

Marco Cantù

**Delphi 2007
Handbook**

Piacenza, Italy, June-September 2007

to Lella, the love of my life,
and to Benedetta and Jacopo,
our love come to life.

Acknowledgments

Any book owes a lot to many people, and this one is no exception. Although I had been thinking of this for a while, it was Nick Hodges (the Delphi product manager) who finally pushed me towards Lulu for publishing, and kept asking me how things were progressing. I have discussed publishing and self-publishing with many authors over years. I've certainly had a lot of input from Bruce Eckel, and also the practical experience of Julian Bucknall with Lulu, described on his blog, provided very valuable input¹.

A lot of people in the Delphi community keep providing me ideas, insights, and feedback about what I write. My thank you reaches the entire community, including the "never happy" ones. The first people who helped me when I started with Delphi at Borland were David I (still at the company) and Zack Urlocker (currently at MySQL). I had many chats and a lot of help from present and past Delphi R&D members including Allen Bauer, Danny Thorpe, Mark Edington, Jim Tierny, Steve Threfethen, Corbin Dunn, Jesper Hogstrom and other CodeGear Developer Relations employees including Anders Ohlsson, John Kaster, Christine Ellis, and (in the past) Charlie Calvert. I'm also in touch with many product managers and product evange-

¹ For more information about Bruce Eckel's work see <http://www.mindview.net>, for more about Julian Bucknall see <http://www.boyet.com>.

lists, including Nick Hodges, Jason Vokes, Malcolm Groves, and Pawel Glowacki.

Special thanks go to my friends and Delphi experts (well, most of them) Bob Swart, Andrea Provaglio, Norm McIntosh, Joanna Pooley, Ray Konopka, Cary Jensen, Brian Long, Holger Flick, Chris Frizelle, Michael Lant, Joanna Carter, Jim Cooper, Christian Gross, Jason Southwell, Olaf Monien, Hadi Hariri, Chad Hower, Mike Orris, Jeroen Plumiers, Fabrizio Bitti, and the entire Italian Delphi&Dintorni Team.

For the actual production of the book I've had a lot of help from my editor Peter Wood and tech reviewers Holger Flick, Fernando Madruga, Sergio Sette, Marco Breveglieri, and (for specific portions) Nando Dessena (dbExpress), Pierre le Riche (memory manager), and Sinan Karaca (InstallAware). The book you are reading would have had many more mistakes and missing details, if it wasn't for their help.

Lacking the formal approach (and money) of an established publisher I had to scout around for available people... and offer some of them limited deals. Given my non-existent graphical ability, and since I wanted to have a nice book cover, I benefited from the help of graphic designer Fabrizio Schiavi² who describes the cover idea as follows:

"I wanted something referring to nature, but also a mathematical reference connecting it to computer science. Therefore, I duplicated the birds using the Fibonacci sequence."

Many thanks go also to the people who share my workload at my training and consulting company, Wintech Italia Srl³, Paolo and Michela. Finally, I could not write this book without "stealing" precious time from my wife and family, so if you can read this you'll have to join with me in thanking Lella, Benedetta, and Jacopo.

The final thanks, as you might have seen in previous books of mine, are for all of the people (at all longitudes and latitudes), many of them unknown and unnoticed, who enjoy life and help to build a better world. Drop of water after drop of water, and you help turn deserts into gardens; love gesture after love gesture, you help bring peace to the world.

2 For more information about Fabrizio Schiavi (and his fonts) see <http://www.fsd.it>.

3 More and more up-to-date information about my training and consulting company on its web site, <http://www.wintech-italia.it>. We currently also sell CodeGear products in Italy.

Introduction

Delphi 2007 is the latest version of a very popular development environment based on the Object Pascal language. Formerly from Borland, now from its CodeGear subsidiary, Delphi has been used for twelve years around the world to build thousands of successful applications. Often regarded as a Swiss-army knife development tool for its versatility and flexibility, it is equally well suited for small shareware applications, company-wide client-server software, Internet tools, Web Service architectures, and much more.

Over the last year, CodeGear has refocused on the product, launching the free Turbo Explorer version, reinvigorating the R&D team, getting a new product manager, trying to figure out the customer needs, and releasing a new version (Delphi 2007) in record time. Indeed, as one team member suggested, it is a "Delphi Renaissance".

From Mastering Delphi to this Handbook

After writing 8 Mastering Delphi books, I've come to the decision that the Delphi community needs something different. Rather than writing an all-

4 This was Steve Shaughnessy, and you can find the blog post at:
<http://blogs.codegear.com/stevshaughnessy/archive/2007/03/21/33376.aspx>

encompassing book, covering both introductory and advanced material, I think it makes more sense to write a book focused on existing experienced Delphi developers.

As many of these developers have missed the last couple of versions of Delphi, the aim of the book is to bring them up to speed covering the most important features added since Delphi 7. In most cases the material goes into some depth as it is not meant for beginners. Some of the new technologies, though, are only briefly covered as a way of introduction.

In the book there are some references to material in *Mastering Delphi 2005* (written by myself and published by Sybex, now an imprint of Wiley), there is a lot from my "Delphi 2006 Update Ebook" (freely available online at my web site), and a clear focus on what's new in Delphi 2007 for Win32, the version of Delphi that this book is focused on.⁵ This also implies that this book exclusively covers the Delphi Win32 personality as it does. I might as well come up with a specific Delphi for .NET book, but I think that one of the advantages of a smaller publication should be its focus. The focus here is Win32 development for Windows 2000, Windows XP and Windows Vista, the three platforms Delphi 2007 officially supports.

Formatting and Conventions

This book is published in Lulu's Crown Quarto Format, 7.44" x 9.68". I used the Georgia font for the text, a Verdana font for the headings, and a Lucida Console font (the same I use by default in the Delphi IDE) for the code:

```
| // this is a sample comment
```

Book conventions are very limited. Keyword and symbol names in the text are in a special font, as in "TObject is the base class". I decided to use footnotes rather than in-line notes as they should be less distractive from the text flow. At the same time, using footnotes I felt more free to add extra text and personal annotations, web site references, and more. Many tech editor comments ended up in notes.

5 The book actually covers Update 1 of Delphi 2007 for Win32, released in June 2007.

Examples and Source Code

As it is true for my past books, there are plenty of examples you can learn from and experiment with. The examples discussed here are freely available on line. You'll find the specific address for the code at the main page of my site devoted to the book:

| <http://www.marcocantu.com/dh2007>

The source code for this book is copyrighted freeware. The copyright prevents you from republishing the code in print media or on the Internet without the prior permission of the author.

Seminars and Consulting

I teach a two-day **"Migrating to Delphi 2007"** seminar onsite at company locations worldwide or at my company site in Italy. The class can be included in a longer training session also covering many classic and advanced Delphi topics, from data access to dynamic architectures, from advanced language topics to XML and Web development in Delphi. Refer to my web site for more details. Online seminars are currently under evaluation.

About the Author

My name is Marco Cantù. I'm the author of the best-selling Mastering Delphi book series and I'm considered a prominent figure in the Delphi community. I was one of the recipients of the "Spirit of Delphi" award in 1999, together with Dutch Delphi guru Bob Swart.

I teach and consult on Delphi, and also on XML-related and Web 2.0 technologies, as you can partially see on my AJAX-focused web site:

| <http://ajax.marcocantu.com>

My main web site, mostly covering books and Delphi programming, is at:

| <http://www.marcocantu.com>

My home-grown "Marco Tech Blog" can be found at:

| <http://blog.marcocantu.com>

I live in Piacenza⁶, an ancient town in northern Italy, with my wife Lella (an interior designer), our two kids Benedetta and Jacopo, and our dog Lillo.

Contacting the Author

To contact me you can certainly email me at marco.cantu@gmail.com. There are other options though. I run a newsgroup with an area devoted to my books. This can be reached via NNTP with the following settings⁷:

```
server: news.marcocantu.com
username: delphi
password: spirit
```

and over the web a the URL:

```
http://delphi.newswhat.com/geoxml/forumlistgroups?
area=marcocantu
```

If you looking for generic support on Delphi, and not something related with the book, consider CodeGear's own newsgroup. If you want to receive a monthly newsletter from me, with information about my work (and some Delphi news), you can sign up for my Google Group at:

```
http://groups.google.com/group/marco_cantu
```

This is an area where only I can write, not a real newsgroup, and where you can also find mirrored download files with book examples and the like.

Looking for Tech Support?

If you are looking for professional support, I offer it as well. You can have a protected newsgroup area specific to your company, where you can "privately" discuss any issue, from coding problems to architectural ones, with me and other top Delphi experts, for a very low annual fee.

Check out <http://support.marcocantu.com> for more details about this brand new offer.

⁶ To figure out where my town is and see some aerial pictures you can refer to Google Maps, at <http://maps.google.com/maps?q=piacenza,italy>.

⁷ The NNTP server uses a password shared by all users to prevent newsgroup spam.

Table Of Contents

Acknowledgments.....	5
Introduction.....	7
From Mastering Delphi to this Handbook.....	7
Formatting and Conventions.....	8
Examples and Source Code.....	9
Seminars and Consulting.....	9
About the Author.....	9
Contacting the Author.....	10
Looking for Tech Support?.....	10
Table of Contents.....	11

Table of Contents - 12

1: The Delphi 2007 IDE.....	17
A Short History of the Delphi IDE.....	18
A Single Personality.....	18
A Non-Breaking Version.....	19
Installing Delphi 2007.....	19
The New Folder Structure.....	20
Running the Delphi IDE.....	21
A New IDE.....	22
A Themed IDE.....	23
The Welcome Page.....	23
The Delphi Editor.....	24
Code Folding and Regions.....	25
Source Code File Encoding.....	26
Sync Edit.....	27
Editor History.....	27
Other (Recent) Editor Features.....	29
The VCL Form Designer.....	30
Text Baseline for Components Developers.....	31
Searching in the Tool Palette.....	32
Making Help Insight Work.....	32
Generate XML Documentation.....	33
Other Features of the Delphi IDE.....	35
Extended Search Results.....	35
File Browser View.....	36
Database Management Tools.....	36
(Limited) ALM Support in the IDE.....	37
The New Help System.....	37
2: Code Templates and Refactoring.....	39
Block Completion.....	40
Live Templates.....	40
Customizing and Writing Live Templates.....	43
Custom Live Templates.....	47
A Live Template for Help Insight.....	48
Refactoring.....	49
Declare Refactoring.....	50
Extract to Resource String.....	51
Rename Refactoring.....	52
Extract Method.....	54
Find Unit Refactoring.....	56
Change Parameters.....	57
Extract Interface and Superclass.....	58
Introduce Field, Introduce Variable, and Inline Variable.....	59
More Together Refactorings.....	60
Find References.....	61

3: Project Management and MSBuild.....	63
The .proj File Format.....	64
The Build Process.....	67
Manual Builds.....	67
Compilation Output.....	68
Build Events.....	69
Custom Targets.....	70
Build Configurations.....	71
The Project Manager.....	72
Using Old-Style Build Tools.....	73
4: The Debugger.....	75
About Breakpoints.....	75
New Stack Breakpoints.....	76
Fly-by Evaluation.....	77
Other Features.....	78
5: Recent Updates to the Delphi Language.....	79
When Private Is Really Private.....	80
Classes and Inheritance.....	81
Abstract Classes.....	81
Sealed Classes and Final Methods.....	81
Class Data.....	82
Class Static Methods.....	83
Class Properties.....	86
Nested Types and Nested Constants.....	86
Exceptions and Constructors.....	88
Class Helpers.....	89
The for..in Loop.....	92
Inlining.....	96
Records on Steroids.....	99
Records or Classes?.....	100
Operators Gain New Ground.....	102
The Delphi 2007 Compiler.....	105
Version.....	106
Automatic RTL, or not?.....	106
DEP, ASLR and NX Support.....	107
6: Core RTL Changes.....	109
FastCode in the RTL.....	110
New Features for Collections.....	111
Enumerating Containers (with a Class Helper).....	111
Strings and StringLists.....	114
WideStrings and Unicode.....	115
WideStringLists.....	115

Table of Contents - 14

Other New RTL Features.....	116
Executable Size under the Microscope.....	117
7: Changes in the VCL.....	119
Flexible UI with Margins and Paddings.....	120
Margins.....	120
Padding.....	121
Layout Panels: Flow and Grid.....	121
Align and the Explicit Properties.....	123
ButtonGroup and CategoryButtons.....	124
CategoryButtons Updates in Delphi 2007.....	126
Modal Forms and PopupMode.....	126
Processing Mouse Activation.....	127
Docking Components: DockedTabSet.....	128
The TrayIcon Component.....	130
Assorted New Features.....	131
8: Memory Management (and Robust Applications).....	133
FastMM4.....	134
Comparing Allocators.....	135
Tracking Leaks and Other	
Global Settings.....	141
Memory Status.....	143
Buffer Overruns in the Full FastMM4.....	144
Safe Coding Techniques.....	147
Constructors, Destructors, and Exceptions.....	147
Finally blocks.....	149
Dynamic Type Checking.....	150
Is this Pointer an Object Reference?.....	151
Mixing Objects and Interfaces.....	153
Unit Testing.....	154
DUnit.....	155
9: Windows Vista and the VCL.....	161
Delphi and Vista: The Basics.....	162
Windows' Version Checking.....	162
The Segoe UI Font.....	163
Vista UI Experience.....	165
VCL Applications and the TaskBar.....	165
Extending Glass.....	167
Task Dialogs.....	170
New Common Dialogs.....	176
Themes and Styles.....	178
Enabling Themes.....	179
Themed VCL Controls.....	179

The New TreeView Style.....	180
Windows Resource Protection and User Account Control.....	181
Windows Resource Protection (WRP).....	181
Virtual Storage.....	183
User Account Control.....	183
Elevation.....	184
More Vista Features and Problems.....	187
Built-in Animations.....	187
Internet Explorer in Protected Mode.....	188
PNG Icons.....	189
Debug Your OnPaint.....	189
Delphi and Vista, Again.....	189
10: Database Support and dbExpress 4.....	191
DB Unicode Support.....	192
dbExpress in Delphi 2006.....	194
dbExpress 4 in Delphi 2007.....	195
Using dbExpress 4 (Upgrading a Program).....	197
Using dbExpress 4 (Writing a New Program).....	200
A Second Look to dbExpress 4 (Configuration Files).....	203
dbExpress Drivers.....	204
Delegate Drivers: Connection Sharing and Tracing.....	205
dbExpress 4 Internals.....	212
Using the DBXCommon classes.....	212
Database Explorer for dbExpress.....	218
Other DB-Related Changes.....	219
11: InstallAware and Other Tools.....	221
InstallAware Express.....	222
Installing Runtime Packages.....	222
Installing a dbExpress Application.....	226
WSDL Importer and SOAP.....	227
My Own Delphi Tools.....	228
CanTools Wizards.....	228
Object Debugger.....	230
Memory Snap.....	230
Other Relevant Free Tools.....	231
Indy.....	231
Project JEDI.....	232
GExperts.....	232
OpenXML and libxml2-pas.....	233
Former TurboPower Components.....	233
What Else?.....	235
Professional Components.....	235

Table of Contents - 16

12: Upgrading Projects to Delphi 2007.....	237
Managing Source Code Files.....	238
Backward Compatibility and the Explicit* Properties.....	238
Variants Troubles.....	239
Conflicting Unit Names.....	241
Porting VCL Applications.....	241
Upgrading Database Projects.....	242
BDE Upgrade Paths.....	243
Porting from Indy to Indy 10.....	244
Using Low-Level Sockets.....	246
Web Development.....	250
Postscript.....	251
Index.....	253

1: The Delphi 2007 IDE

This chapter introduces the new features of the Delphi 2007 Integrated Development Environment (IDE). The IDE changed considerably since Delphi 7, but most of the features are quite intuitive, particularly if you have a lot of experience with past versions of Delphi. This chapter provides only a summary of the most relevant features of Delphi 2005 and Delphi 2006 IDEs, with a little more detail on the specific new features of Delphi 2007.

Separate chapters are devoted to the most important new features of the IDE, namely Code Templates and Refactoring (Chapter 2), MSBuild integration (Chapter 3), and the integrated debugger (Chapter 4).

A Short History of the Delphi IDE

The original Delphi IDE, that worked under Windows 3.1, provided the groundwork for all following versions right up to Delphi 7. Around that time, Borland decided to redesign the IDE to open it up to multiple programming languages (or personalities, as they were later called), to support .NET development, and to integrate several Application Lifetime Management (ALM) tools they had bought and were focusing on.

The so-called “Galileo” project is the foundation of all recent Delphi IDEs, from Delphi 8 to Delphi 2007. For Delphi 2005 and Delphi 2006 the IDE was called Borland Developer Studio (BDS), in Delphi 2007 it is called RAD Studio and is now produced by CodeGear.

Delphi 2007 is the 5th version of the Galileo IDE (also including the original C#Builder) and it is the first Delphi IDE produced by CodeGear. Delphi 2007 is also the first IDE with a class library specifically supporting and targeting Windows Vista, but that’s something I’ll cover in detail in Chapter 9.

A Single Personality

After releasing Delphi 2006 as a “Studio” version encompassing multiple personalities, CodeGear has now released Delphi 2007 for Win32, which is an updated version of the same IDE with only one personality installed. If you need to do Delphi for .NET development or use any of the other programming languages supported by Delphi 2006 (C++ and C#), you’ll need to keep or buy the older version. Needless to say, the two can be installed on the same computer, as has always been the case with different versions of Delphi.

To be absolutely factually correct, there might be a few global libraries installed in Windows\System32 that might be replaced when installing a new version of Delphi, such as the Midas.dll library. They are backward compatible, but there might be subtle differences. Also, due to the fact you want to end up with the latest version, it is advisable to install different versions of Delphi in the order they were released (for example, Delphi 7, then Delphi 2006, and finally Delphi 2007). Moreover, the binary compatibility of Delphi 2007 with Delphi 2006 might cause a few troubles, as discussed below.

A Non-Breaking Version

For the first time ever, Delphi 2007 is binary compatible with the previous version. Yes, as odd as it might sound to experienced Delphi users, Delphi 2007 is binary compatible (DCU and BPL compatible) with Delphi 2006. Differently from the past, you can link a Delphi 2007 application that uses a compiled unit (DCU) built with Delphi 2006. Not that this will become the norm, as the next version is likely to break the binary compatibility, but with more frequent versions being released, it makes sense to make some of them compatible to reduce the problems with updating third party components.

Even if there are areas like the new DBExpress library, that could not be compatible, most of the changes in compiler, runtime library, and VCL were made with great care to avoid changing the unit interfaces in a way that would break compatibility. As you can see in the `Forms.pas` unit, for example, the R&D team had to put in some extra effort to add properties (like the Glass support) without changing the forms. I'll cover this specific case in more detail in Chapter 8.

What is relevant to note is that to obtain binary compatibility CodeGear replaced some of the files of the Delphi 2006 runtime packages with new versions. This means that installing Delphi 2007 can affect the Delphi 2006 IDE (and it did cause some troubles like the missing TaskBar button upon Delphi 2006 startup, at least until the Update 1 of Delphi 2007 was released). The same can happen to your applications if they use runtime packages and the corresponding BPL files were deployed in the `Windows/System32` folder. Another application installing updated Delphi 2007 packages could affect the behavior of your program. Had you installed the runtime packages in the application folder, the two programs can coexist and even run side by side.

Installing Delphi 2007

The Delphi 2007 installation sequence is quite different from the past. There is a single installation DVD, including all languages and editions. By providing the initial license key, the installer determines which version the

user is entitled to and what to install. As an alternative, the Electronic Software Delivery (ESD) versions has a small setup “stub” of a few megabytes and will download the various prerequisites (like the .NET 2.0 SDK) and the Delphi modules it needs to install as it progresses through the installation.

More information about the InstallAware installer (mostly related to using the light version included in Delphi 2007) is in Chapter 11. CodeGear is hoping to be able to use this new installer to push electronic licensing (less physical DVDs to be shipped) and to better manage the installation of updates, previously quite messy with formal updates, hot fixes, hot fix rollups, and more. With Delphi 2006 developers had to follow newsgroups and blogs to figure out which updates were available, when switching to Delphi 2007, the IDE is now capable of downloading updates.

The New Folder Structure

After installing Delphi 2007 you can notice that the software is installed on your hard drive in a different location than in the past, at least if you pick the default installation folders. A few changes have already taken place with previous Galileo versions, but Delphi 2007 is the first version that installs properly on Vista supporting the “Limited User Account Control” (Limited UAC) rules. As we’ll see in Chapter 8, by default Vista doesn’t allow programs to use the file system at will, limiting access to key folders such as Program Files.

The other reason that affected the Delphi installation folder structure is the change in the name of the company owning the product (from Borland to CodeGear) and the name of the IDE itself (from BDS to RAD Studio). This is why the main installation folder in Delphi 7 was:

```
| C:\Program Files\Borland\Delphi7
```

in Delphi 2006 it was

```
| C:\Program Files\Borland\BDS\4.0
```

and in Delphi 2007 it is:

```
| C:\Program Files\CodeGear\RAD Studio\5.0
```

Recent versions of Delphi also moved the Projects folder from the Program Files area to the user’s documents area. The latest version also moves the examples and other configuration files to the user’s shared documents area.

This is a summary of most relevant folders on Windows Vista (users folders have a different physical structure in Windows XP)⁸:

Main folder:

| C:\Program Files\CodeGear\RAD Studio\5.0

Common files:

| C:\Program Files\Common Files\CodeGear Shared

Examples:

| C:\Users\Public\Documents\RAD Studio\Demos

Compiled packages:

| C:\Users\Public\Documents\RAD Studio\5.0

Projects:

| C:\Users\<username>\Documents\RAD Studio

Running the Delphi IDE

As Delphi 2007 provides only Win32 support, you don't use the `-p` flag to start up an individual personality⁹. However, like with past versions of Delphi you can use the `-r` flag to run the IDE with a different set of registry entries. This allows you to install different sets of components and change any IDE setting without disrupting your standard configuration. You can also create a stripped down version of Delphi (for faster startup and reduced memory consumption) whilst keeping the full version ready to use.

Technically, the Delphi `-r` command line parameter specifies the base registry key to use. For example, create a shortcut like this:

```
| "C:\Program Files\CodeGear\RAD Studio\5.0\bin\bds.exe"  
| -rSmall
```

8 Some of the actual folders also depend on the installation choice of making Delphi available to "All users" or "Just me".

9 For more information on the `-p` flag, Delphi startup times, custom registry settings and ... the kitchen sink, see Mark Edington's article at:
<http://homepages.codegear.com/modington/DelphiStartupTimes.htm>.

The first time you run it, Delphi creates a brand new set of registry keys, using default settings from:

```
| HKEY_LOCAL_MACHINE\SOFTWARE\Borland\BDS\5.0
```

and copying them into:

```
| HKEY_CURRENT_USER\Software\Borland\Small\5.0
```

In other words, the name you provide replaces the BDS name in the registry tree¹⁰. If you don't want to start with brand new settings but edit your current configuration, export the standard Borland\BDS\5.0 registry key, edit it to change the key name, and re-import it in the registry. A nice tool to handle multiple start up scenarios is Jeremy North's Delphi Configuration Manager, which can be found at:

```
| http://www.jed-software.com/dcm.htm
```

In addition to the `-r` parameter (which worked in past version of Delphi, but was not documented), there are other command line parameters you can use when starting Delphi 2007, including:

`-ns` (no splash): disables the display of the splash screen

`-np` (no page): disables the Welcome page

A New IDE

If you come from Delphi 7 or previous versions, you'll certainly notice that the IDE is quite different. Still, if you gloss over the central pane with the Welcome Page (that hosts the editor as soon as you open a file) and the fact that all panes are hooked inside the main windows by default, the content of these panes looks quite familiar.

For example, the content of the Structure View is similar to that of the Object Tree View of Delphi 7 when a designer is active, while it resembles the Code Explorer when you are working in the editor. The most striking difference is in the Tool Palette, which replaces the Components Palette but can also contain the elements of the New Items dialog box (obtained with the File | New | Other menu command) when you are working on the code.

¹⁰ Yes, oddly enough the registry still refers to "Borland" and "BDS". Probably fixing all of these references would have caused problems to third-party plugin tools for the Delphi IDE.

In the RAD Studio IDE, most panes are more flexible and host different content depending on what the user is doing, particularly which main window you are working on (visual designer, editor, UML viewer, and so on). For this reason, it takes some time to get used to it. Indeed, some of the new features can be disabled and you can revert back to the “undocked” mode (simply pick the Classic Undocked layout in the Layouts combo box to see how it looks). As the IDE tends to be quite productive after a while, I suggest you try out the new layout at least until you get used to it.

In any case, there are a large number of new features in the IDE, but I don't want to devote too many pages to them. In this chapter I'm focusing on features you can easily miss or are not so obvious, for everything else just try things out.

A Themed IDE

The new IDE is themed. It looks much better, particularly on Vista but also under Windows XP. You can disable the themes in the IDE by adding an un-themed *bds.exe.manifest* file to the Delphi bin folder and restarting the IDE.

As we'll see in Chapter 9, if you turn on the corresponding project option you can see the forms of the your projects themed also at design time¹¹. This is a significant difference from the past.

The Welcome Page

As you run the IDE, your starting point will be the Welcome page. The Welcome page is a pane hosting Internet Explorer and allowing you to view some pertinent information and browse the Web as well. While in Delphi 2005 and 2006 the prime goal of the Welcome page was to read RSS feeds (something I'd rather do with a specific tool or a generic Web Browser), the focus in Delphi 2007 is to manage projects.

The most recent projects you've opened are listed with some details (including the location and the last change), but you can also save your favorite projects and arrange them within Favorite Groups. Clicking on the manage Favorites in the Welcome Page takes you to the favorites management page.

¹¹ You have to close and reopen the project in the IDE to enable themes at design time.

In the top menu there are still links to RSS feeds, product updates, online resources, help, and documentation files. Since I started using Delphi 2007 I generally keep the Welcome Page open (displaying the Project page) and don't close it as I used to do with past versions.

In any case, if you want to customize this page, you can simply modify some of its configuration files by default available in the folder:

| C:\Program Files\CodeGear\RAD Studio\5.0\welcomepage

There you can find the core HTML and CSS files, along with the images displayed, plus a lot of JavaScript code, the XML-based configuration files for the menus and the RSS feeds, and the XSLT code for generating the HTML from the feeds and the projects list.

The Delphi Editor

The editor is probably the part of the IDE you spend most time with and it is also the one that saw the largest number of real and significant improvements since Delphi 7. Compared to past versions, as soon as you open an Object Pascal source code file you will notice that there are line numbers on the side of the source code, a feature you can disable by turning off the corresponding editor setting.

Instead of seeing the number of each and every line like in Delphi 2005, you see only every tenth line, with a hyphen indicating the intermediate lines (every fifth line), and a single dot for the others. The only exception is the line with the edit cursor, which has the line number displayed. Here is an example (the cursor is on line 17):

```

1 | type
2 | TForm0 = class(TForm)
3 |     Label1: TLabel;
4 |     Edit1: TEdit;
5 |     Button1: TButton;
6 |     procedure Button1Click(Sender: TObject);
7 | private
8 |     { Private declarations }
9 | public
10 |     { Public declarations }
11 | end;
12 |
13 | var
14 |
15 |
16 |
17 |
18 |
19 |
20 |
21 | var

```

Notice that the line numbers are not placed in the same gutter with breakpoints and bookmarks, but have their own separate area. On the side of the editor you can also notice some colored lines. These mark the source code lines modified since the last time the file was saved (green line) or within the current editing session (yellow line).

Beside these visual elements, the most useful changes in the editor are related to several code writing helpers, from Block Completion to the new Live Templates, which will be covered in detail in Chapter 2. Notice that most of these new features are not activated by shortcut keys you have to learn, but show up automatically as you type, being triggered by simple keys like Enter, the Space Bar, or the Tab. Having new features embedded in Code Completion (manually triggered by Ctrl+Space, but often displayed automatically) is much better than having to invoke a specific request, like you had to do by pressing Ctrl+J with the old Code Templates (a much less intuitive sequence).

Code Folding and Regions

Since Delphi 2005, the IDE allows you to fold (or collapse) any declaration, including class declarations, methods, or entire sections such as the implementation section of a unit. What's important and the main reason this feature was introduced, though, is that you can define custom collapsible regions using the `$REGION` and `$ENDREGION` directives (which look like compiler directives, but are actually ignored by the compiler).

For example, in the `RegionsTest` program, I've written the following code¹²:

```
{ $REGION 'extra code' }
// this is code you want to hide by default
procedure TestMessage;
begin
    ShowMessage ('Test');
end;
{ $ENDREGION }
```

¹² The easiest way to write these two directives is to select some text and use the "region" Surround Template, with the Surround command of the context menu of the editor window.

When you open the file with this code, by default, you'll see this section folded with a caption corresponding to the string provided in the `$REGION` directive:

```

1  implementation
2
3  {SR *.dfm}
4
5  Source Code
6
7  procedure TForm1.Button1Click(Sender: TObject);
8  begin
9      TestMessage;
10 end;
```

You can enable or disable folding with the `BDS/5.0/Editor/Options/Enable Elisions` registry key and determine the automatic collapsing of region blocks in files you open with `BDS/5.0/Editor/Options/Auto Collapse Region Blocks`.

There are also some new shortcut keys you can use with regions and folding, including:

- **Ctrl+Shift+K+O** to enable and disable code folding¹³.
- **Ctrl+Shift+K+E** to collapse the nearest block.
- **Ctrl+Shift+K+U** to expand the nearest block.
- **Ctrl+Shift+K+A** to expand all code blocks.

Source Code File Encoding

In the past, the Delphi editor allowed you to work only with ANSI source code files. While this is still the default format of the files you produce, the code editor internally works with the UTF-8 encoding and supports saving and converting files in a number of formats, including UTF-8 and UTF-16 with Little and Big Endian mode. You can use the File Format submenu of the editor context menu to change the file format.

¹³ When you are working on very large source code files, disabling code folding can apparently make the editor faster, as suggested by Chris Bensen on: <http://chrisbensen.blogspot.com/2007/07/delphi-tips-and-tricks-turn-off-code.html>.

You can also control the default file format by adding the registry key:

```
| BDS\5.0\Editor\Options\DefaultFileFilter
```

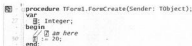
For example, you can use the string `Borland.FileFilter.UTF8ToUTF8` to use UTF8 encoded files by default.

Sync Edit

Another feature worth mentioning is Synchronized Editing, or Sync Edit. Since Delphi 2005, every time you select a block of code that has at least a repeated symbol, you'll see a small icon with two pencils appearing in the gutter on the side of the editor. By selecting this icon or pressing `Ctrl+Shift+J`, you activate Sync Edit mode.

The editor will highlight repeated words with a box. You can select one of the repeated words with the mouse or even more easily by pressing the `Tab` key, and start typing to edit each of the occurrences at the same time.

Notice that Sync Edit is not fully syntax based but is only a smart search and replace mechanism, so it might pick up a word within a comment (which might actually be what you want, if the comment refers to a local variable you're modifying). This is an example:



```
procedure TForm1.FormCreate(Sender: TObject);
var
  i: Integer;
begin
  i := 20;
end;
```

Sync Edit is generally used to modify the names of local variables within method or other relatively small blocks. For replacing symbols at a larger scale I strongly suggest you to use *Rename Refactoring*, discussed in Chapter 2.

Editor History

In the past, Delphi kept a single backup copy of the previous version of each source code file. This meant that by saving a modified file twice you'd lose the previous version of the code, unless you had a backup copy or a source

code versioning system. Now Delphi has a rather flexible local history of each file. You determine how many versions to keep (the default is 10) and Delphi saves them in a hidden `__history` subfolder under the folder hosting the source code file.

You can control this setting using the "Create backup files" check box and the "File backup limit" edit box of the main Editor Options page of the Options dialog box. Delphi creates a physical file for each version, naming them like this:

```
| MyForm.pas..~1~
| MyForm.pas..~2~
```

You can reopen these files with any (external) editor or browse them from the History page of the IDE. This pane has three further sub panes:

- The *Contents* page shows a combo box with each file (PAS and designer files) and, for the selected files, a list of versions in the main window. The selected version is displayed in a read-only viewer (it is not an editor so you cannot update the history file). Notice that the list shows both local versions and versions stored in a Version Control System, if you've installed one and are using it for the project files.
- The *Info* page is mostly to interact with a Version Control System, as it shows a description of the version and other information not relevant to local backup files.
- The *Diff* page allows you to pick any two versions (including an editor buffer still not saved to a file) and compare them with a simple differences view. Both here and in the *Contents* page you can copy lines from an older version and paste them into the edit buffer.

Notice also that this local versioning includes designer files, like DFM files for VCL applications. In this pane you can reactivate an older version of a file with the *Revert* command. As designer files (like DFM files) have almost no Undo capability, you can use the History to actually Undo unwanted changes to a DFM file.

When I need to compare two Delphi units, including the local backups produced by Delphi, I invariably use *Beyond Compare*:

| <http://www.scootersoftware.com>

This is a superb, low-cost file comparison utility written in Delphi and capable of understanding even binary DFM files. A Delphi-specific version of

this tool was made available as a free download to Delphi 2006 users. Furthermore, there is an integration package you can find at:

| http://www.jed-software.com/docs/bc_readme.htm

Other (Recent) Editor Features

There are many other brand new features in the editor worth mentioning briefly, in case you haven't noticed them. Here is a summary:

- **Matching parentheses.** As you type or move around the code, the editor shows matching parentheses and quotes. This long-awaited feature is a time saver when you have code with many nested parentheses.
- **Macros.** The macro toolbar at the left of the status bar at the bottom of the editor can be used to record and replay identical sequence of keystrokes. This feature was already available in the past with the corresponding shortcut keys (Ctrl+Shift+R for recording, Ctrl+Shift+P for play). Editor macros can be useful for performing multistep operations over and over again, such as reformatting source code or arranging data more legibly in the source code.
- **Persistent Bookmarks.** The editor now saves the bookmarks you set along with other project desktop settings. This means the bookmarks persist between editing sessions, unlike in past versions of Delphi. By the way, bookmarks are created using Ctrl+Shift plus a number key from 0 to 9, as you'll see in a "gutter" margin on the side of the editor. To jump back to the bookmark, press the Ctrl key plus the number key. To see a list of active bookmarks you have to install a third-party plug in like my own Cantools Wizard (see Chapter 11).
- **Toggle Comments.** One of my favorite new features is the Toggle Comment command, which is used to add or remove the // comment from the current line of source code or all of the selected source code lines at once. This feature is activated by pressing the Ctrl key plus the *physical* '/' key. By *physical*, I mean the low-level keyboard key, which generally changes on foreign keyboards. For example, on my Italian keyboard I have to use Ctrl+ù. The context

menu of the editor shows the actual shortcut key on the side of the Toggle Comment menu item.

- **Close all Other Pages.** If you right click on a tab of the editor, you can use the new command "Close all other pages".
- **Click and Close.** You can middle-click (or click the mouse wheel) on a tab of the editor to close it.
- **Class Navigation.** You can move from one method to the next one (or the previous one) using the key combination Ctrl+Alt+Down Arrow and Ctrl+Alt+Up Arrow. Ctrl+Alt+Home and Ctrl+Alt+End bring you to the first and last method. This is called Class Navigation. You can also restrict moving within the current class by activating the Class Lock, press Ctrl+Q and then L (keeping Ctrl pressed). (I have to say I don't find this feature particularly useful, compared to the classic navigation with Ctrl+Shift+Arrow to jump back and forth from the definition to the declaration of a method).

The VCL Form Designer

Beside the editor, the other core window Delphi developers typically use a lot is the designer, which comes in several flavors as form designer, data module designer, frame designer, web module designer, and many more.

As soon as you place a new control on the VCL form designer (or the frame designer), you'll immediately see something new. The corners of the control and the central portion of each side have a small light blue circle. These mark the resize areas of the control: moving the mouse over them you can resize it (like in the past).

Now if you place a second control in the form, you'll be able to see a more useful new feature, "Design Guidelines". This is a visual aid to properly aligning controls on a form:



Not only you can align the sides of a control with those of another one, but you can even align the text baseline, as in the example above. Controls automatically snap to the guidelines when they are close to them. They also snap when they are at a given margin to the border of the container control.

Another new feature is the “Form Positioner” in the bottom right corner of the designer surface:



There you can see, in small, the position of the form on the screen, which is useful if the form uses absolute positioning. You can also use the Form Positioner to modify the Top and Left properties of the form visually. This tool can only be used when you use the default embedded form designer. A little known feature is that you can click the Form Positioner to get (temporarily) a bigger view and position the form more precisely.

Text Baseline for Components Developers

Working with the Design Guidelines for the borders of the controls is quite obvious. But how does the designer know about the text baseline for a control? Of course, it doesn't: You need to provide a `TComponentGuidelines` class and register it with the `RegisterComponentGuidelines` function. You can find more details in the `DesignEditors` and `VCLEditors` units of the `ToolsAPI VCL` source code folder.

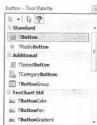
There is a `gtBaseline` value in the `TDesignerGuideType` enumeration, but it is not really referenced in the code, while the class method `TControlGuidelines.GetTextBaseline` seems to provide a default implementation. It looks like you can fully customize the behavior, but having a ready-to-use example would make this easier.¹⁴

¹⁴ I know; I could have asked CodeGear for more information...

Searching in the Tool Palette

The Tool Palette that replaces the Component Palette can be configured in many ways, which you can change by using the Properties item of the Palette shortcut menu. Another key difference is that the Tool Palette is context sensitive and shows either the list of components (when a designer is active) or the list of the New Items options (when the editor is active).

What is nice is that you can select a new component and add it to the current form (or create a new unit or form) without having to use the mouse but activating the Tool Palette by pressing Ctrl+Alt+P and type the component item name, eventually move over it with the arrow keys, and press the Enter key to select it. While in Delphi 2005 and 2006 the selection was for the initial portion of the text (without considering the initial "T"), the selection is now against any occurrence of the typed text within the component or item name. For example, if you type "button" you get:



Making Help Insight Work

Another recent feature of the editor is Help Insight. This shows a summary help window as you move the mouse over a symbol or select an entry in the code completion window. If there is no information available for the symbol, the program shows a summary of the symbol definition (where it is defined,

its type itself, the method parameters and return value, and so on), like the following parameterless method:

```
40 procedure TForm1.Button1Click(Sender: TObject);
    begin
      Label1.Caption := 'Hello';
      CustomMethod;
43   TForm1.CustomMethod Method - HelpInsightDemoForm.pas \(23,15\)
    end
      Declared in HelpInsightDemoForm.TForm1
```

The Help Insight window is a small version of Internet Explorer. Its output depends on the `HelpInsight.xsl` XSL transformation file used to turn the core XML data into proper HTML and the `HelpInsight.css` file used to determine the color and fonts of the display. Both files are stored in the `ObjRepos` folder of Delphi (again by default `CodeGear\RAD Studio\5.0\ObjRepos`). If you want to customize the user interface of Help Insight, you can change these two files.

Generate XML Documentation

What is interesting, though, is that Help Insight can be customized using the Generate XML Documentation feature of the compiler. This forces the compiler to create an XML file for each unit (and the project source code) with a detailed list of internal symbols. The XML file includes all the fields and methods, even the private and protected ones.

This is an excerpt (with lengthy base form class information removed):

```
<?xml version="1.0" encoding="utf-8"?>
<namespace name="HelpInsightDemoForm">
  <class name="TForm1">
    <ancestor name="TForm" namespace="Forms">
      ... omitted
    </ancestor>
    <members>
      <field name="Label1" type="TLabel"
        visibility="published"/>
      <field name="Edit1" type="TEdit"
        visibility="published" />
      <field name="Button1" type="TButton"
        visibility="published"/>
```

```

    <procedure name="ButtonClick"
      visibility="published">
      <parameters>
        <parameter name="Sender" type="TObject"/>
      </parameters>
    </procedure>
    <procedure name="CustomMethod"
      visibility="public"/>
  </members>
</class>
<variable name="Form1" type="TForm1"/>
</namespace>

```

Now you can add custom comments to the XML Documentation by using the "triple slash" syntax `///` or the multi line versions: `{!}` and `{*!}`. For example, you can write:

```

public
  /// This is a custom method, of course
  procedure CustomMethod;

```

The information is added to the XML output, as follows:

```

<procedure name="CustomMethod" visibility="public">
  <devnotes>
    This is a custom method, of course
  </devnotes>
</procedure>

```

Now, if you provide a **summary** section within the comment, this will show up in the Help Insight window:

```

public
  /// <summary>This is a custom method, of course</summary>
  procedure CustomMethod;

```

The Help Insight for the method now becomes:

```

40 procedure TForm1.Button1Click(Sender: TObject);
41 begin
42   Label1.Caption := 'Hello';
43   CustomMethod;
44   TForm1.CustomMethod Method - HelpInsightForm.pas (23,15)
45 end
46   This is a custom method, of course
47   Declared in HelpInsightForm.TForm1

```

If you get into the habit of using this commenting syntax, you'll immediately get information about the symbols you refer to while typing the code. Help Insight is, in fact, also triggered by the Code Completion window:



To help with this syntax, you can benefit from using a custom Code Template, like the one described in the section “Template for Help Insight Comments” of Chapter 2.

Notice that you don't need to recompile the program to be able to see the comments in the designer, as they immediately show up as soon as you type them. As a ready-to-use example, with several custom Help Insight comments, you can refer to the HelpInsightDemo example.

Other Features of the Delphi IDE

There are many other new features in the Delphi IDE worth mentioning, beside the editor and the designer.

Extended Search Results

The Messages pane, generally placed at the bottom of the IDE, displays both compiler messages and search results. Even if simple to use, this window has several improvements over past versions of Delphi. Since Delphi 7 you have been able to display search results in a different tab so they do not interfere with compiler messages as they did in the past. Second, every time you do a different search you can request that Delphi show the results in a different page, so the results of previous search operations remain available.

Since Delphi 2005 you can also use a check box in the Find Text dialog box to group the search results by source code file, which often results in more manageable search results if there are many hits. You can press the Alt+Page Down and Alt+Page Up key combinations to cycle through the tabs of this window. (The same commands work for some other tabbed views, but not all of them.)

File Browser View

The IDE in Delphi 2007 has a new window, called File Browser, which hosts a miniature version of Windows Explorer that allows you to work on the file system within the IDE.

It is not kept in sync with project folders or files in the editor (might be a nice extension for somebody to write). It supports any Explorer extension, including (for example) TortoiseCVS and TortoiseSVN¹⁵.

Another feature is that you can provide a filter to limit the files actually displayed in the File Browser. Use wild cards separated by semicolons, like in:

```
| *.pas;*.dpr;*.dproj
```

Database Management Tools

Since Delphi 2005, the IDE integrates a Data Explorer window that provides limited access to database data and metadata through the dbExpress library. In Delphi 2005 this was limited to the .NET Borland Data Providers, in Delphi 2006 it offered both, while Delphi 2007 supports only the Win32 database access technology, dbExpress.

The integrated database management tool of the Delphi IDE allows you to see the configured connections, test them, get a list of the tables for each

¹⁵ TortoiseCVS and TortoiseSVN are two plugin tools for Windows Explorer, providing support for CVS and Subversion, two popular open-source version control systems. They can be found at <http://www.tortoisecvs.org/> and at <http://tortoisesvn.tigris.org>, respectively.

database, and the list of fields and indexes of each table:



Unlike the BDP drivers in Delphi 2006, with the dbExpress library you have access to metadata but cannot modify it¹⁶. However, you can still execute an SQL query by typing it directly into a custom pane.

(Limited) ALM Support in the IDE

Contrary to Delphi 2006, the new version offers no direct Borland StarTeam¹⁷ integration. Other ALM-related tools like the Together UML designer and the Unit Test integration (provided by DUnit) are still part of the product. I'll cover DUnit briefly in Chapter 8.

The New Help System

Compared to the classic versions of Delphi, since Delphi 2005 the product comes with a help file in the Microsoft Document Explorer format. The old

¹⁶ **Last Minute Update:** In Delphi 2007 Release 2 (part of RAD Studio 2007) you can modify the dbExpress metadata in Data Explorer. This tool features also a very nice Visual Query Builder.

¹⁷ For more information see the StarTeam page on <http://www.borland.com>.

WinHelp format has been deprecated by Microsoft and is not supported at all on Windows Vista.

The advantage is that a single help system can include many more documents, so now both the Windows Platform SDK (the API documentation) and the Delphi help are connected and can be searched directly. The drawback to this large integrated documentation is that you'll often get many references to topics you are totally not interested in, from Windows scripting languages to the .NET platform help¹⁸.

If you are interested in learning about how CodeGear creates the help system and how to report errors, you might want to check Chris Bensen's blog, at:

| <http://chrisbensen.blogspot.com>.

In particular, you might want to search for a four-part post titled "Documentation/Help In Delphi 2007".

¹⁸ If you don't care about the .NET help, you can remove it (reducing extraneous search results) as suggested by Fernando Madruga in his blog at: <http://memmyselfanddelphi.blogspot.com/2007/04/delphi-2007-for-win32-getting-rid-of.html>.