не определено. Секции CrossXXX, специфичные для данного типа отчетов, располагаются не горизонтально, а вертикально. При этом объекты «Текст» располагаются на пересечениях горизонтальных (например «Master data») и вертикальных (например «Cross data») секций. Отчет в этом случае формируется по следующему принципу: для всех записей источника данных горизонтальной секции перебираются все записи источника данных вертикальной секции.

Так, при печати содержимого таблицы источником данных для «Master data» являются записи самой таблицы, а для секции «Cross data» источником является список полей каждой конкретной записи.

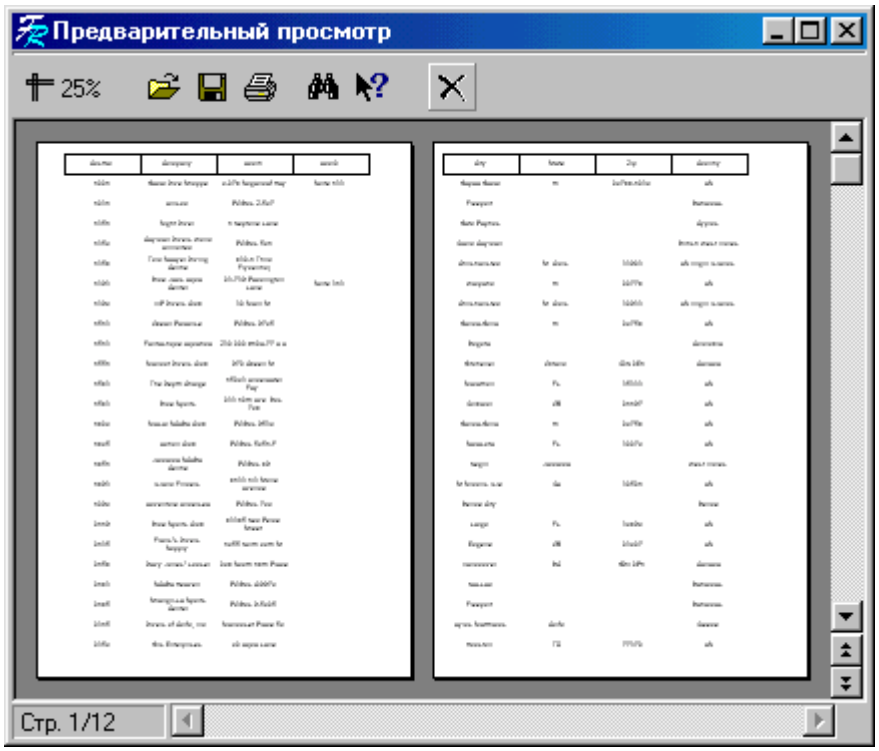
Простейший пример CrossTab-отчета находится в каталоге PRNTBL1. Форма отчета из этого примера в дизайнере выглядит так, как показано на рисунке.



На этом рисунке видно, что объекты «Текст» располагаются на пересечениях секций «Master header» и «Cross data», а также «Master data» и «Cross data». При построении отчета для секции «Master data» в качестве источника записей используется таблица *CUSTOMER.DB*, а количество столбцов отчета определяется двумя виртуальными источниками TfrUserDataset, количество «записей» в которых соответствует количеству полей таблицы *CUSTOMER.DB*. Значения полей выбираются в событии OnGetValue объекта TfrReport:

```
procedure TForm1.frReport1GetValue(const ParName: String; var ParValue: Variant);
begin
  if ParName = 'Cell' then
    ParValue := Table1.Fields[frUserDataset1.RecNo].Value;
  if ParName = 'Header' then
    ParValue := Table1.Fields[frUserDataset2.RecNo].FieldName;
end;
```

В результате при запуске данного примера на предварительный просмотр выводится содержимое всех записей таблицы *CUSTOMER.DB*. Изображение окна предварительного просмотра с данными таблицы приведено на рисунке.



Табличные отчеты с переменной шириной столбца

В предыдущем примере подразумевалось, что все столбцы табличного отчета имеют одинаковую ширину. Однако на практике, в большинстве случаев, ширина каждого столбца определяется представляемыми им данными. Так если, например, имеется два столбца, один из которых содержит номера строк, а второй представляет данные из Memo-поля, то ясно, что первый столбец будет иметь ширину намного меньшую, чем тот, в котором выводится текст Memo-поля.

В подкаталоге PRNTBL2 находится модифицированный пример. Этот пример полностью аналогичен предыдущему за тем лишь исключением, что ширина столбца отчета подбирается в зависимости от содержащихся в нем данных. В этом примере у объекта TfrReport появились два новых обработчика событий, которые и обеспечивают динамическое изменение ширины столбцов.

Обработчик события OnPrintColumn объекта TfrReport отвечает за определение ширины столбца на основании представляемых им данных. В рассматриваемом примере ширина столбца для текстовых полей определяется как произведение размера поля на ширину буквы «W»; для полей «Дата и время» ширина столбца определяется аналогичным образом, но размер поля принимается равным 15; ширина всех остальных столбцов принимается равной 64 пикселя.

Полный текст обработчика события OnPrintColumn приведен ниже:

```
procedure TForm1.frReport1PrintColumn(ColNo: Integer; var Width: Integer);
var
  Field: TField;
begin
  Field := Table1.Fields[ColNo - 1];
  if Field is TStringField then
    Width := Field.Size * Canvas.TextWidth('W')
  else if Field is TDateTimeField then
    Width := 15 * Canvas.TextWidth('W')
  else
    Width := 64;
  FWidth := Width;
end;
```

Обработчик события OnEnterRect реально устанавливает ширину столбца определенную в предыдущем обработчике события:

```
procedure TForm1.frReport1EnterRect(Memo: TStringList; View: TfrView);
begin
  View.dx := FWidth;
end;
```

Таким образом в этом примере достигается более эффективное заполнение места на листе при печати отчета. При этом каждый столбец имеет ширину, достаточную для вывода представляемых им данных. На этом принципе работают компоненты TfrPrintTable и TfrPrintGrid, которые печатают содержимое таблицы и компонента TDBGrid.