

Contents

Chapter 1

Introduction 1-1

What's in this manual?	1-1
Manual conventions	1-2
Contacting developer support	1-3

Part I

Programming with C++Builder

Chapter 2

Programming with C++Builder 2-1

The integrated development environment	2-1
Designing applications	2-2
Understanding the VCL	2-2
Properties	2-2
Methods	2-2
Events	2-3
User events	2-3
System events	2-3
Objects, components, and controls in the VCL	2-3
The TObject branch	2-4
The TPersistent branch	2-5
The TComponent branch	2-5
The TControl branch	2-6
The TWinControl branch	2-7
Properties common to TControl	2-7
Action properties	2-8
Position, size, and alignment properties	2-8
Display properties	2-8
Parent properties	2-9
A navigation property	2-9
Drag-and-drop properties	2-9
Drag-and-dock properties	2-9
Standard events common to TControl	2-10
Properties common to TWinControl	2-10
General information properties	2-11
Border style display properties	2-11
Navigation properties	2-11
Drag-and-dock properties	2-12
Events common to TWinControl	2-12
Creating the application user interface	2-12
Using components	2-13

VCL standard components	2-14
Text controls	2-15
Specialized input controls	2-16
Buttons and similar controls	2-17
Button controls	2-17
Bitmap buttons	2-18
Speed buttons	2-18
Check boxes	2-18
Radio buttons	2-18
Toolbars	2-18
Cool bars	2-19
Handling lists	2-19
List boxes and check-list boxes	2-19
Combo boxes	2-20
Tree views	2-20
List views	2-21
Date-time pickers and month calendars	2-21
Grouping components	2-21
Group boxes and radio groups	2-21
Panels	2-22
Scroll boxes	2-22
Tab controls	2-22
Page controls	2-22
Header controls	2-22
Visual feedback	2-22
Labels and static-text components	2-23
Status bars	2-23
Progress bars	2-23
Help and hint properties	2-23
Grids	2-24
Draw grids	2-24
String grids	2-24
Graphics display	2-24
Images	2-25
Shapes	2-25
Bevels	2-25
Paint boxes	2-25
Animation control	2-25
Windows common dialog boxes	2-26
Using windows common dialog boxes	2-26
Using helper objects	2-26
Working with lists	2-27

Working with string lists	2-27
Loading and saving string lists	2-27
Creating a new string list	2-28
Manipulating strings in a list	2-29
Associating objects with a string list	2-31
Windows registry and INI files	2-31
Using TINIFile	2-32
Using TRegistry	2-33
Using TRegIniFile	2-33
Using TCanvas	2-34
Using TPrinter.	2-34
Using streams.	2-35
Developing applications	2-35
Editing code.	2-35
Debugging applications	2-35
Deploying applications	2-36

Chapter 3

Building applications, components, and libraries **3-1**

Creating applications	3-1
Windows applications	3-1
User interface models	3-2
Setting IDE, project, and compilation options	3-2
Programming templates	3-3
Console applications	3-3
Using the VCL in console applications	3-3
Service applications	3-4
Service threads	3-6
Service name properties	3-8
Debugging services.	3-8
Creating packages and DLLs	3-9
When to use packages and DLLs	3-9
Using DLLs in C++Builder	3-10
Creating DLLs in C++Builder	3-10
Creating DLLs containing VCL components	3-11
Linking DLLs	3-14
Writing database applications	3-14
Building distributed applications	3-15
Distributing applications using TCP/IP	3-15
Using sockets in applications	3-15
Creating Web server applications	3-16
Distributing applications using COM and DCOM	3-16
COM and DCOM	3-16
MTS and COM+	3-17
Distributing applications using CORBA.	3-17

Distributing database applications	3-17
Using data modules and remote data modules	3-17
Creating and editing data modules	3-18
Creating business rules in a data module	3-18
Accessing a data module from a form	3-19
Adding a remote data module to an application server project.	3-19
Using the Object Repository	3-19
Sharing items within a project	3-19
Adding items to the Object Repository	3-20
Sharing objects in a team environment	3-20
Using an Object Repository item in a project.	3-20
Copying an item	3-20
Inheriting an item	3-21
Using an item.	3-21
Using project templates.	3-21
Modifying shared items	3-21
Specifying a default project, new form, and main form	3-22

Chapter 4

Developing the application user interface **4-1**

Understanding TApplication, TScreen, and TForm	4-1
Using the main form	4-1
Adding additional forms	4-2
Linking forms	4-2
Hiding the main form.	4-2
Working at the application level.	4-3
Handling the screen.	4-3
Managing layout	4-3
Working with messages	4-4
More details on forms	4-5
Controlling when forms reside in memory.	4-5
Displaying an auto-created form.	4-5
Creating forms dynamically	4-5
Creating modeless forms such as windows	4-6
Using a local variable to create a form instance	4-7
Passing additional arguments to forms.	4-7
Retrieving data from forms.	4-8
Retrieving data from modeless forms	4-8
Retrieving data from modal forms.	4-10

Reusing components and groups of components	4-12
Creating and using component templates	4-12
Working with frames	4-13
Creating frames.	4-13
Adding frames to the Component palette	4-14
Using and modifying frames	4-14
Sharing frames	4-15
Creating and managing menus.	4-15
Opening the Menu Designer	4-16
Building menus.	4-17
Naming menus	4-18
Naming the menu items	4-18
Adding, inserting, and deleting menu items	4-19
Creating submenus.	4-20
Adding images to menu items	4-22
Viewing the menu	4-22
Editing menu items in the Object Inspector.	4-22
Using the Menu Designer context menu.	4-23
Commands on the context menu	4-23
Switching between menus at design time.	4-24
Using menu templates.	4-24
Saving a menu as a template	4-25
Naming conventions for template menu items and event handlers	4-26
Manipulating menu items at runtime	4-27
Merging menus.	4-27
Specifying the active menu: Menu property	4-27
Determining the order of merged menu items: GroupIndex property	4-27
Importing resource files	4-28
Designing toolbars and cool bars	4-28
Adding a toolbar using a panel component.	4-29
Adding a speed button to a panel.	4-30
Assigning a speed button's glyph.	4-30
Setting the initial condition of a speed button.	4-30
Creating a group of speed buttons	4-31
Allowing toggle buttons	4-31
Adding a toolbar using the toolbar component.	4-31
Adding a tool button	4-32
Assigning images to tool buttons	4-32

Setting tool button appearance and initial conditions	4-33
Creating groups of tool buttons	4-33
Allowing toggled tool buttons	4-33
Adding a cool bar component	4-34
Setting the appearance of the cool bar	4-34
Responding to clicks	4-34
Assigning a menu to a tool button.	4-35
Adding hidden toolbars	4-35
Hiding and showing toolbars	4-35
Using action lists	4-36
Action objects	4-36
Using Actions	4-37
Centralizing code	4-38
Linking properties	4-38
Executing actions	4-38
Updating actions.	4-40
Pre-defined action classes	4-41
Standard edit actions	4-41
Standard Window actions.	4-41
Standard Help actions.	4-42
DataSet actions.	4-42
Writing action components.	4-43
How actions find their targets	4-43
Registering actions.	4-44
Writing action list editors	4-45

Chapter 5	
Working with controls	5-1
Implementing drag-and-drop in controls	5-1
Starting a drag operation.	5-1
Accepting dragged items.	5-2
Dropping items	5-2
Ending a drag operation	5-3
Customizing drag and drop with a drag object	5-3
Changing the drag mouse pointer.	5-4
Implementing drag-and-dock in controls	5-4
Making a windowed control a docking site.	5-4
Making a control a dockable child.	5-4
Controlling how child controls are docked	5-5
Controlling how child controls are undocked.	5-6
Controlling how child controls respond to drag-and-dock operations	5-6

Working with text in controls.	5-6
Setting text alignment	5-6
Adding scroll bars at runtime.	5-7
Adding the Clipboard object	5-7
Selecting text	5-8
Selecting all text	5-8
Cutting, copying, and pasting text	5-8
Deleting selected text	5-9
Disabling menu items	5-9
Providing a pop-up menu.	5-10
Handling the OnPopup event.	5-10
Adding graphics to controls	5-11
Indicating that a control is owner-drawn	5-11
Adding graphical objects to a string list	5-12
Adding images to an application	5-12
Adding images to a string list	5-12
Drawing owner-drawn items	5-13
Sizing owner-drawn items	5-14
Drawing each owner-drawn item	5-14

Chapter 6

Working with graphics and multimedia

6-1

Overview of graphics programming.	6-1
Refreshing the screen	6-2
Types of graphic objects	6-2
Common properties and methods of Canvas	6-3
Using the properties of the Canvas object	6-4
Using pens.	6-5
Using brushes	6-7
Reading and setting pixels.	6-9
Using Canvas methods to draw graphic objects	6-9
Drawing lines and polylines.	6-9
Drawing shapes.	6-10
Handling multiple drawing objects in your application	6-11
Keeping track of which drawing tool to use	6-12
Changing the tool with speed buttons	6-12
Using drawing tools	6-13
Drawing on a graphic	6-16
Making scrollable graphics	6-16
Adding an image control	6-16
Loading and saving graphics files	6-18
Loading a picture from a file	6-18

Saving a picture to a file.	6-19
Replacing the picture	6-19
Using the Clipboard with graphics	6-20
Copying graphics to the Clipboard	6-21
Cutting graphics to the Clipboard	6-21
Pasting graphics from the Clipboard	6-21
Rubber banding example.	6-22
Responding to the mouse.	6-22
Adding a field to a form object to track mouse actions	6-25
Refining line drawing	6-26
Working with multimedia	6-28
Adding silent video clips to an application	6-28
Example of adding silent video clips	6-29
Adding audio and/or video clips to an application	6-30
Example of adding audio and/or video clips	6-32

Chapter 7

Writing multi-threaded applications

7-1

Defining thread objects.	7-1
Initializing the thread.	7-2
Assigning a default priority	7-2
Indicating when threads are freed	7-3
Writing the thread function	7-3
Using the main VCL thread.	7-4
Using thread-local variables	7-5
Checking for termination by other threads	7-5
Writing clean-up code.	7-6
Coordinating threads.	7-6
Avoiding simultaneous access	7-6
Locking objects.	7-6
Using critical sections	7-7
Using the multi-read exclusive-write synchronizer	7-7
Other techniques for sharing memory.	7-8
Waiting for other threads.	7-8
Waiting for a thread to finish executing	7-8
Waiting for a task to be completed.	7-9
Executing thread objects	7-10
Overriding the default priority	7-10
Starting and stopping threads	7-11
Debugging multi-threaded applications	7-11

Chapter 8

Exception handling 8-1

C++ exception handling	8-1
ANSI requirements for exception handling	8-1
Exception handling syntax	8-2
Exception declarations	8-2
Throwing an exception	8-3
Examples	8-3
Handling an exception	8-6
Exception specifications	8-9
Constructors and destructors in exception handling	8-10
Unhandled exceptions	8-10
Setting exception handling options	8-11
Structured exceptions under Win32	8-11
Syntax of structured exceptions	8-12
Handling structured exceptions	8-12
Exception filters	8-13
Mixing C++ with structured exceptions	8-15
C-based exceptions in C++ program example	8-16
Defining exceptions	8-17
Raising exceptions	8-17
Termination blocks	8-18
VCL exception handling	8-19
Differences between C++ and VCL exception handling	8-20
Handling operating system exceptions	8-20
Handling VCL exceptions	8-21
VCL exception classes	8-21
Portability considerations	8-22

Chapter 9

C++ language support for the VCL 9-1

C++ and Object Pascal object models	9-1
Object identity and instantiation	9-1
Distinguishing C++ and Object Pascal references	9-2
Copying objects	9-2
Objects as function arguments	9-3
Object construction for C++Builder VCL classes	9-4
C++ object construction	9-4
Object Pascal object construction	9-4
C++Builder object construction	9-4

Calling virtual methods in base class

constructors	9-6
Object Pascal model	9-6
C++ model	9-7
C++Builder model	9-7
Example: calling virtual methods	9-7
Constructor initialization of data members for virtual functions	9-8
Object destruction	9-9
Exceptions thrown from constructors	9-10
Virtual methods called from destructors	9-11
AfterConstruction and BeforeDestruction	9-11
Class virtual functions	9-11
Support for Object Pascal data types and language concepts	9-11
Typedefs	9-12
Classes that support the Object Pascal language	9-12
C++ language counterparts to the Object Pascal language	9-12
Var parameters	9-12
Untyped parameters	9-13
Open arrays	9-13
Calculating the number of elements	9-13
Temporaries	9-14
array of const	9-14
OPENARRAY macro	9-15
EXISTINGARRAY macro	9-15
C++ functions that take open array arguments	9-15
Types defined differently	9-15
Boolean data types	9-16
Char data types	9-16
Resource strings	9-16
Default parameters	9-17
Runtime type information	9-18
Unmapped types	9-18
6-byte Real types	9-18
Arrays as return types of functions	9-19
Keyword extensions	9-19
__classid	9-19
__closure	9-19
__property	9-20
__published	9-21

The __declspec keyword extension.	9-21
__declspec(delphiclass)	9-21
__declspec(delphireturn).	9-22
__declspec(dynamic).	9-22
__declspec(hidesbase)	9-22
__declspec(package)	9-23
__declspec(pascalimplementation)	9-23
 Chapter 10	
Working with packages and components	10-1
Why use packages?	10-2
Packages and standard DLLs	10-2
Runtime packages	10-2
Using packages in an application.	10-2
Dynamically loading packages	10-3
Deciding which runtime packages to use	10-3
Custom packages.	10-4
Design-time packages	10-4
Installing component packages.	10-5
Creating and editing packages	10-6
Creating a package	10-6
Editing an existing package	10-7
Package source files and project option files.	10-7
Packaging components.	10-8
Understanding the structure of a package	10-9
Naming packages.	10-9
The Requires list	10-9
The Contains list	10-10
Compiling packages	10-10
Package-specific compiler directives.	10-10
Using the command-line compiler and linker	10-12
Package files created by a successful compilation	10-12
Deploying packages	10-13
Deploying applications that use packages.	10-13
Distributing packages to other developers.	10-13
Package collection files	10-13

Chapter 11	
Creating international applications	11-1
Internationalization and localization	11-1
Internationalization	11-1
Localization	11-1
Internationalizing applications	11-2
Enabling application code	11-2
Character sets	11-2
OEM and ANSI character sets	11-2
Double byte character sets	11-2
Wide characters	11-3
Including bi-directional functionality in applications	11-3
BiDiMode property	11-5
Locale-specific features	11-7
Designing the user interface	11-8
Text	11-8
Graphic images	11-8
Formats and sort order	11-8
Keyboard mappings.	11-9
Isolating resources.	11-9
Creating resource DLLs.	11-9
Using resource DLLs	11-10
Dynamic switching of resource DLLs.	11-11
Localizing applications.	11-12
Localizing resources.	11-12

Chapter 12	
Deploying applications	12-1
Deploying general applications	12-1
Using installation programs	12-2
Identifying application files	12-2
Application files	12-2
Package files	12-3
ActiveX controls	12-3
Helper applications	12-3
DLL locations.	12-3
Deploying database applications	12-4
Providing the database engine.	12-4
Borland Database Engine	12-4
Third-party database engines	12-5
SQL Links.	12-5

Multi-tiered Distributed Application Services (MIDAS)	12-6
Deploying Web applications	12-6
Programming for varying host environments	12-6
Screen resolutions and color depths	12-7
Considerations when not dynamically resizing	12-7
Considerations when dynamically resizing forms and controls	12-7
Accommodating varying color depths	12-8
Fonts	12-9
Windows versions	12-9
Software license requirements	12-10
DEPLOY.TXT	12-10
README.TXT	12-10
No-nonsense license agreement	12-10
Third-party product documentation	12-10

Part II

Developing database applications

Chapter 13

Designing database applications 13-1

Using databases	13-1
Types of databases	13-2
Local databases	13-2
Remote database servers	13-2
Database security	13-3
Transactions	13-3
Data Dictionary	13-4
Referential integrity, stored procedures, and triggers	13-5
Database architecture	13-6
Planning for scalability	13-7
Single-tiered database applications	13-8
Two-tiered database applications	13-9
Multi-tiered database applications	13-9
Designing the user interface	13-11
Displaying a single record	13-11
Displaying multiple records	13-12
Analyzing data	13-12
Selecting what data to show	13-13
Writing reports	13-15

Chapter 14

Building one- and two-tiered applications 14-1

BDE-based applications	14-2
BDE-based architecture	14-2
Understanding databases and datasets	14-3
Using sessions	14-3
Connecting to databases	14-4
Using transactions	14-5
Explicitly controlling transactions	14-5
Using a database component for transactions	14-6
Using the TransIsolation property	14-7
Using passthrough SQL	14-8
Using local transactions	14-8
Caching updates	14-9
Creating and restructuring database tables	14-10
ADO-based applications	14-10
ADO-based architecture	14-10
Understanding ADO databases and datasets	14-11
Connecting to ADO databases	14-11
Retrieving data	14-12
Creating and restructuring ADO database tables	14-12
Flat-file database applications	14-13
Creating the datasets	14-14
Creating a new dataset using persistent fields	14-14
Creating a dataset using field and index definitions	14-14
Creating a dataset based on an existing table	14-15
Loading and saving data	14-16
Using the briefcase model	14-16
Scaling up to a three-tiered application	14-17

Chapter 15

Creating multi-tiered applications 15-1

Advantages of the multi-tiered database model	15-2
Understanding MIDAS technology	15-2
Overview of a MIDAS-based multi-tiered application	15-3

The structure of the client application	15-4
The structure of the application server. . . .	15-4
Using transactional data modules	15-5
Pooling remote data modules	15-7
Using the IAppServer interface	15-8
Choosing a connection protocol	15-9
Using DCOM connections	15-9
Using Socket connections	15-9
Using Web connections.	15-10
Building a multi-tiered application	15-11
Creating the application server.	15-11
Setting up the remote data module. . . .	15-13
Configuring the remote data module when it is not transactional.	15-13
Configuring a transactional remote data module	15-14
Creating a data provider for the application server.	15-15
Extending the application server's interface	15-15
Adding callbacks to the application server's interface	15-16
Extending a transactional application server's interface	15-16
Creating the client application	15-16
Connecting to the application server. . . .	15-17
Specifying a connection using DCOM	15-18
Specifying a connection using sockets	15-18
Specifying a connection using HTTP	15-19
Brokering connections	15-19
Managing server connections.	15-20
Connecting to the server	15-20
Dropping or changing a server connection	15-20
Calling server interfaces.	15-21
Managing transactions in multi-tiered applications.	15-21
Supporting master/detail relationships. . . .	15-22
Supporting state information in remote data modules	15-23
Writing MIDAS Web applications	15-24
Distributing a client application as an ActiveX control	15-26
Creating an Active Form for the client application	15-26

Building Web applications using InternetExpress	15-27
Building an InternetExpress application	15-27
Using the javascript libraries	15-28
Granting permission to access and launch the application server.	15-29
Using an XML broker	15-30
Fetching XML data packets.	15-30
Applying updates from XML delta packets	15-31
Creating Web pages with a MIDAS page producer	15-32
Using the Web page editor	15-32
Setting Web item properties	15-33
Customizing the MIDAS page producer template	15-34

Chapter 16

Using provider components **16-1**

Determining the source of data	16-1
Choosing how to apply updates	16-2
Controlling what information is included in data packets.	16-2
Specifying what fields appear in data packets	16-2
Setting options that influence the data packets	16-3
Adding custom information to data packets	16-4
Responding to client data requests	16-5
Responding to client update requests	16-6
Editing delta packets before updating the database	16-6
Influencing how updates are applied	16-7
Screening individual updates	16-9
Resolving update errors on the provider . .	16-9
Applying updates to datasets that do not represent a single table	16-9
Responding to client-generated events. . . .	16-10
Handling server constraints	16-10

Chapter 17

Managing database sessions **17-1**

Working with a session component.	17-1
Using the default session	17-2
Creating additional sessions	17-3
Naming a session	17-4
Activating a session	17-4

Customizing session start-up	17-5
Specifying default database connection behavior	17-6
Creating, opening, and closing database connections	17-6
Closing a single database connection.	17-7
Closing all database connections	17-7
Dropping temporary database connections	17-7
Searching for a database connection	17-8
Retrieving information about a session	17-8
Working with BDE aliases.	17-9
Specifying alias visibility.	17-10
Making session aliases visible to other sessions and applications	17-10
Determining known aliases, drivers, and parameters	17-10
Creating, modifying, and deleting aliases	17-10
Iterating through a session's database components	17-12
Specifying Paradox directory locations	17-12
Specifying the control file location	17-13
Specifying a temporary files location.	17-13
Working with password-protected Paradox and dBASE tables	17-13
Using the AddPassword method	17-13
Using the RemovePassword and RemoveAllPasswords methods	17-14
Using the GetPassword method and OnPassword event.	17-14
Managing multiple sessions	17-16
Using a session component in data modules	17-17

Chapter 18 Connecting to databases **18-1**

Understanding persistent and temporary database components.	18-1
Using temporary database components	18-2
Creating database components at design time	18-2
Creating database components at runtime	18-3
Controlling connections.	18-4
Associating a database component with a session	18-4
Specifying a BDE alias	18-4

Setting BDE alias parameters	18-5
Controlling server login	18-6
Connecting to a database server.	18-7
Special considerations when connecting to a remote server	18-8
Working with network protocols.	18-8
Using ODBC	18-8
Disconnecting from a database server	18-9
Closing datasets without disconnecting from a server	18-9
Iterating through a database component's datasets	18-9
Understanding database and session component interactions.	18-9
Using database components in data modules	18-10
Executing SQL statements from a TDatabase component	18-10
Executing SQL statements from TDatabase	18-10
Executing parameterized SQL statements	18-11

Chapter 19 Understanding datasets **19-1**

What is TDataSet?.	19-2
Types of datasets	19-2
Opening and closing datasets	19-3
Determining and setting dataset states.	19-3
Inactivating a dataset	19-5
Browsing a dataset	19-6
Enabling dataset editing	19-7
Enabling insertion of new records.	19-7
Enabling index-based searches and ranges on tables	19-8
Calculating fields	19-8
Filtering records	19-9
Updating records	19-9
Navigating datasets.	19-9
Using the First and Last methods	19-10
Using the Next and Prior methods	19-10
Using the MoveBy method.	19-11
Using the Eof and Bof properties	19-11
Eof.	19-11
Bof.	19-12
Marking and returning to records.	19-13
Searching datasets	19-15
Using Locate	19-15
Using Lookup	19-16

Displaying and editing a subset of data	
using filters	19-17
Enabling and disabling filtering	19-17
Creating filters	19-17
Setting the Filter property	19-18
Writing an OnFilterRecord event handler.	19-19
Switching filter event handlers	
at runtime	19-19
Setting filter options	19-19
Navigating records in a filtered dataset . .	19-20
Modifying data.	19-21
Editing records	19-21
Adding new records	19-22
Inserting records	19-23
Appending records	19-23
Deleting records	19-23
Posting data to the database	19-23
Canceling changes	19-24
Modifying entire records	19-24
Using dataset events.	19-26
Aborting a method	19-26
Using OnCalcFields	19-26
Using BDE-enabled datasets	19-27
Overview of BDE-enablement	19-28
Handling database and session connections	19-28
Using the DatabaseName and SessionName properties	19-29
Working with BDE handle properties	19-29
Using cached updates	19-29
Caching BLOBs	19-30

Chapter 20

Working with field components	20-1
Understanding field components	20-2
Dynamic field components	20-3
Persistent field components.	20-4
Creating persistent fields	20-5
Arranging persistent fields	20-6
Defining new persistent fields	20-6
Defining a data field	20-7
Defining a calculated field.	20-8
Programming a calculated field	20-9
Defining a lookup field	20-9

Defining an aggregate field	20-11
Deleting persistent field components	20-12
Setting persistent field properties and events	20-12
Setting display and edit properties at design time	20-12
Setting field component properties at runtime	20-14
Creating attribute sets for field components	20-14
Associating attribute sets with field components	20-15
Removing attribute associations.	20-15
Controlling and masking user input . . .	20-15
Using default formatting for numeric, date, and time fields.	20-16
Handling events	20-17
Working with field component methods at runtime	20-17
Displaying, converting, and accessing field values.	20-18
Displaying field component values in standard controls	20-18
Converting field values.	20-19
Accessing field values with the default dataset property	20-20
Accessing field values with a dataset's Fields property.	20-20
Accessing field values with a dataset's FieldByName method.	20-21
Checking a field's current value.	20-21
Setting a default value for a field	20-21
Working with constraints	20-22
Creating a custom constraint.	20-22
Using server constraints	20-22
Using object fields	20-23
Displaying ADT and array fields	20-24
Working with ADT fields.	20-24
Accessing ADT field values.	20-24
Working with array fields	20-25
Accessing array field values	20-25
Working with dataset fields	20-26
Displaying dataset fields	20-26
Accessing data in a nested dataset. . .	20-26
Working with reference fields	20-27
Displaying reference fields	20-27
Accessing data in a reference field . . .	20-27

Chapter 21

Working with tables **21-1**

Using table components.	21-1
Setting up a table component.	21-2
Specifying a database location	21-2
Specifying a table name	21-3
Specifying the table type for local tables.	21-3
Opening and closing a table.	21-4
Controlling read/write access to a table.	21-4
Searching for records	21-5
Searching for records based on indexed fields	21-5
Executing a search with Goto methods	21-6
Executing a search with Find methods	21-7
Specifying the current record after a successful search	21-7
Searching on partial keys	21-7
Searching on alternate indexes	21-8
Repeating or extending a search	21-8
Sorting records	21-8
Retrieving a list of available indexes with GetIndexNames	21-9
Specifying an index with IndexName	21-9
Specifying a dBASE index file.	21-9
Specifying sort order for SQL tables	21-10
Specifying fields with IndexFieldNames	21-11
Examining the field list for an index	21-11
Working with a subset of data	21-11
Understanding the differences between ranges and filters.	21-11
Creating and applying a new range	21-12
Setting the beginning of a range	21-12
Setting the end of a range	21-13
Setting start- and end-range values.	21-14
Specifying a range based on partial keys.	21-14
Including or excluding records that match boundary values.	21-15
Applying a range	21-15
Canceling a range.	21-15
Modifying a range	21-15
Editing the start of a range.	21-16
Editing the end of a range	21-16
Deleting all records in a table.	21-16
Deleting a table.	21-16
Renaming a table.	21-17

Creating a table	21-17
Importing data from another table	21-19
Using TBatchMove	21-19
Creating a batch move component	21-20
Specifying a batch move mode	21-21
Appending records	21-21
Updating records	21-21
Appending and updating records	21-22
Copying datasets.	21-22
Deleting records	21-22
Mapping data types.	21-22
Executing a batch move.	21-23
Handling batch move errors	21-23
Synchronizing tables linked to the same database table.	21-24
Creating master/detail forms	21-25
Building an example master/detail form	21-25
Working with nested tables	21-26
Setting up a nested table component	21-26

Chapter 22

Working with queries **22-1**

Using queries effectively	22-1
Queries for desktop developers	22-2
Queries for server developers	22-3
What databases can you access with a query component?	22-4
Using a query component: an overview	22-4
Specifying the SQL statement to execute.	22-5
Specifying the SQL property at design time.	22-6
Specifying an SQL statement at runtime	22-7
Setting the SQL property directly	22-7
Loading the SQL property from a file	22-8
Loading the SQL property from string list object.	22-8
Setting parameters	22-8
Supplying parameters at design time.	22-9
Supplying parameters at runtime	22-10
Using a data source to bind parameters	22-10
Executing a query.	22-12
Executing a query at design time	22-12
Executing a query at runtime	22-12
Executing a query that returns a result set.	22-13

Executing a query without a result set	22-13
Preparing a query	22-13
Unpreparing a query to release resources.	22-14
Creating heterogeneous queries	22-14
Improving query performance	22-15
Disabling bi-directional cursors.	22-15
Working with result sets	22-16
Enabling editing of a result set	22-16
Local SQL requirements for a live result set	22-16
Restrictions on live queries	22-17
Remote server SQL requirements for a live result set	22-17
Restrictions on updating a live result set.	22-17
Updating a read-only result set.	22-17

Chapter 23

Working with stored procedures 23-1

When should you use stored procedures?	23-2
Using a stored procedure	23-2
Creating a stored procedure component.	23-3
Creating a stored procedure.	23-4
Preparing and executing a stored procedure	23-5
Using stored procedures that return result sets	23-5
Retrieving a result set with a TQuery.	23-5
Retrieving a result set with a TStoredProc	23-6
Using stored procedures that return data using parameters	23-7
Retrieving individual values with a TQuery.	23-7
Retrieving individual values with a TStoredProc	23-7
Using stored procedures that perform actions on data	23-8
Executing an action stored procedure with a TQuery.	23-8
Executing an action stored procedure with a TStoredProc	23-9
Understanding stored procedure parameters	23-10
Using input parameters	23-10
Using output parameters	23-11
Using input/output parameters	23-11

Using the result parameter	23-12
Accessing parameters at design time	23-12
Setting parameter information at design time	23-13
Creating parameters at runtime	23-14
Binding parameters	23-15
Viewing parameter information at design time.	23-15
Working with Oracle overloaded stored procedures	23-16

Chapter 24

Working with ADO components 24-1

Overview of ADO components	24-1
Connecting to ADO data stores	24-2
Connecting to a data store using TADOConnection	24-3
Using a TADOConnection versus a dataset's ConnectionString	24-3
Specifying the connection.	24-3
Accessing the connection object	24-4
Activating and deactivating the connection	24-4
Determining what a connection component is doing	24-5
Fine-tuning a connection	24-5
Specifying connection attributes	24-5
Controlling timeouts	24-7
Controlling the connection login.	24-7
Listing tables and stored procedures	24-8
Accessing the connection's datasets	24-8
Accessing the connection's commands	24-8
Listing available tables	24-9
Listing available stored procedures	24-10
Working with (connection) transactions.	24-10
Using transaction methods	24-10
Using transaction events	24-10
Using ADO datasets	24-11
Features common to all ADO dataset components	24-11
Modifying data.	24-11
Navigating in a dataset	24-12
Using visual data-aware controls	24-12
Connecting to a data store using ADO dataset components.	24-13
Working with record sets	24-13

Using batch updates	24-14	Copying data from another dataset.	25-12
Loading data from and saving data to files	24-16	Assigning data directly	25-12
Using parameters in commands	24-17	Cloning a client dataset cursor.	25-13
Using TADODataSet	24-18	Using a client dataset with a data provider	25-14
Retrieving a dataset using a command	24-18	Specifying a data provider	25-14
Using TADOTable	24-19	Getting parameters from the application server	25-15
Specifying the table to use	24-19	Passing parameters to the application server	25-15
Using TADOQuery.	24-20	Sending query or stored procedure parameters	25-16
Specifying SQL statements	24-20	Limiting records with parameters . . .	25-16
Executing SQL statements	24-21	Overriding the dataset on the application server	25-17
Using TADOStoredProc	24-21	Requesting data from an application server	25-17
Specifying the stored procedure	24-22	Handling constraints	25-18
Executing the stored procedure	24-23	Handling constraints from the server	25-19
Using parameters with stored procedures	24-23	Adding custom constraints	25-19
Executing commands	24-25	Updating records	25-20
Specifying the command	24-26	Applying updates	25-20
Using the Execute method.	24-26	Reconciling update errors.	25-21
Canceling commands	24-27	Refreshing records.	25-22
Retrieving result sets with commands . .	24-27	Communicating with providers using custom events.	25-23
Handling command parameters	24-28	Using a client dataset with flat-file data . .	25-24
		Creating a new dataset	25-24
		Loading data from a file or stream	25-24
		Merging changes into data	25-25
		Saving data to a file or stream	25-25
Chapter 25		Chapter 26	
Creating and using a client dataset 25-1		Working with cached updates 26-1	
Working with data using a client dataset . . .	25-2	Deciding when to use cached updates	26-1
Navigating data in client datasets	25-2	Using cached updates	26-2
Limiting what records appear.	25-2	Enabling and disabling cached updates . .	26-3
Representing master/detail relationships.	25-3	Fetching records	26-4
Constraining data values	25-3	Applying cached updates	26-4
Making data read-only.	25-4	Applying cached updates with a database component method	26-5
Editing data.	25-4	Applying cached updates with dataset component methods	26-6
Undoing changes	25-5	Applying updates for master/detail tables	26-6
Saving changes	25-5	Canceling pending cached updates	26-7
Sorting and indexing.	25-6	Canceling pending updates and disabling further cached updates . . .	26-8
Adding a new index	25-6		
Deleting and switching indexes.	25-7		
Using indexes to group data.	25-7		
Representing calculated values	25-8		
Using internally calculated fields in client datasets.	25-9		
Using maintained aggregates	25-9		
Specifying aggregates	25-10		
Aggregating over groups of records . .	25-11		
Obtaining aggregate values	25-11		
Adding application-specific information to the data.	25-12		

Canceling pending cached updates.	26-8
Canceling updates to the current record	26-8
Undeleting cached records	26-9
Specifying visible records in the cache	26-9
Checking update status	26-10
Using update objects to update a dataset.	26-11
Specifying the UpdateObject property for a dataset	26-12
Using a single update object.	26-12
Using multiple update objects.	26-12
Creating SQL statements for update components	26-13
Creating SQL statements at design time	26-13
Understanding parameter substitution in update SQL statements	26-14
Composing update SQL statements	26-15
Using an update component's Query property	26-16
Using the DeleteSQL, InsertSQL, and ModifySQL properties	26-17
Executing update statements	26-18
Calling the Apply method	26-18
Calling the SetParams method	26-19
Calling the ExecSQL method	26-19
Using dataset components to update a dataset	26-20
Updating a read-only result set	26-21
Controlling the update process.	26-21
Determining if you need to control the updating process	26-22
Creating an OnUpdateRecord event handler.	26-22
Handling cached update errors	26-23
Referencing the dataset to which to apply updates.	26-24
Indicating the type of update that generated an error	26-24
Specifying the action to take	26-25
Working with error message text	26-26
Accessing a field's OldValue, NewValue, and CurValue properties	26-26

Chapter 27

Using data controls

27-1

Using common data control features	27-1
Associating a data control with a dataset	27-2
Editing and updating data	27-3
Enabling editing in controls on user entry	27-3
Editing data in a control.	27-3
Disabling and enabling data display	27-4
Refreshing data display.	27-5
Enabling mouse, keyboard, and timer events	27-5
Using data sources	27-5
Using TDataSource properties.	27-6
Setting the DataSet property	27-6
Setting the Name property	27-6
Setting the Enabled property	27-7
Setting the AutoEdit property	27-7
Using TDataSource events	27-7
Using the OnDataChange event	27-7
Using the OnUpdateData event	27-7
Using the OnStateChange event	27-7
Controls that represent a single field	27-8
Displaying data as labels	27-8
Displaying and editing fields in an edit box	27-9
Displaying and editing text in a memo control	27-9
Displaying and editing text in a rich edit memo control	27-10
Displaying and editing graphics fields in an image control	27-10
Displaying and editing data in list and combo boxes	27-11
Displaying and editing data in a list box.	27-11
Displaying and editing data in a combo box	27-12
Displaying and editing data in lookup list and combo boxes	27-12
Specifying a list based on a lookup field	27-13
Specifying a list based on a secondary data source.	27-13
Setting lookup list and combo box properties	27-14
Searching incrementally for list item values	27-14

Handling Boolean field values with checkboxes	27-14
Restricting field values with radio controls	27-15
Viewing and editing data with TDBGrid	27-16
Using a grid control in its default state	27-17
Creating a customized grid	27-17
Understanding persistent columns	27-18
Determining the source of a column property at runtime	27-19
Creating persistent columns	27-19
Deleting persistent columns	27-20
Arranging the order of persistent columns	27-20
Defining a lookup list column	27-20
Defining a pick list column	27-21
Putting a button in a column	27-21
Setting column properties at design time	27-21
Restoring default values to a column	27-22
Displaying ADT and array fields	27-23
Setting grid options	27-24
Editing in the grid	27-25
Rearranging column order at design time	27-26
Rearranging column order at runtime	27-26
Controlling grid drawing	27-26
Responding to user actions at runtime.	27-27
Creating a grid that contains other data-aware controls	27-28
Navigating and manipulating records.	27-29
Choosing navigator buttons to display	27-30
Hiding and showing navigator buttons at design time	27-30
Hiding and showing navigator buttons at runtime	27-30
Displaying fly-over help.	27-31
Using a single navigator for multiple datasets	27-31

Chapter 28

Using decision support components

28-1

Overview	28-1
About crosstabs	28-2
One-dimensional crosstabs	28-3

Multidimensional crosstabs	28-3
Guidelines for using decision support components	28-3
Using datasets with decision support components	28-5
Creating decision datasets with TQuery or TTable	28-5
Creating decision datasets with the Decision Query editor	28-6
Using the Decision Query editor.	28-6
Decision query properties	28-7
Using decision cubes	28-7
Decision cube properties and events	28-7
Using the Decision Cube editor	28-8
Viewing and changing dimension settings	28-8
Setting the maximum available dimensions and summaries.	28-9
Viewing and changing design options	28-9
Using decision sources	28-9
Properties and events	28-9
Using decision pivots.	28-10
Decision pivot properties.	28-10
Creating and using decision grids	28-11
Creating decision grids	28-11
Using decision grids	28-11
Opening and closing decision grid fields	28-12
Reorganizing rows and columns in decision grids	28-12
Drilling down for detail in decision grids	28-12
Limiting dimension selection in decision grids.	28-12
Decision grid properties	28-12
Creating and using decision graphs	28-13
Creating decision graphs	28-14
Using decision graphs	28-14
The decision graph display.	28-15
Customizing decision graphs	28-16
Setting decision graph template defaults	28-17
Customizing decision graph series	28-17
Decision support components at runtime	28-18
Decision pivots at runtime	28-19
Decision grids at runtime.	28-19
Decision graphs at runtime.	28-19

Decision support components and
memory control 28-20
Setting maximum dimensions,
summaries, and cells 28-20
Setting dimension state 28-20
Using paged dimensions 28-21

Part III
Writing distributed applications

Chapter 29
Writing CORBA applications 29-1

Overview of a CORBA application 29-1
Understanding stubs and skeletons 29-2
Using Smart Agents 29-3
Activating server applications 29-3
Binding interface calls dynamically 29-4
Writing CORBA servers 29-4
Defining object interfaces 29-5
Using the CORBA Server Wizard. 29-5
Generating stubs and skeletons from
an IDL file 29-6
Using the CORBA Object
Implementation Wizard 29-6
Instantiating CORBA objects 29-7
Using the delegation model 29-8
Viewing and editing changes 29-9
Implementing CORBA Objects 29-9
Guarding against thread conflicts. 29-11
Changing CORBA interfaces 29-12
Registering server interfaces 29-12
Writing CORBA clients 29-13
Using stubs 29-14
Using the dynamic invocation
interface 29-15
Testing CORBA servers 29-16
Setting up the testing tool 29-17
Recording and running test scripts. 29-17

Chapter 30
Creating Internet server applications 30-1

Terminology and standards. 30-1
Parts of a Uniform Resource Locator. 30-2
URI vs. URL 30-2
HTTP request header information 30-3
HTTP server activity. 30-3
Composing client requests 30-3

Serving client requests 30-4
Responding to client requests 30-4
Web server applications 30-5
Types of Web server applications 30-5
ISAPI and NSAPI 30-5
CGI stand-alone 30-5
Win-CGI stand-alone 30-5
Creating Web server applications 30-6
The Web module. 30-6
The Web Application object 30-7
The structure of a Web server application 30-7
The Web dispatcher. 30-8
Adding actions to the dispatcher 30-9
Dispatching request messages 30-9
Action items 30-10
Determining when action items fire. 30-10
The target URL. 30-10
The request method type 30-10
Enabling and disabling action items. 30-11
Choosing a default action item. 30-11
Responding to request messages
with action items 30-12
Sending the response 30-12
Using multiple action items 30-12
Accessing client request information 30-13
Properties that contain request header
information. 30-13
Properties that identify the target 30-13
Properties that describe the Web
client. 30-13
Properties that identify the
purpose of the request. 30-14
Properties that describe the
expected response 30-14
Properties that describe the content 30-14
The content of HTTP request messages. 30-15
Creating HTTP response messages 30-15
Filling in the response header 30-15
Indicating the response status 30-15
Indicating the need for client action 30-16
Describing the server application 30-16
Describing the content 30-16
Setting the response content 30-16
Sending the response 30-17
Generating the content of response
messages 30-17
Using page producer components. 30-18
HTML templates. 30-18
Specifying the HTML template. 30-19

Converting HTML-transparent tags	30-19	Using socket components	31-5
Using page producers from an action item	30-19	Using client sockets	31-5
Chaining page producers together	30-20	Specifying the desired server.	31-6
Using database information in responses	30-21	Forming the connection.	31-6
Adding a session to the Web module	30-22	Getting information about the connection	31-6
Representing database information in HTML.	30-22	Closing the connection	31-6
Using dataset page producers.	30-22	Using server sockets	31-6
Using table producers	30-23	Specifying the port.	31-7
Specifying the table attributes.	30-23	Listening for client requests	31-7
Specifying the row attributes	30-23	Connecting to clients	31-7
Specifying the columns.	30-24	Getting information about connections	31-7
Embedding tables in HTML documents	30-24	Closing server connections	31-8
Setting up a dataset table producer.	30-24	Responding to socket events.	31-8
Setting up a query table producer	30-24	Error events	31-8
Debugging server applications.	30-25	Client events	31-9
Debugging ISAPI and NSAPI applications	30-25	Server events.	31-9
Debugging under Windows NT.	30-25	Events when listening.	31-9
Debugging with a Microsoft IIS server.	30-25	Events with client connections	31-9
Debugging under MTS.	30-26	Reading and writing over socket connections	31-10
Debugging with a Windows 95 Personal Web Server	30-27	Non-blocking connections	31-10
Debugging with Netscape Server Version 2.0	30-28	Reading and writing events	31-10
Debugging CGI and Win-CGI applications	30-29	Blocking connections	31-11
Simulating the server.	30-29	Using threads with blocking connections	31-11
Debugging as a DLL	30-29	Using TWinSocketStream.	31-12
		Writing client threads	31-12
		Writing server threads.	31-13
 Chapter 31		 Part IV	
Working with sockets	31-1	Developing COM-based applications	
Implementing services	31-1	 Chapter 32	
Understanding service protocols	31-2	Overview of COM technologies	32-1
Communicating with applications	31-2	COM as a specification and implementation	32-1
Services and ports	31-2	COM extensions	32-2
Types of socket connections.	31-2	Parts of a COM application	32-2
Client connections	31-3	COM interfaces	32-3
Listening connections	31-3	The fundamental COM interface, IUnknown	32-4
Server connections	31-3	COM interface pointers	32-4
Describing sockets	31-3	COM servers	32-5
Describing the host.	31-4	CoClasses and class factories	32-6
Choosing between a host name and an IP address	31-4	In-process, out-of-process, and remote servers	32-6
Using ports	31-5		

The marshaling mechanism	32-8
Aggregation	32-8
COM clients	32-9
COM extensions	32-9
Automation servers	32-11
Active Server Pages	32-12
ActiveX controls	32-12
Active Documents	32-13
Transactional objects	32-13
Type libraries	32-14
The content of type libraries	32-14
Creating type libraries	32-15
When to use type libraries	32-15
Accessing type libraries	32-16
Benefits of using type libraries	32-16
Using type library tools	32-17
Implementing COM objects with	
wizards	32-17
Code generated by wizards	32-20

Chapter 33

Working with type libraries 33-1

Type Library editor	33-2
Parts of the Type Library editor.	33-2
Toolbar	33-3
Object list pane	33-4
Status bar	33-5
Pages of type information	33-5
Type library elements	33-7
Interfaces.	33-8
Dispinterfaces	33-9
CoClasses	33-9
Type definitions.	33-9
Modules	33-10
Using the Type Library editor.	33-10
Valid types.	33-11
Creating a new type library.	33-12
Opening an existing type library	33-12
Adding an interface to the type	
library	33-13
Modifying an interface using the	
type library	33-13
Adding properties and methods	
to an interface or dispinterface	33-14
Adding a CoClass to the type	
library	33-15
Adding an interface to a CoClass	33-15

Adding an enumeration to the	
type library	33-15
Adding an alias to the type library	33-16
Adding a record or union to the	
type library	33-16
Adding a module to the type	
library	33-16
Saving and registering type library	
information	33-17
Saving a type library	33-17
Refreshing the type library	33-18
Registering the type library.	33-18
Exporting an IDL file	33-18
Deploying type libraries	33-18

Chapter 34

Creating COM clients 34-1

Importing type library information.	34-2
Using the Import Type Library dialog	34-3
Using the Import ActiveX dialog	34-4
Code generated when you import	
type library information	34-5
Controlling an imported object	34-6
Using component wrappers	34-6
ActiveX wrappers	34-7
Automation object wrappers	34-7
Using data-aware ActiveX controls	34-8
Example: Printing a document with	
Microsoft Word	34-10
Step 1: Prepare C++Builder for	
this example	34-10
Step 2: Import the Word	
type library	34-10
Step 3: Use a VTable or dispatch	
interface object to control	
Microsoft Word.	34-11
Step 4: Clean up the example.	34-12
Writing client code based on type	
library definitions	34-12
Connecting to a server	34-12
Controlling an Automation server	
using a dual interface	34-13
Controlling an Automation server	
using a dispatch interface.	34-13
Handling events in an automation	
controller	34-14
Creating Clients for servers that do not	
have a type library	34-16

Chapter 35

Creating simple COM servers 35-1

Overview of creating a COM object	35-1
Designing a COM object	35-2
Using the COM object wizard	35-2
Using the Automation object wizard	35-4
Choosing a threading model	35-5
Writing an object that supports the free threading model	35-6
Writing an object that supports the apartment threading model	35-7
Writing an object that supports the neutral threading model	35-8
Specifying ATL options	35-8
Defining a COM object's interface	35-9
Adding a property to the object's interface	35-9
Adding a method to the object's interface	35-10
Exposing events to clients	35-10
Managing events in your Automation object	35-11
Automation interfaces	35-11
Dual interfaces	35-12
Dispatch interfaces	35-13
Custom interfaces	35-14
Marshaling data	35-14
Automation compatible types	35-14
Type restrictions for automatic marshaling	35-15
Custom marshaling	35-15
Registering a COM object	35-16
Registering an in-process server	35-16
Registering an out-of-process server	35-16
Testing and debugging the application	35-17

Chapter 36

Creating an Active Server Page 36-1

Creating an Active Server Object	36-2
Using the ASP intrinsics	36-3
Application	36-3
Request	36-4
Response	36-4
Session	36-5
Server	36-6
Creating ASPs for in-process or out-of-process servers	36-7
Registering an Active Server Object	36-7
Registering an in-process server	36-7

Registering an out-of-process server	36-8
Testing and debugging the Active Server Page application.	36-8

Chapter 37

Creating an ActiveX control 37-1

Overview of ActiveX control creation	37-2
Elements of an ActiveX control	37-2
VCL control.	37-3
ActiveX wrapper.	37-3
Type library.	37-3
Property page	37-3
Designing an ActiveX control	37-4
Generating an ActiveX control from a VCL control	37-4
Generating an ActiveX control based on a VCL form.	37-6
Licensing ActiveX controls.	37-7
Customizing the ActiveX control's interface	37-8
Adding additional properties, methods, and events	37-9
Adding properties and methods	37-9
Adding events	37-10
Enabling simple data binding with the type library.	37-11
Creating a property page for an ActiveX control	37-13
Creating a new property page	37-13
Adding controls to a property page	37-14
Associating property page controls with ActiveX control properties	37-14
Updating the property page	37-14
Updating the object	37-15
Connecting a property page to an ActiveX control.	37-15
Registering an ActiveX control	37-15
Testing an ActiveX control	37-16
Deploying an ActiveX control on the Web.	37-16
Setting options.	37-17

Chapter 38

Creating MTS or COM+ objects 38-1

Understanding transactional objects	38-2
Requirements for a transactional object	38-2
Managing resources	38-3
Accessing the object context	38-3

Just-in-time activation	38-4
Resource pooling	38-5
Database resource dispensers	38-5
Shared property manager	38-6
Releasing resources	38-9
Object pooling	38-9
MTS and COM+ transaction support	38-9
Transaction attributes	38-10
Setting the transaction attribute	38-11
Stateful and stateless objects	38-12
Influencing how transactions end	38-12
Initiating transactions	38-13
Setting up a transaction object on the client side	38-13
Setting up a transaction object on the server side	38-14
Transaction timeout	38-15
Role-based security	38-16
Overview of creating transactional objects	38-17
Using the Transactional Object wizard	38-17
Choosing a threading model for a transactional object	38-18
Activities	38-19
Generating events under COM+	38-20
Using the Event Object wizard	38-20
Firing events using a COM+ event object	38-21
Passing object references	38-22
Using the SafeRef method	38-22
Callbacks	38-23
Debugging and testing transactional objects	38-23
Installing transactional objects	38-24
Administering transactional objects	38-25

Part V

Creating custom components

Chapter 39

Overview of component creation 39-1

Visual Component Library	39-1
Components and classes	39-2
How do you create components?	39-2
Modifying existing controls	39-3
Creating windowed controls	39-3
Creating graphic controls	39-4

Subclassing Windows controls	39-4
Creating nonvisual components	39-4
What goes into a component?	39-5
Removing dependencies	39-5
Properties, methods, and events	39-5
Properties	39-6
Events	39-6
Methods	39-6
Graphics encapsulation	39-7
Registration	39-7
Creating a new component	39-7
Using the Component wizard	39-8
Creating a component manually	39-11
Creating a unit file	39-11
Deriving the component	39-11
Declaring a new constructor	39-12
Registering the component	39-12
Testing uninstalled components	39-14
Testing installed components	39-16
Installing a component on the Component palette	39-16
Component file locations	39-17
Adding the component	39-17

Chapter 40

Object-oriented programming for component writers 40-1

Defining new classes	40-1
Deriving new classes	40-2
To change class defaults to avoid repetition	40-2
To add new capabilities to a class	40-2
Declaring a new component class	40-3
Ancestors, descendants, and class hierarchies	40-3
Controlling access	40-4
Hiding implementation details	40-4
Defining the component writer's interface	40-6
Defining the runtime interface	40-6
Defining the design-time interface	40-7
Dispatching methods	40-8
Regular methods	40-8
Virtual methods	40-9
Overriding methods	40-9
Abstract class members	40-10
Classes and pointers	40-10

Chapter 41	
Creating properties	41-1
Why create properties?	41-1
Types of properties.	41-2
Publishing inherited properties	41-2
Defining properties	41-3
The property declaration	41-3
Internal data storage	41-4
Direct access.	41-4
Access methods.	41-5
The read method	41-6
The write method.	41-6
Default property values	41-7
Specifying no default value	41-7
Creating array properties	41-8
Storing and loading properties.	41-9
Using the store-and-load mechanism	41-10
Specifying default values	41-10
Determining what to store.	41-11
Initializing after loading.	41-12
Storing and loading unpublished properties	41-12
Creating methods to store and load property values	41-12
Overriding the DefineProperties method.	41-13
Chapter 42	
Creating events	42-1
What are events?	42-1
Events are closures.	42-2
Events are properties.	42-2
Event types are closure types	42-3
Event handlers have a return type of void	42-3
Event handlers are optional.	42-3
Implementing the standard events.	42-4
Identifying standard events.	42-4
Standard events for all controls	42-4
Standard events for standard controls.	42-5
Making events visible	42-5
Changing the standard event handling	42-5
Defining your own events	42-6
Triggering the event	42-6
Two kinds of events	42-6
Defining the handler type	42-7
Simple notifications	42-7
Event-specific handlers	42-7
Returning information from the handler	42-7
Declaring the event	42-7
Event names start with "On".	42-8
Calling the event.	42-8
Chapter 43	
Creating methods	43-1
Avoiding dependencies	43-1
Naming methods	43-2
Protecting methods.	43-3
Methods that should be public.	43-3
Methods that should be protected.	43-3
Making methods virtual	43-3
Declaring methods	43-4
Chapter 44	
Using graphics in components	44-1
Overview of graphics.	44-1
Using the canvas	44-2
Working with pictures	44-3
Using a picture, graphic, or canvas	44-3
Loading and storing graphics	44-4
Handling palettes	44-4
Specifying a palette for a control.	44-5
Off-screen bitmaps	44-5
Creating and managing off-screen bitmaps	44-6
Copying bitmapped images	44-6
Responding to changes.	44-6
Chapter 45	
Handling messages	45-1
Understanding the message-handling system	45-1
What's in a Windows message?	45-2
Dispatching messages.	45-2
Tracing the flow of messages.	45-3
Changing message handling.	45-3
Overriding the handler method	45-4
Using message parameters.	45-4
Trapping messages	45-5
Creating new message handlers.	45-5
Defining your own messages	45-6
Declaring a message identifier	45-6

Declaring a message-structure type	45-6
Declaring a new message-handling method.	45-7
Chapter 46	
Making components available at design time	46-1
Registering components.	46-1
Declaring the Register function.	46-2
Writing the Register function	46-2
Specifying the components	46-2
Specifying the palette page	46-3
Using the RegisterComponents function	46-3
Adding palette bitmaps	46-4
Providing Help for your component.	46-5
Creating the Help file	46-5
Creating the entries.	46-5
Making component help context-sensitive	46-7
Adding component help files	46-7
Adding property editors	46-7
Deriving a property-editor class	46-8
Editing the property as text	46-9
Displaying the property value.	46-9
Setting the property value	46-9
Editing the property as a whole	46-9
Specifying editor attributes	46-10
Registering the property editor	46-11
Adding component editors	46-12
Adding items to the context menu	46-12
Specifying menu items	46-13
Implementing commands	46-13
Changing the double-click behavior	46-14
Adding clipboard formats.	46-14
Registering the component editor	46-15
Property categories	46-15
Registering one property at a time	46-16
Registering multiple properties at once	46-16
Property category classes	46-17
Built-in property categories	46-17
Deriving new property categories	46-18
Using the IsPropertyInCategory function	46-18
Compiling components into packages.	46-19
Troubleshooting custom components	46-19

Chapter 47	
Modifying an existing component	47-1
Creating and registering the component	47-1
Modifying the component class	47-2
Overriding the constructor.	47-3
Specifying the new default property value	47-3

Chapter 48	
Creating a graphic component	48-1
Creating and registering the component	48-1
Publishing inherited properties	48-2
Adding graphic capabilities	48-3
Determining what to draw	48-3
Declaring the property type	48-4
Declaring the property	48-4
Writing the implementation method	48-4
Overriding the constructor and destructor.	48-5
Changing default property values	48-5
Publishing the pen and brush	48-6
Declaring the data members	48-6
Declaring the access properties.	48-6
Initializing owned classes.	48-7
Setting owned classes' properties	48-8
Drawing the component image	48-9
Refining the shape drawing	48-10

Chapter 49	
Customizing a grid	49-1
Creating and registering the component	49-1
Publishing inherited properties	49-3
Changing initial values.	49-3
Resizing the cells	49-4
Filling in the cells	49-5
Tracking the date	49-6
Storing the internal date	49-6
Accessing the day, month, and year	49-7
Generating the day numbers	49-8
Selecting the current day	49-10
Navigating months and years	49-11
Navigating days.	49-12
Moving the selection	49-12
Providing an OnChange event.	49-12
Excluding blank cells	49-13

Chapter 50	
Making a control data aware	50-1
Creating a data-browsing control	50-1
Creating and registering the component.	50-2
Making the control read-only	50-3
Adding the ReadOnly property	50-3
Allowing needed updates	50-4
Adding the data link	50-5
Declaring the data member	50-5
Declaring the access properties	50-5
An example of declaring access properties	50-6
Initializing the data link	50-7
Responding to data changes	50-7
Creating a data-editing control	50-8
Changing the default value of FReadOnly	50-9
Handling mouse-down and key-down messages	50-9
Responding to mouse-down messages	50-9
Responding to key-down messages	50-10

Updating the field datalink class	50-11
Modifying the Change method	50-12
Updating the dataset	50-12

Chapter 51	
Making a dialog box a component	51-1
Defining the component interface.	51-1
Creating and registering the component	51-2
Creating the component interface.	51-3
Including the form unit files	51-3
Adding interface properties	51-4
Adding the Execute method	51-5
Testing the component	51-6

Appendix A	
ANSI implementation-specific standards	A-1
Index	I-1

Tables

1.1	Typefaces and symbols	1-2	16.2	UpdateStatus values	16-7
2.1	Component palette pages	2-14	16.3	UpdateMode values	16-8
4.2	Menu Designer context menu commands	4-23	16.4	ProviderFlags values	16-8
4.3	Setting speed buttons' appearance.	4-31	17.1	Database-related informational methods for session components	17-8
4.4	Setting tool buttons' appearance.	4-33	17.2	TSessionList properties and methods	17-17
4.5	Setting a cool button's appearance.	4-34	19.1	Values for the dataset State property	19-3
5.1	Properties of selected text.	5-8	19.2	Navigational methods of datasets	19-9
5.2	Fixed vs. variable owner-draw styles	5-12	19.3	Navigational properties of datasets.	19-10
6.1	Graphic object types.	6-3	19.4	Comparison and logical operators that can appear in a filter	19-18
6.2	Common properties of the Canvas object	6-3	19.5	FilterOptions values	19-20
6.3	Common methods of the Canvas object	6-4	19.6	Filtered dataset navigational methods	19-20
6.4	Mouse-event parameters	6-23	19.7	Dataset methods for inserting, updating, and deleting data	19-21
6.5	Multimedia device types and their functions	6-31	19.8	Methods that work with entire records	19-24
7.1	Thread priorities	7-3	19.9	Dataset events.	19-26
7.2	WaitFor return values	7-9	19.10	TDBDataSet database and session properties and function	19-28
8.1	Exception handling compiler options	8-11	19.11	Properties, events, and methods for cached updates.	19-30
8.2	Selected exception classes.	8-22	20.1	Field components.	20-1
9.1	Object model comparison.	9-6	20.2	TFloatField properties that affect data display	20-2
9.2	Equality comparison !A == !B of BOOL variables	9-16	20.3	Special persistent field kinds	20-6
9.3	Examples of RTTI mappings from Object Pascal to C++.	9-18	20.4	Field component properties	20-12
10.1	Design-time packages.	10-4	20.5	Field component formatting routines	20-16
10.2	Package-specific compiler directives	10-10	20.6	Field component events	20-17
10.3	Package-specific command-line linker switches	10-12	20.7	Selected field component methods	20-18
10.4	Compiled package files	10-12	20.8	Field component conversion functions.	20-19
11.1	VCL objects that support BiDi	11-3	20.9	Special conversion results	20-19
11.2	Estimating string lengths	11-8	20.10	Types of object field components.	20-23
12.1	Application files	12-2	20.11	Common object field descendant properties	20-23
12.2	SQL database client software files	12-5	21.1	Table types recognized by the BDE based on file extension	21-3
13.1	Data Dictionary interface	13-5	21.2	TableType values	21-4
14.1	Possible values for the TransIsolation property.	14-7	21.3	Index-based search methods	21-6
14.2	Transaction isolation levels	14-7	21.4	BatchMove import modes	21-19
15.1	MIDAS components.	15-3	21.5	Batch move modes	21-21
15.2	Connection components	15-4	24.1	ADO components.	24-2
15.3	AppServer interface members	15-8			
15.4	Javascript libraries.	15-28			
16.1	Provider options	16-3			

24.2	Parameter direction property.	24-23	32.2	C++Builder wizards for implementing COM, Automation, and ActiveX objects.	32-18
25.1	Summary operators for maintained aggregates	25-10	33.1	Type library pages	33-5
25.2	Client datasets properties and method for handling data requests . . .	25-17	35.1	Threading models for COM objects . . .	35-5
26.1	TUpdateRecordType values	26-9	36.1	IApplicationObject interface members.	36-4
26.2	Return values for UpdateStatus	26-10	36.2	IRequest interface members	36-4
26.3	UpdateKind values	26-24	36.3	IResponse interface members	36-5
26.4	UpdateAction values	26-25	36.4	ISessionObject interface members	36-6
27.1	Data controls	27-2	36.5	IServer interface members	36-6
27.2	Properties affecting editing in data controls	27-4	38.1	IObjectContext methods for transaction support.	38-12
27.3	Data-aware list box and combo box controls.	27-11	38.2	Threading models for transactional objects	38-18
27.4	TDBLookupListBox and TDBLookupComboBox properties. . . .	27-14	38.3	Call synchronization options	38-20
27.5	Column properties.	27-22	39.1	Component creation starting points . . .	39-3
27.6	Expanded TColumn Title properties . .	27-22	40.1	Levels of visibility within an object. . .	40-4
27.7	Properties that affect the way ADT and array fields appear in a TDBGrid	27-23	41.1	How properties appear in the Object Inspector.	41-2
27.8	Expanded TDBGrid Options properties.	27-24	44.1	Canvas capability summary.	44-3
27.9	Grid control events	27-27	44.2	Image-copying methods	44-6
27.10	Selected database control grid properties	27-28	46.1	Predefined property-editor types.	46-8
27.11	TDBNavigator buttons	27-29	46.2	Methods for reading and writing property values	46-9
30.1	Web server application components. . .	30-5	46.3	Property-editor attribute flags.	46-10
30.2	MethodType values	30-11	46.4	Property categories	46-17
32.1	COM object requirements.	32-11	A.1	Options needed for ANSI compliance . .	A-1
			A.2	Identifying diagnostics in C++	A-3