Chapter 1

Introduction

The *Developer's Guide* describes intermediate and advanced development topics, such as building client/server database applications, writing custom components, creating Internet Web server applications, and including support for industry-standard specifications such as TCP/IP, OLE, and ActiveX. The *Developer's Guide* assumes you are familiar with using C++Builder and understand fundamental C++Builder programming techniques. For an introduction to C++Builder programming and the integrated development environment (IDE), see the *Quick Start* and the online Help.

What's in this manual?

This manual contains five parts, as follows:

- **Part I, "Programming with C++Builder,"** describes how to build general-purpose C++Builder applications. This part provides details on programming techniques you can use in any C++Builder application. For example, it describes how to use common Visual Component Library (VCL) objects that make user interface programming easy such as handling strings, manipulating text, implementing the Windows common dialog, toolbars, and cool bars. It also includes chapters on working with graphics, error and exception handling, using DLLs, OLE automation, and writing international applications.

  Generally, it rarely matters that C++Builder's underlying VCL is written in Object Pascal. However, there are a few instances where it affects your C++Builder programs. A chapter on C++ language support and the VCL details such language issues as how C++ class instantiation differs when using VCL classes and the C++ language extensions added to support the C++Builder "component-property-event" model of programming.

  The chapter on deployment details the tasks involved in deploying your application to your application users. For example, it includes information on effective compiler options, using InstallShield Express, licensing issues, and how to determine which packages, DLLs, and other libraries to use when building the production-quality version of your application.

- **Part II, "Developing database applications,"** describes how to build database applications using database tools and components. C++Builder lets you access many types of databases. With the forms and reports you create, you can access local databases such as Paradox and dBASE, network SQL server databases like InterBase and Sybase, and any data source accessible through open database connectivity (ODBC) or ActiveX Data Objects (ADO).
- **Part III, "Writing distributed applications,"** describes how to create Web server applications as CGI applications or dynamic-link libraries (DLLs). C++Builder provides Internet-specific components that make it easy to handle events associated with a specific Uniform Resource Identifier (URI) and to programmatically construct HTML documents.

  This part also provides a chapter on the C++Builder socket components that let you create applications that can communicate with other systems using TCP/IP and related protocols. Sockets provide connections based on the TCP/IP protocol, but are sufficiently general to work with related protocols such as Xerox Network System (XNS), Digital's DECnet, or Novell's IPX/SPX family.
- **Part IV, "Developing COM-based applications,"** describes how to build applications that can interoperate with other COM-based API objects. C++Builder supports COM applications that are based on the Active Template Library (ATL). Wizards and a Type Library editor ease the development of COM servers, and an importing tool lets you quickly create client applications. Support for COM clients is available in all editions of C++Builder. To create COM servers, you need the Professional or Enterprise edition.
- **Part V, "Creating custom components,"** describes how to design and implement your own components, and how to make them available on the Component palette of the IDE. A component can be almost any program element that you want to manipulate at design time. Implementing custom components entails deriving a new class from an existing class type in the VCL class library.

Manual conventions

This manual uses the typefaces and symbols described in Table 1.1 to indicate special text.

Table 1.1 Typefaces and symbols

Typeface or symbol	Meaning
`Monospace type`	Monospaced text represents text as it appears on screen or in C++ code. It also represents anything you must type.
[ ]	Square brackets in text or syntax listings enclose optional items. Text of this sort should not be typed verbatim.
Boldface	Boldfaced words in text or code listings represent C++ reserved words or compiler options.

Table 1.1 Typefaces and symbols (continued)

Typeface or symbol	Meaning
Italics	Italicized words in text represent C++ identifiers, such as variable or type names. Italics are also used to emphasize certain words, such as new terms.
Keycaps	This typeface indicates a key on your keyboard. For example, "Press Esc to exit a menu."

Contacting developer support

Inprise offers a variety of support options. These include free services on the Internet, where you can search our extensive information base and connect with other users of Borland products. In addition, you can choose from several categories of support, ranging from support on installation of the Borland product to fee-based consultant-level support and detailed assistance.

For more information about Inprise's developer support services, please see our Web site at http://www.borland.com/devsupport, call Borland Assist at (800) 523-7070, or contact our Sales Department at (831) 431-1064. For customers outside of the United States of America, see our web site at http://www.borland.com/bww/intlcust.html.

When contacting support, be prepared to provide complete information about your environment, the version of the product you are using, and a detailed description of the problem.

For information about year 2000 issues and our products, see the following URL: http://www.borland.com/about/y2000/.