

Евгений Марков
Петр Дарахвелидзе



Delphi 2005

для Win32

- Пользовательский интерфейс и бизнес-логика программирования
- Использование XML
- Современные технологии доступа к данным
- Распределенные многозвенные приложения
- Web-приложения и Web-службы

**Наиболее
полное
руководство**

+CD-ROM



В ПОДЛИННИКЕ®

УДК 681.3.068+800.92Delphi2005
ББК 32.973.26-018.1
Д20

Дарахвелидзе П. Г., Марков Е. П.

Д20 Delphi 2005 для Win32. — СПб.: БХВ-Петербург, 2005. — 1136 с.: ил.
ISBN 5-94157-700-1

Рассмотрены практические аспекты программирования в Borland Delphi 2005 для Win32. Приведено детальное описание объектной модели Delphi, обсуждаются вопросы разработки бизнес-логики приложений. Большое внимание уделено созданию приложения для работы с базами данных. Рассмотрены технологии доступа к данным Borland DataBase Engine, dbExpress, Interbase Express, dbGo, а также создание распределенных приложений баз данных и технология DataSnap. Показана работа с редактором отчетов Rave Report 6.0 и преобразование данных в формат XML. Описана разработка Internet-приложений, распределенные Web-приложения и Web-службы. Все рассматриваемые темы сопровождаются подробными примерами, исходные коды которых находятся на прилагаемом компакт-диске.

Для подготовленных программистов

УДК 681.3.068+800.92Delphi2005
ББК 32.973.26-018.1

Группа подготовки издания:

Главный редактор	<i>Екатерина Кондукова</i>
Зам. главного редактора	<i>Игорь Шишигин</i>
Зав. редакцией	<i>Григорий Добин</i>
Редактор	<i>Леонид Кочин</i>
Компьютерная верстка	<i>Натальи Смирновой</i>
Корректор	<i>Наталья Першакова</i>
Дизайн обложки	<i>Игоря Цырульникова</i>
Зав. производством	<i>Николай Тверских</i>

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 31.08.05.

Формат 70×100^{1/16}. Печать офсетная. Усл. печ. л. 91,59.

Тираж 3000 экз. Заказ №

"БХВ-Петербург", 194354, Санкт-Петербург, ул. Есенина, 5Б.

Санитарно-эпидемиологическое заключение на продукцию № 77.99.02.953.Д.006421.11.04
от 11.11.2004 г. выдано Федеральной службой по надзору
в сфере защиты прав потребителей и благополучия человека.

Отпечатано с готовых диапозитивов
в ОАО "Техническая книга"
190005, Санкт-Петербург, Измайловский пр., 29.

ISBN 5-94157-700-1

© Дарахвелидзе П. Г., Марков Е. П., 2005
© Оформление, издательство "БХВ-Петербург", 2005

Оглавление

Часть I. Основы	1
Глава 1. Что и как можно разрабатывать в Delphi 2005.....	3
Многоязычная среда разработки.....	3
Язык программирования Delphi.....	4
Язык программирования C#.....	6
Язык программирования Visual Basic.....	6
Программные платформы.....	7
Компоненты .NET и VCL.....	8
Какие приложения можно создавать в Delphi.....	9
Перенос существующих приложений на платформу .NET.....	11
Резюме.....	13
Глава 2. Типы данных в Delphi.....	14
Простые типы.....	14
Строковый тип.....	17
Структурные типы.....	18
Множества.....	19
Массивы.....	19
Статические массивы.....	19
Динамические массивы.....	21
Многомерные динамические массивы.....	22
Записи.....	23
Варианты.....	24
Резюме.....	26
Глава 3. Объектно-ориентированное программирование.....	27
Объект и класс.....	28
Поля, свойства и методы.....	31
События.....	34
Инкапсуляция.....	38
Наследование.....	39
Полиморфизм.....	40
Методы.....	42
Перегрузка методов.....	45
Области видимости.....	47
Объект изнутри.....	51
Резюме.....	56

Глава 4. Библиотека визуальных компонентов VCL и ее базовые классы.....	57
Иерархия базовых классов	57
Класс <i>TObject</i>	60
Класс <i>TPersistent</i>	62
Класс <i>TComponent</i>	63
Базовые классы элементов управления	66
Класс <i>TControl</i>	67
Местоположение и размер элемента управления	67
Выравнивание элемента управления.....	69
Внешний вид элемента управления.....	70
Связь с родительским элементом управления	72
Класс <i>TWinControl</i>	73
Класс <i>TCustomControl</i>	75
Класс <i>TGraphicControl</i>	76
Резюме.....	76
Глава 5. Обработка исключительных ситуаций.....	77
Исключительная ситуация как класс.....	77
Защитные конструкции языка Delphi.....	83
Блок <i>try..except</i>	83
Блок <i>try..finally</i>	86
Использование исключительных ситуаций	88
Протоколирование исключительных ситуаций	91
Коды ошибок в исключительных ситуациях.....	92
Исключительная ситуация <i>EAbort</i>	96
Функция <i>Assert</i>	97
Резюме.....	98
Часть II. ИНТЕРФЕЙС И ЛОГИКА ПРИЛОЖЕНИЯ	99
Глава 6. Приложение и проект	101
Проект как основа разработки приложения	101
Репозиторий Delphi.....	108
Класс <i>TApplication</i>	109
Атрибуты приложения.....	116
Обработка сообщений.....	117
Реакция на действия пользователей.....	121
Система помощи	121
Резюме.....	122
Глава 7. Форма.....	123
Роль формы в приложении	124
Класс <i>TForm</i>	127
Создание и уничтожение формы.....	135
Визуализация формы	137

Атрибуты и стили формы	139
Управление компонентами формы	140
Кадры (Frames). Класс <i>TFrame</i>	140
Резюме	141
Глава 8. Элементы управления Win32 и XP	142
Что такое библиотека <i>ComCtl32</i>	142
Многостраничный блокнот — компоненты <i>TPageControl</i> и <i>TTabControl</i>	144
Компонент <i>TToolBar</i>	149
Компонент <i>TImageList</i>	153
Компоненты <i>TTreeView</i> и <i>TListView</i>	155
Календарь	169
Компонент <i>TMonthCalendar</i>	169
Компонент <i>TDateTimePicker</i>	170
Панель состояния <i>TStatusBar</i>	173
Расширенный комбинированный список <i>TComboBoxEx</i>	173
Создание нового компонента на базе элементов управления из библиотеки <i>ComCtl32</i>	174
Пользовательский интерфейс Windows XP	186
Манифест Windows XP	187
Компонент <i>TXPManifest</i>	189
Включение манифеста Windows XP в ресурсы приложения	190
Визуальные стили и темы оформления	190
Визуальные стили в Delphi	191
Theme API	194
Компоненты настройки цветовой палитры	196
Резюме	197
Глава 9. Меню и действия	198
Редактор меню	199
Как работает меню	200
Главное меню приложения	204
Всплывающее меню	207
Действия. Компонент <i>TActionList</i>	207
События, связанные с действиями	209
Свойства, распространяемые на клиентов действия	210
Прочие свойства	212
Стандартные действия	213
Категория <i>Edit</i>	215
Категория <i>Search</i>	216
Категория <i>Help</i>	216
Категория <i>File</i>	216
Категория <i>Dialog</i>	217
Категория <i>Window</i>	217
Категория <i>Tab</i>	217

Категория <i>List</i>	217
Категория <i>Internet</i>	218
Категория <i>Format</i>	219
Категория <i>Dataset</i>	220
Категория <i>Tools</i>	220
Компонент <i>TActionManager</i>	220
Изменение и настройка внешнего вида панелей	222
Ручное редактирование коллекций панелей и действий	224
Резюме	227
Глава 10. Списки и коллекции	228
Список строк	229
Класс <i>TStrings</i>	229
Класс <i>TStringList</i>	230
Список указателей	237
Класс <i>TList</i>	238
Пример использования списка указателей	241
Коллекции	245
Класс <i>TCollection</i>	246
Класс <i>TCollectionItem</i>	247
Резюме	248
Глава 11. Файлы и потоки	249
Использование файловых переменных. Типы файлов	249
Операции ввода-вывода	251
Ввод-вывод с использованием функций Windows API	258
Отложенный (асинхронный) ввод-вывод	262
Контроль ошибок ввода-вывода	264
Атрибуты файла. Поиск файла	265
Потоки	267
Базовые классы <i>TStream</i> и <i>THandleStream</i>	267
Класс <i>TFileStream</i>	269
Класс <i>TMemoryStream</i>	271
Класс <i>TStringStream</i>	272
Оповещение об изменениях в файловой системе	273
Использование отображаемых файлов	274
Резюме	278
Глава 12. Использование графики	279
Графические инструменты Delphi	279
Класс <i>TFont</i>	280
Класс <i>TPen</i>	281
Класс <i>TBrush</i>	282
Класс <i>TCanvas</i>	282
Класс <i>TGraphic</i>	288

Класс <i>TPicture</i>	290
Класс <i>TMetafile</i>	292
Класс <i>TIcon</i>	293
Класс <i>TBitmap</i>	293
Графический формат JPEG. Класс <i>TJPEGImage</i>	298
Компонент <i>TImage</i>	301
Использование диалогов для загрузки и сохранения графических файлов	302
Класс <i>TClipboard</i>	310
Класс <i>TScreen</i>	311
Вывод графики с использованием отображаемых файлов.....	315
Класс <i>TAnimate</i>	318
Резюме.....	319
Часть III. Использование XML.....	321
Глава 13. Основы XML.....	323
Что такое XML	323
Основы синтаксиса XML	326
Пролог	328
Определение.....	329
Тело документа. Корневой элемент	330
Объектная модель документа.....	331
Интерфейсы семейства <i>IDOMNode</i>	332
Свойства <i>nodeType</i> , <i>nodeName</i> и <i>nodeValue</i>	334
Свойства и методы, управляющие другими вершинами.....	335
Пространства имен.....	337
Интерфейс <i>IDOMDocument</i>	338
Пример создания приложения, использующего модель DOM.....	338
Реализация модели DOM в Delphi.....	340
Модуль <i>xmldom</i>	342
Модуль <i>msxml</i>	342
Модуль <i>msxmldom</i>	342
Модуль <i>XMLIntf</i>	342
Модуль <i>XMLDoc</i>	342
Интерфейс <i>IXMLNode</i> и его отличия от стандарта DOM.....	343
Взаимосвязь между всеми интерфейсами.....	345
Загрузка XML	347
Асинхронная загрузка	349
Функции, создающие экземпляр документа	349
Обработка ошибок анализатора.....	350
Пример использования интерфейсов <i>IXMLNode</i> и <i>IXMLDocument</i>	352
Анализатор MSXML, или Microsoft XML Core Services	364
Резюме.....	371

Глава 14. Преобразование данных в формате XML	373
Преобразование данных в формате XML.....	373
Схема преобразования данных XML	374
Формат пакета данных Delphi.....	375
Инструментарий преобразования данных XML	376
Утилита XML Mapper	377
Выбор исходного файла.....	378
Создание пакета данных и документа XML и сохранение преобразованных данных	379
Связывание элементов XML и полей пакета данных	380
Создание трансформационного файла и преобразование данных	381
Резюме.....	382
Часть IV. ПРИЛОЖЕНИЯ БАЗ ДАННЫХ	383
Глава 15. Архитектура приложений баз данных	385
Как работает приложение баз данных	387
Модуль данных.....	389
Подключение набора данных	391
Настройка компонента <i>TDataSource</i>	393
Отображение данных	395
Резюме.....	397
Глава 16. Набор данных.....	398
Абстрактный набор данных	400
Стандартные компоненты	405
Компонент таблицы	406
Компонент запроса	408
Компонент хранимой процедуры	411
Индексы в наборе данных.....	412
Механизм подключения индексов	413
Список описаний индексов	413
Описание индекса	414
Использование описаний индексов	415
Параметры запросов и хранимых процедур.....	417
Класс <i>TParams</i>	420
Класс <i>TParam</i>	421
Состояния набора данных.....	423
Резюме.....	426
Глава 17. Поля и типы данных	427
Объекты полей.....	427
Статические и динамические поля	430
Класс <i>TField</i>	432

Виды полей	436
Поля синхронного просмотра.....	437
Вычисляемые поля.....	439
Внутренние вычисляемые поля	440
Агрегатные поля	440
Объектные поля.....	442
Типы данных.....	442
Ограничения	447
Резюме	451
Глава 18. Управление данными	452
Связанные таблицы.....	452
Отношение "один ко многим"	453
Отношение "многие ко многим".....	455
Поиск данных.....	456
Поиск по индексам	456
Поиск в диапазоне	458
Поиск по произвольным полям	458
Фильтры	459
Быстрый переход к помеченным записям	461
Диапазоны	464
Резюме.....	465
Глава 19. Компоненты отображения данных.....	466
Классификация компонентов отображения данных.....	466
Табличное представление данных.....	468
Компонент <i>TDBGrid</i>	468
Компонент <i>TDBCtrlGrid</i>	479
Навигация по набору данных	481
Представление отдельных полей.....	484
Компонент <i>TDBText</i>	485
Компонент <i>TDBEdit</i>	485
Компонент <i>TDBCheckBox</i>	486
Компонент <i>TDBRadioGroup</i>	486
Компонент <i>TDBListBox</i>	487
Компонент <i>TDBComboBox</i>	487
Компонент <i>TDBMemo</i>	487
Компонент <i>TDBImage</i>	488
Компонент <i>TDBRichEdit</i>	488
Синхронный просмотр данных	489
Механизм синхронного просмотра	490
Компонент <i>TDBLookupListBox</i>	492
Компонент <i>TDBLookupComboBox</i>	492
Графическое представление данных	493
Резюме.....	495

ЧАСТЬ V. ТЕХНОЛОГИИ ДОСТУПА К ДАННЫМ.....	497
Глава 20. Процессор баз данных Borland Database Engine.....	499
Архитектура и функции BDE	500
Псевдонимы баз данных и настройка BDE	503
Интерфейс прикладного программирования BDE.....	513
Соединение с источником данных.....	524
Компоненты доступа к данным.....	529
Класс <i>TBDEDataSet</i>	529
Класс <i>TDBDataSet</i>	534
Компонент <i>TTable</i>	536
Компонент <i>TQuery</i>	542
Компонент <i>TStoredProc</i>	545
Резюме.....	546
Глава 21. Технология dbExpress.....	547
Доступ к данным dbExpress.....	548
Драйверы доступа к данным.....	550
Соединение с сервером баз данных.....	550
Управление наборами данных.....	556
Транзакции	558
Использование компонентов наборов данных	560
Класс <i>TCustomSQLDataSet</i>	560
Компонент <i>TSQLDataSet</i>	563
Компонент <i>TSQLTable</i>	563
Компонент <i>TSQLQuery</i>	564
Компонент <i>TSQLStoredProc</i>	565
Компонент <i>TSimpleDataSet</i>	567
Способы редактирования данных	570
Интерфейсы dbExpress.....	575
Интерфейс <i>ISQLDriver</i>	575
Интерфейс <i>ISQLConnection</i>	576
Интерфейс <i>ISQLCommand</i>	577
Интерфейс <i>ISQLCursor</i>	579
Отладка приложений с технологией dbExpress.....	579
Распространение приложений с технологией dbExpress.....	582
Резюме.....	582
Глава 22. Технология InterBase Express.....	584
Механизм доступа к данным InterBase Express.....	585
Компонент <i>TIBDatabase</i>	585
Компонент <i>TIBTransaction</i>	590
Компоненты доступа к данным.....	594
Область дескрипторов <i>XSQLDA</i>	596
Структура <i>XSQLVAR</i>	597

Компонент <i>TIBTable</i>	599
Компонент <i>TIBQuery</i>	599
Компонент <i>TIBStoredProc</i>	601
Компонент <i>TIBDataSet</i>	601
Компонент <i>TIBSQL</i>	604
Обработка событий	607
Информация о состоянии базы данных	609
Компонент <i>TIBDatabaseInfo</i>	609
Компонент <i>TIBSQLMonitor</i>	611
Резюме	612
Глава 23. Технология dbGo	613
Основы ADO	613
Перечислители	616
Объекты соединения с источниками данных	617
Сессия	617
Транзакции	618
Наборы рядов	618
Команды	619
Провайдеры ADO	620
Технология dbGo	621
Компоненты ADO	621
Механизм соединения с хранилищем данных ADO	622
Компонент <i>TADOConnection</i>	622
Настройка соединения	623
Управление соединением	628
Доступ к связанным наборам данных и командам ADO	632
Объект ошибок ADO	634
Транзакции	634
Наборы данных ADO	636
Класс <i>TCustomADODataSet</i>	636
Набор данных	637
Курсор набора данных	638
Локальный буфер	639
Состояние записи	640
Фильтрация	642
Поиск	643
Сортировка	644
Команда ADO	644
Групповые операции	646
Параметры	647
Класс <i>TParameters</i>	647
Класс <i>TParameter</i>	648
Компонент <i>TADODataSet</i>	650
Компонент <i>TADOTable</i>	650

Компонент <i>TADOQuery</i>	651
Компонент <i>TADOStoredProc</i>	652
Команды ADO	652
Объект ошибок ADO	654
Пример приложения ADO	655
Соединение с источником данных.....	659
Реализация групповых операций.....	660
Реализация фильтрации.....	660
Реализация сортировки	660
Резюме.....	660
Часть VI. Приложения COM и COM+	663
Глава 24. Механизмы COM	665
Базовые понятия	666
Объект.....	669
Интерфейс.....	670
Интерфейс <i>IUnknown</i>	672
Сервер.....	672
Библиотека COM.....	673
Фабрика класса.....	675
Библиотека типов.....	676
Объекты COM в Delphi	676
Класс <i>TComObject</i>	677
Класс <i>TTypedComObject</i>	679
Интерфейс <i>IUnknown</i> в Delphi.....	679
Тип глобального идентификатора	680
Класс <i>TInterfacedObject</i>	680
Фабрика класса в Delphi	681
Класс <i>TComObjectFactory</i>	682
Класс <i>TTypedComObjectFactory</i>	684
Класс <i>TComClassManager</i>	684
Сервер COM в Delphi	685
Класс <i>TComServer</i>	685
Библиотека типов в Delphi.....	687
Простой объект COM в составе внутреннего сервера	689
Создание объекта	690
Создание методов интерфейса.....	697
Регистрация внутреннего сервера.....	704
Использование интерфейсов внутреннего сервера COM	704
Резюме.....	706
Глава 25. Технология Автоматизация.....	707
Базовые понятия Автоматизации	707
Виды интерфейсов Автоматизации	708

Диспинтерфейсы.....	708
Дуальные интерфейсы.....	710
Библиотека типов.....	710
Маршалинг интерфейсов Автоматизации	710
Объект Автоматизации.....	711
Сервер Автоматизации.....	711
Контроллер Автоматизации	711
Реализация Автоматизации в Delphi	711
Интерфейсы Автоматизации	712
Интерфейс <i>IDispatch</i>	712
Интерфейсы диспетчеризации.....	713
Дуальные интерфейсы	714
Объект Автоматизации	714
Класс <i>TAuto Object</i>	715
Обработка событий объекта Автоматизации.....	717
Создание методов-аналогов событий	720
Создание класса-оболочки	720
Инициализация объекта Автоматизации	721
Фабрика класса.....	722
Класс <i>TAutoInfObject</i>	723
Сервер Автоматизации.....	724
Контроллер автоматизации	725
Пример приложения Автоматизации	728
Сервер Автоматизации.....	728
Контроллер Автоматизации	732
Резюме.....	733
Глава 26. Элементы управления ActiveX.....	734
Как работают элементы управления ActiveX	735
Контейнеры и регистрация элементов управления ActiveX.....	738
Предоставление методов.....	739
События.....	739
Свойства	739
Страницы свойств	740
Лицензирование	740
Реализация компонентов ActiveX в Delphi.....	741
Класс компонента ActiveX	742
Фабрика класса компонента ActiveX	743
Среда разработки Delphi как контейнер ActiveX.....	743
Регистрация компонентов ActiveX	744
Использование готовых компонентов ActiveX	744
Инсталляция готовых элементов управления ActiveX	744
Пример инсталляции элемента управления <i>TWebBrowser</i>	746
Резюме.....	750

ЧАСТЬ VII. РАСПРЕДЕЛЕННЫЕ ПРИЛОЖЕНИЯ БАЗ ДАННЫХ.....	751
Глава 27. Технология Data Snap.....	753
Структура многозвенного приложения в Delphi	754
Трехзвенное приложение в Delphi	756
Сервер приложения.....	757
Клиентское приложение.....	758
Механизм удаленного доступа к данным DataSnap	759
Компонент <i>TDCOMConnection</i>	759
Компонент <i>TSocketConnection</i>	761
Компонент <i>TWebConnection</i>	764
Вспомогательные компоненты-брокеры соединений	765
Компонент <i>TSimpleObjectBroker</i>	765
Компонент <i>TLocalConnection</i>	767
Компонент <i>TSharedConnection</i>	768
Компонент <i>TConnectionBroker</i>	769
Резюме.....	769
Глава 28. Сервер приложения	770
Структура сервера приложения	771
Интерфейс <i>AppServer</i>	772
Удаленные модули данных.....	775
Удаленный модуль данных для сервера Автоматизации.....	775
Дочерние удаленные модули данных.....	780
Провайдеры данных.....	781
Интерфейс <i>IProviderSupport</i>	786
Регистрация сервера приложения	786
Пример простого сервера приложения.....	787
Главная форма сервера приложения.....	787
Главный удаленный модуль данных.....	788
Дочерний удаленный модуль данных	789
Регистрация сервера приложения	791
Резюме.....	791
Глава 29. Клиент многозвенного распределенного приложения	792
Структура клиентского приложения	793
Компонент <i>TClientDataSet</i>	794
Получение данных от компонента-провайдера	795
Кэширование и редактирование данных.....	797
Управление запросом на сервере.....	799
Использование индексов.....	800
Сохранение набора данных в файлах.....	802
Работа с данными типа BLOB	803
Представление данных в формате XML	804

Агрегаты	804
Объекты-агрегаты	805
Агрегатные поля	807
Группировка и использование индексов	809
Вложенные наборы данных	809
Дополнительные свойства полей клиентского набора данных	810
Обработка ошибок	811
Пример "тонкого" клиента	814
Соединение клиента с сервером приложения	816
Наборы данных клиентского приложения	818
Резюме	819
Глава 30. Клиент распределенного приложения на основе данных XML.....	820
Преобразование данных XML в распределенных приложениях	820
Использование данных XML в распределенных приложениях	824
Данные XML в клиентском наборе данных	826
Данные XML в документе XML	826
Данные XML на странице HTML	827
Пример приложения, использующего данные XML	830
Резюме	833
ЧАСТЬ VIII. ГЕНЕРАТОР ОТЧЕТОВ RAVE REPORTS 6.0.....	835
Глава 31. Отчеты Rave Reports в приложении	837
Генератор отчетов Rave Reports 6.....	837
Компоненты Rave Reports и их назначение.....	838
Отчет в приложении Delphi	839
Компонент отчета <i>TRvProject</i>	841
Компонент управления отчетом <i>TRvSystem</i>	843
Резюме	849
Глава 32. Визуальная среда создания отчетов.....	850
Инструментарий визуальной среды создания отчетов.....	851
Проект отчета	854
Библиотека отчетов	855
Каталог глобальных страниц.....	856
Словарь просмотра данных	856
Стандартные элементы оформления и их свойства	857
Элементы для представления текста и изображений	858
Графические элементы	859
Штрихкоды	859
Обработка событий	860
Внешние источники данных в отчете	861
Соединение с источником данных и просмотра	861
Безопасность доступа к данным	863

Отображение данных в отчетах	863
Структурные элементы отчета	863
Элементы отображения данных.....	866
Резюме.....	867
Глава 33. Разработка, просмотр и печать отчетов.....	868
Этапы создания отчета и включение его в приложение.....	869
Простой отчет в визуальной среде Rave Reports	869
Нумерация страниц отчета.....	870
Использование элемента <i>FontMaster</i>	871
Добавление страниц к отчету.....	871
Отчет в приложении	873
Просмотр и печать отчета	875
Сохранение отчета во внешнем файле	876
Компонент <i>TRvNDRWriter</i>	877
Преобразование форматов данных.....	879
Резюме.....	880
Глава 34. Отчеты для приложений баз данных	881
Соединения с источниками данных в Rave Reports.....	882
Соединения с источниками данных в визуальной среде Rave Reports.....	884
Соединение через драйвер Rave Reports.....	884
Соединение через компонент приложения Delphi	886
Соединения с источниками данных в приложении	887
Компонент <i>TRvDataSetConnection</i>	887
Компоненты, использующие BDE	890
Компонент <i>TRvCustomConnection</i>	890
Аутентификация пользователя в отчете.....	893
Типы отчетов	894
Простой табличный отчет	894
Отчет "один ко многим"	896
Группирующий отчет	898
Использование вычисляемых значений	899
Вычисляемые значения по одному источнику	900
Вычисляемые значения по нескольким источникам	901
Управляющие вычислительные элементы.....	903
Резюме.....	904
Часть IX. Основы разработки INTERNET-ПРИЛОЖЕНИЙ	905
Глава 35. Введение в архитектуру сетей.....	907
Модель OSI	908
Физический уровень	910
Протоколы канального уровня.....	911

Управление доступом к среде	911
Управление логическим каналом (Logical Link Control)	913
Функции сетевого уровня	914
Транспортный уровень	917
Концепции сессионного уровня, уровней представления и приложений.....	918
Сессионный (session) уровень	919
Уровень представления.....	919
Уровень приложения	919
Резюме.....	920
Глава 36. Криптографическая защита информации в Internet	921
Основные термины и понятия криптографии	922
Секретность.....	922
Аутентификация	925
Целостность	926
Цифровые подписи, сертификаты и их применение.....	927
Использование <i>CryptoAPI</i>	933
Структура <i>CryptoAPI</i>	934
Пример использования <i>CryptoAPI</i> – менеджер сертификатов	936
Служба сертификатов Microsoft.....	941
Протоколы Internet для защищенных соединений	943
Настройка SSL на стороне сервера IIS 6.....	945
Настройка протокола SSL на клиентской стороне	949
Резюме.....	951
ЧАСТЬ X. РАСПРЕДЕЛЕННЫЕ WEB-ПРИЛОЖЕНИЯ И WEB-СЛУЖБЫ.....	953
Глава 37. Серверные Web-приложения.....	955
Архитектура серверных Web-приложений	956
Типы серверных Web-приложений	958
Введение в интерфейсы CGI и ISAPI.....	959
Как работает серверное Web-приложение	961
Классы приложений.....	962
Действия.....	963
Запросы и ответы	965
Модули данных.....	966
Использование отладочного сервера Web App Debugger.....	967
Перенос тестового приложения на рабочую платформу	968
Особенности отладки приложений ISAPI	968
Резюме.....	973
Глава 38. Технология Web Broker	974
Обзор технологии Web Broker.....	975
Структура серверного Web-приложения на основе технологии Web Broker.....	976

Создание серверного Web-приложения.....	978
Диспетчеризация запросов. Глобальный объект приложения.....	979
Web-модули и действия.....	979
Компоненты-продюсеры.....	982
Страницы HTML в серверных Web-приложениях.....	984
Отображение данных.....	985
Ввод и редактирование данных.....	987
Cookies.....	991
Использование баз данных.....	993
Публикация записей таблиц баз данных.....	993
Генерация отчетов.....	995
Редактирование данных.....	1000
Пример простого серверного Web-приложения.....	1002
Главное окно приложения.....	1002
Просмотр таблицы Country.....	1003
Тестирование приложения.....	1006
Резюме.....	1007
Глава 39. Протокол SOAP и Web-службы. Клиентская часть.....	1008
Почему SOAP?.....	1008
Delphi и <i>TSomeConnection</i>	1009
SOAP — решение проблем.....	1010
Web-службы.....	1012
Описание протокола WSDL.....	1013
Архитектура Web-служб в Delphi.....	1019
Клиент Web-службы.....	1019
Генерация интерфейса Web-службы.....	1020
O, RIO, RIO.....	1024
Как связаться с нужной службой.....	1026
Решение коммуникационных проблем.....	1028
Что такое <i>InvokeRegistry</i> . Регистрация интерфейсов и типов данных.....	1031
Резюме.....	1034
Глава 40. Серверная часть Web-службы.....	1035
Назначение и настройки компонентов серверной части.....	1038
Компонент <i>TWSDLHTMLPublish</i>	1039
Компоненты <i>THTTPSopDispatcher</i> и <i>THTTPSopPascalInvoker</i>	1042
Клиент службы <i>SimpleEcho</i>	1043
Обработка исключительных ситуаций.....	1043
Средства разработки для SOAP: подход Microsoft.....	1046
Краткий обзор архитектуры Web-служб в SOAP Toolkit.....	1047
Сценарий клиентской части.....	1049
Создание COM-объекта <i>SimpleEchoCOM</i> и публикация Web-службы на его основе.....	1051
Использование утилиты SOAP Trace.....	1054
Резюме.....	1056

Глава 41. Технология WebSnap.....	1057
Структура приложения WebSnap.....	1058
Обработка запросов.....	1060
Web-модули.....	1060
Модуль данных.....	1061
Модуль страницы.....	1062
Модуль приложения.....	1063
Компоненты уровня приложения.....	1064
Управление компонентами приложения.....	1064
Адаптер и глобальная переменная приложения.....	1066
Информация о пользователях.....	1067
Сессии.....	1070
Управление файлами приложения.....	1071
Компоненты-диспетчеры.....	1071
Компоненты-продюсеры.....	1073
Компоненты-адаптеры.....	1074
Создание приложения WebSnap.....	1077
Конструирование интерфейса и управление данными.....	1081
Использование скриптов.....	1082
Навигация по страницам приложения.....	1083
Использование полей и действий компонентов-адаптеров.....	1084
Взаимодействие с базами данных.....	1087
Специализированные элементы управления.....	1087
Просмотр данных.....	1089
Редактирование данных.....	1091
Обработка ошибок.....	1091
Аутентификация пользователей.....	1093
Списки пользователей и сессии.....	1093
Создание страницы аутентификации.....	1094
Настройка приложения.....	1096
Использование XML и XSL.....	1096
Использование файла XML и шаблона XSL.....	1097
Использование набора данных и шаблона XSL.....	1099
Дополнительный компонент <i>TAdapterXMLBuilder</i>	1101
Резюме.....	1102
Приложение. Описание компакт-диска.....	1103
Предметный указатель.....	1105



Глава 2

Типы данных в Delphi

В этой главе рассматриваются основные типы данных, применяемые в Delphi, а также некоторые часто используемые конструкции языка. Знание основных типов данных облегчает программирование и позволяет избежать ряда ошибок начинающим программистам.

Все существующие в Delphi типы данных можно разделить на две основные группы. К первой относятся стандартные типы, предопределенные в среде разработки. На их основе разработчик может описывать собственные (определяемые) типы данных, которые относятся ко второй группе.

Определяемые типы данных существенно расширяют возможности программистов. Достаточно открыть любой файл с исходным кодом VCL, чтобы убедиться, что этот инструмент широко применяется разработчиками фирмы Borland. Однако при определении собственных типов данных требуется соблюдать осторожность и тщательно продумывать, какой предопределенный тип данных положить в основу собственного. Для этого необходимо иметь максимум информации о стандартных типах данных.

В главе рассматриваются следующие вопросы.

- Типы данных, используемые в Delphi.
- Тип вариант.
- Применение записей.
- Работа с массивами.

Материал главы рассчитан на читателя, имеющего представление о самом языке Delphi, его операторах и основных возможностях.

Простые типы

В эту категорию входят традиционные типы, большинство из которых сопровождают разработчиков чуть ли не с начала компьютерной эры. Сюда входят числовые, логический и символьный типы данных.

Несмотря на привычность и ординарность, в современной реализации простых типов данных имеется ряд особенностей, которые разработчик должен знать. В табл. 2.1 представлено описание простых типов, предназначенных для хранения чисел.

Таблица 2.1. Простые числовые типы данных

Тип данных	Значения	Число байтов в значении
Integer	-2147483648..2147483647	4 (со знаком)
Cardinal	0..4294967295	4
ShortInt	-128..127	1 (со знаком)
SmallInt	-32768..32767	2 (со знаком)
LongInt	-2147483648..2147483647	4 (со знаком)
Int64	$-2^{63}..2^{63}-1$	8 (со знаком)
Byte	0..255	1
Word	0..65535	2
LongWord	0..4294967295	4
Real	$5.0 \times 10^{-324} .. 1.7 \times 10^{308}$	8 (со знаком)
Real48	$2.9 \times 10^{-39} .. 1.7 \times 10^{38}$	6 (со знаком)
Single	$1.5 \times 10^{-45} .. 3.4 \times 10^{38}$	4 (со знаком)
Double	$5.0 \times 10^{-324} .. 1.7 \times 10^{308}$	8 (со знаком)
Extended	$3.6 \times 10^{-4951} .. 1.1 \times 10^{4932}$	10 (со знаком)
Comp	$-2^{63}+1 .. 2^{63}-1$	8 (со знаком)
Currency	-922337203685477.5808.. 922337203685477.5807	8 (со знаком)

Для хранения целочисленных значений рекомендуется использовать базовые типы Integer (для положительных и отрицательных чисел) и Cardinal (для положительных чисел, вдвое больших Integer), так как они оптимизированы для выполнения вычислительных операций процессором.

Тип данных с плавающей точкой в языке Pascal появился, еще когда наличие сопроцессора (скажем, 80287) было скорее исключением, чем правилом. Все операции выполнялись в режиме эмуляции, для этого был выбран 6-байтовый (48-разрядный) формат, названный Real.

Сейчас процессоров без сопроцессора не бывает. Они работают со стандартизованными Американским институтом инженеров электротехники и электроники (IEEE) форматами: 80-разрядным (в Delphi — `Extended`), 64-разрядным (`Double`) и 32-разрядным (`Single`). Соответственно, старый добрый `Real` стал настоящим рудиментом, пожирающим массу времени на преобразование чисел.

На самом деле в Delphi под именем `Real` скрывается тип `Double`, что позволяет намного ускорить вычисления. Поэтому здесь необходимо учитывать следующие тонкости. Во-первых, размер всех массивов и структур с `Real` увеличится на 25%, что в среде Win32 не так уж страшно. Во-вторых, не работает код, в явном виде ориентированный на длину `Real` (6 байт). В этом случае можно описать переменные с новым типом `Real48`:

```
var r:real; r48:real48;
...
  ShowMessage(IntToStr(SizeOf(r)));
  ShowMessage(IntToStr(SizeOf(r48)));
...
```

В первом случае будет выдан результат 8, во втором — прежние 6.

Можно использовать директиву компилятора `{$REALCOMPATIBILITY ON}`, которая отменяет замену `Real` на `Double`. В этом случае описанный выше пример два раза выдаст ответ 6.

Для символьных данных (значение переменной данного типа хранит один символ) базовым является тип `Char`. На его основе созданы два дополнительных типа — `AnsiChar` и `WideChar`. Они обеспечивают работу с двумя кодировками символов: `ANSI` и `Unicode`.

Примечание

Кодировка `ANSI` отводит для хранения символа один байт. Кодировка `Unicode` является более сложной. В ней символы могут быть представлены одним или двумя байтами.

В Delphi расширены возможности логического типа данных. Помимо базового типа `Boolean` существуют дополнительные типы `ByteBool` (один байт), `WordBool` (два байта), `LongBool` (четыре байта), которые необходимы для обеспечения совместимости с функциями `Windows` и другими средами разработки (`C++`).

В базовом логическом типе все осталось по-прежнему: его возможные значения `True` (1) и `False` (0). В остальных типах значению `False` соответствует 0, всем остальным числам соответствует `True`.

Перечисляемый тип не претерпел никаких изменений. Он представляет собой упорядоченное множество значений:

```
type
    TSomeEnumeration = (seFirst, seSecond, seThrid);
var Enum1: TSomeEnumeration;
    Enum2: (Val1, Val2, Val3, Val4);
...
Enum1 := seSecond;
Enum2 := Val1;
...
```

Последний простой тип — поддиапазон — представляет собой подмножество любого другого базового простого типа:

```
type TOtherEnum = (Win95, NT4, Win98, Win2000, XP, Win 2003, LongHorn);
    TSub = 0..127;
var Sub1: TSub;
    Sub2: Win98..Win2000;
    Sub3: 'a'..'z';
...
Sub1 := 100;
Sub2 := NT4;
Sub3 := 'g';
...
```

Строковый тип

Самые разнообразные тексты встречаются в любых приложениях. Для их хранения можно применять константы и переменные. Для представления текстовых значений в Delphi есть три типа.

- Тип `ShortString` — может представлять текст длиной до 255 символов. Переменная этого типа имеет фиксированный размер 256 байтов. Первый байт отводится для хранения длины текста в символах. Остальные байты отведены для хранения символов.
- `AnsiString` — предназначен для создания переменных для хранения текстов произвольной длины. Тексты состоят из символов в кодировке ANSI (на один символ отводится один байт). Память для переменных этого типа выделяется динамически и ограничена только доступным адресным пространством. Сама переменная представляет собой указатель размером 4 байта. Если переменная пуста, указатель равен `Nil`. Две переменные,

имеющие одинаковые значения, ссылаются на одну область памяти (их указатели равны).

- Тип `WideString` — также осуществляет динамическое выделение памяти для хранения текста. Текст должен содержать символы в кодировке Unicode (два байта на символ).

Тип `ShortString` оставлен для обеспечения обратной совместимости с более ранними версиями. По умолчанию в проекте устанавливается директива компилятора `{$H+}`, которая определяет, что объявление

```
var SomeStr: String;
```

соответствует типу `AnsiString`. Директива `{$H-}` определяет, что такое объявление соответствует типу `ShortString`. Впрочем, можно употреблять типы `ShortString` и `AnsiString` напрямую.

Длину текста для строковой переменной можно ограничить числом символов, меньшим 255:

```
var SomeString[64];
```

К отдельному символу текста можно обратиться по его порядковому номеру (индексу):

```
var S: Char;
...
S := SomeString[4];
```

Помните, что для типа `ShortString` нулевой индекс соответствует одному байту длины текста.

Структурные типы

Структурные типы отличаются от простых тем, что переменная такого типа может содержать несколько значений простого типа одновременно. В зависимости от конкретного структурного типа это могут быть значения одного простого типа (множество, массив) или нескольких (запись).

Все структурные типы, кроме набора, могут объединять значения не только простых, но и структурных типов.

Данные в переменной структурного типа по умолчанию выравниваются по границе слова или двойного слова. Это сделано для ускорения доступа к данным.

Если переменная занимает много места, то быстротой можно пожертвовать ради экономии места — значения можно хранить упакованными. Для этого предназначено ключевое слово `packed`:

```
type TSomeStruct = packed set of 1..100;
```


Множества

Множество представляет собой набор значений одного из простых типов (кроме вещественного). Переменную-множество можно объявить через тип или напрямую:

```
type TSomeSet = set of Byte;
      TNumSet = 0..100;
var SetVar1: TSomeSet;
    SetVar2: set of 'A'..'Z';
```

В программе множеству можно присваивать произвольное подмножество:

```
SetVar1 := [1, 3, 5, 7, 11];
SetVar2 := ['X', 'Y', 'Z'];
```

Для контроля значений множеств имеется специальный оператор `in`:

```
if SetVar1 in [1, 3, 5]
then ShowMessage('Простое число');
```

Массивы

В распоряжении программистов в Delphi есть массивы нескольких видов. Массивы могут быть открытыми, их размер теперь можно задавать динамически, также доступны многомерные динамические массивы. Однако обо всем по порядку.

Статические массивы

В Delphi нельзя при описании массива опустить ни нижнюю, ни верхнюю границы. Как правило, нумерация массивов начинается с нуля. Но если уж очень хочется начать нумерацию с единицы, теперь нужно написать:

```
var A: array[1..5] of integer;
```

При передаче массивов в качестве параметров процедур нельзя явно указывать границы. Следующее объявление функции

```
function atest(x: array[1..5] of integer): integer;
```

будет отвергнуто компилятором. Допускается только "безграничный" вариант:

```
function atest(x: array of integer): integer;
```

В Delphi существует строгая типизация. Так, в случае

```
Var
  A1: array[1..5] of integer;
  A2: array[1..5] of integer;
```

переменные `A1` и `A2` будут иметь разный тип! Естественно, массив `A1` и формальный параметр функции `Atest` тоже будут иметь разный тип. Чтобы не нарушать канонов языка, массивы в функции передаются как динамические, т. е. вместе с адресом массива передается и число элементов в нем. Внутри тела функции можно получить значения граничных индексов с помощью функций `Low()` и `High()`:

```
function atest(x: array of integer): integer;
var i: Integer;
begin
    Result := 0;
    for i:=Low(x) to High(x) do
        Result := Result + x[i];
    end;
```

Есть и другой способ не отступить от канонов и остаться в рамках статических массивов — типизировать их:

```
type
    TMyArray = array[1..5] of integer;
Var
    A1: TMyArray;
function atest2(x: TMyArray): integer;
```

В этом случае компилятор не передает вызываемой функции длину массива — она ведь известна заранее. Это очень важно, если вы вызываете из программ на Delphi функции, написанные на других языках, — C, Fortran и т. д. Там "лишнее" двойное слово, содержащее длину массива, нарушит правильность работы. Поэтому присваивайте всем передаваемым массивам определенный тип.

Примечание

Обычно массивы передаются по ссылке, поэтому перед описанием таких параметров в функции следует указать ключевое слово `var`. Передача массивов, особенно больших, по значению (т. е. без ключевого слова `var`) может создать проблемы с памятью. Если вы боитесь употреблять тип `var`, а значения элементов не должны изменяться, опишите этот параметр с ключевым словом `const`.

Кстати, нумерация элементов массива не с 0 настолько несовременна, что может привести к курьезной ситуации. Передадим массив типа `TMyArray` в функцию, рассчитанную на динамический массив, и покажем значения границ:

```
function atest(x: array of integer): integer;
begin
```