

Designed for
Microsoft®
Windows NT®
Windows 98



CD-ROM
Included

Microsoft® 程序设计系列



内附
配套光盘

Microsoft® Visual J++ 6.0™

程序开发环境

掌握使用 Visual J++ Windows
基类进行 Windows 快速开发的
技术

北京大学出版社

Microsoft Press

11-51
11-335
Microsoft Visual J++ 6.0

程序开发环境

[美]Shannon Dunn 著

北京博彦科技发展有限公司 译

北 京 大 学 出 版 社

· 北 京 ·

著作权合同登记 图字：01-1999-1918 号

Microsoft Visual J++ 6.0 Developer's Workshop

Shannon Dunn

TS182/31

本书版权为 Shannon Dunn 所有, 1998。(Copyright © 1998 by Shannon Dunn. All rights reserved.)

本书中文版由美国 Microsoft 出版社授权北京大学出版社独家出版, 2000。

本书封面贴有北京大学出版社的激光防伪标签, 无标签者不得销售。

版权所有, 翻印必究。

图书在版编目(CIP)数据

Microsoft Visual J++ 6.0 程序开发环境/[美]Shannon Dunn 著;北京博彦科技发展有限公司译. —北京: 北京大学出版社, 2000.7

ISBN 7-301-04183-7

I. M… II. ①大… ②北… III. JAVA 语言-程序设计 IV. TP312

中国版本图书馆 CIP 数据核字(2000)第 63150 号

书 名: Microsoft Visual J++ 6.0 程序开发环境

著作责任者: [美]Shannon Dunn 著 北京博彦科技发展有限公司 译

责任编辑: 邱淑清 徐 谦 章景涛

标准书号: ISBN 7-301-04183-7/TP·459

出版者: 北京大学出版社

地 址: 北京市海淀区中关村北京大学校内 100871

网 址: <http://cbs.pku.edu.cn>

电 话: 出版部 62752015 发行部 62754140 编辑室 62757065

电子信箱: zpup@pup.pku.edu.cn

排 印 者: 北京大学印刷厂

发 行 者: 北京大学出版社

经 销 者: 新华书店

787 毫米×1092 毫米 16 开本 26.75 印张 595 千字

2000 年 7 月第 1 版 2000 年 7 月第 1 次印刷

定 价: 67.00 元(含光盘)

致谢

如此众多的帮助我完成了这部书,有些帮助是直接的,有些是间接的。如果没有以下谈到的每个为热情支持,这本书现在就不会在您手中。

首先将我的感谢送给 Chris Anderson 和 Brian Pepin,感谢他们丰富的知识、深刻的理解力和对我问的耐心解答。我要感谢 Microsoft Visual J++ 用户教育小组的成员们能容忍我在本写作期间每天早晨连连的哈欠和朦胧的睡眼。许多的掌声和赞誉都应该送给 Visual J. 开发小组,他们完成的这个产品真是令人激动!同时我还因能与 Microsoft 出版社的一个很优秀的小组一起工作,才保证本书能提供最好和最清晰的内容。赞誉还应归于好的诸位:Eric Stroh、Sally Stickney、Victoria Thulman、Chris Hotchkiss、Donnie Cameron、Pela Buitrago、Barb Runyan、Stuart Greenman,以及 Joel Panchot。

我在工作中遇过的许多优秀人士,他们给予我很多的帮助,使我拥有目前的丰富经验,成为面面俱到的软件开发人员。在此我要特别感谢 Roland Houle、Becky Himmelstein、Juan Cordero,以及 Tom Griggs,他们都热情地帮助和支持过我。

同时我也要感谢我的父母。感谢妈妈做了如此出色的一位 Microsoft 福音传教士,也感谢爸爸在很早以前就鼓励我从事写作。最为重要的,我要感谢我现在家庭中的每位成员:我妻子 Susar 和我们的 4 个儿子 Patrick、Andrew、Alex 和 Jonathan(他们的名字和形象点缀在全书的多个示例中),他们是我生活的一大部分,也是促使我写作并完成此书的主要动力。他们一直非常地鼓舞着我和我的工作。

如果我忘记提及某一位,那么就请您多包涵吧!

引言

如果您建造高楼大厦的话,最好建造一座城堡而不是一间纸板房。

G. C. Lichtenberg (1742—1799)^①

在计划 and 设计应用程序时,有许多问题需要仔细考虑。明确目标客户是谁,详细了解用户硬件条件的限制和整个系统配置的要求,可以使您尽早弄清楚一个应用程序应该是什么样子。开发应用程序的最重要的决定应当是——这一点往往颇受争议——采用什么编程语言。面向对象的能力、可获得组件模型支持、易于使用、广泛的适用性等特点,常常是选择“正确”的开发语言时最需要考虑的。当您拨开所有这些迷雾,回到您需要什么的原始问题时,您将发觉开发语言的确是最关键的因素,它必须能够胜任工作!

Java 是一种具有很大潜力的奇妙的编程语言,几乎对于任何应用开发,它都能够满足其需求。Java 具有这样一些特征,如垃圾收集、干净的多线程、单一继承性,以及抛弃指针,在不损失性能的情况下避免在 Microsoft Visual C++ 中常遇到的复杂性方面的问题等。Java 的能力和稳定性使它成为一种适应性很强的语言,而且任何有其他语言经验的开发人员都会发觉转入 Java 相当容易。但是归根结底,Java 只是一种编程语言而已。Java 最初被设想成一种您可以用来编写通用于任何计算机系统的应用程序的语言。实际上,正是这一简单的动机成为开发这一语言的催化剂。

但是 Java 作为一个平台的想法却在“现实世界”里失败了。Java 的设计者假设每个生产商都会设计一个 Java“虚拟机”(Virtual Machine, VM),以在每台计算机上完全一致地运行 Java 应用程序。但是,绝大多数生产虚拟机的公司同时也是硬件或操作系统制造商,他们必须保证其虚拟机能够提供他们的客户期望达到的系统性能。这样,一个个

^① G. C. Lichtenberg (1742—1799), 德国物理学家、哲学家。格言集“Notebook F,”格言 39 (写于 1765—1799, 译者 R. J. Hollingdale, 1990 年)。《哥伦比亚格言辞典(Columbia Dictionary of Quotations)》由哥伦比亚大学出版社出版。版权为哥伦比亚大学出版社所有, 1993 年。

虚拟机的运行就有些互不相同了。正因为如此,Java 开发人员不得预先决定他或她写的程序将要运行于哪种虚拟机上。这使得“一次编写,到处运行”目标成为了泡影。

更为要命的是,在 Java 中所有开发人员都被限定使用一组相的 API。如果您在这组 API 以外编写程序,或者开发一些附加的接口,使之在您的操作系统上能够更快地运行,那么您就会被认为“不纯”。这种要求每个人以相同的方式用相同的 API 的“专制”发展模式,是大多数开发人员都不愿意接受的。开发人员需要的是提供客户想要的功能。如果一种功能不被 Java API 所支持,那么开发人员就不必改用其他的 API 或 VM 来满足客户的需要。“专制”主义作为一种政府形式未能成功,也肯定不会在计算机的领域里获得成功。在这一领域,一件事有不同的发展方向认为是必然的,而且这样做也有它的好处。

Microsoft 关于 Java 的观点是认为 Java 是一种很优秀的编程语言,可以用来开发世界级的 Microsoft Windows 应用程序。编写 applet 是个好主意,开发人员觉得用 Java 创建 Windows 应用程序更为重要。Microsoft 没有像其他一些开发工具供应商那样遵循“一次编写,到处运行”的教条,而是决定让 Visual J++ 6.0 和新的“快速应用程序开发”(Rapid Application Development, RAD),以及“Windows 基类”(Windows Foundation Classes, WFC)框架一起,成为使开发人员能够应用 Java 语言创建优秀的 Windows 应用程序、充分发挥 Windows 系统功能的高效率工具。您当然可以用 Visual J++ 6.0 编写 applet,但更为重要的是,您现在可以全面发挥 Java 强大的功能,应用于开发世界级的 Windows 应用程序。

《Microsoft Visual J++ 6.0 程序开发环境》这本书以示例学习的方式讨论了 Visual J++ 和 WFC 开发的各个主要方面。每章都循序渐进地列举了一些例子来帮助读者全面掌握 Visual J++ Windows 应用程序设计的方法。在各章中讨论的内容主要包括设计窗体、通过 COM 展示对象、使用数据库,以及创建 WFC 控件等。

本书大部分章节中的内容都是自成一体的,可以单独拿来阅读,这使您能在单独一章中得到您想要的东西。当然您也可以通过通读本书来全面准确地掌握 WFC 的各个方面的。许多章节中的示例程序在后面的章节中得到了进一步的应用和改进。第十章总结了全书的内容,并应用前面章节的示例程序代码创建了一个相当庞大和健壮的应用程序。不管您以何种方式使用本书,《Microsoft Visual J++ 6.0 程序开发环境》都将及时地为您提供学习 Visual J++ 和 WFC 的信息,而且会使开发 Windows 应用程序更加容易,功能更加丰富。

读者对象

本书的大部分内容适合于有一定的使用 Visual Basic 或 Visual C++ 语言开发 Windows 应用程序经验的读者。书中的讨论和示例并不涵盖 Windows 应用程序开发的许多细节,

所以具有这方面的经验会对您有很大的帮助。尽管如此,如果您是 Windows 应用程序开发的新手,那也不要紧,具有强大功能的 Visual J++ 和 RAD 一起可以使您很容易地开始编写 Windows 应用程序。本书将帮助您理解并掌握如何使用 Visual J++ 和 WFC 框架。在一更高级的领域,例如 COM 和数据库应用程序开发,我并没有提供许多可能对读者有用的理论性内容,所以读者也许需要阅读其他的一些资料。通常我都会指出一些材料以供参考。

如果您是 Java applet 开发人员,没有什么编写 Windows 应用程序的经验,但希望通过学习 Windows 编程来增强您的 Java 编程经验……非常欢迎!您将学会如何使用 Visual J++,并了解 WFC 框架究竟有何作用。书中的例子会对这些问题提供充分的说明,特别是如果您拥有一定的“用户界面”(UI)设计经验的话。

对那些依然坚持开发 Java applet 或是信奉 Java“一次编写,到处运行”原则的人,这本书将不会有什么用处,因为本书的中心内容是用 Visual J++ 开发优秀的 Windows 应用程序。我没有涉及 Java applet 编程的内容,或者“纯粹 Java”思想和“Java 异教讨伐运动”。当然,您也许会企图通过阅读书中的某些章节来弄清楚 Windows 应用程序开发究竟可以给您的客户提供什么样的东西!

使用本书的要求

Visual J++ 6.0 有三种版本:标准版、专业版和企业版(企业版只与 Microsoft Visual Studio Enterprise Edition 一起发售)。要使用本书,您需要以上三种版本之一。旧版本的 Visual J++ 并不支持 WFC 应用程序框架或使用 Java 开发 Windows 应用程序。如果要创建 WFC 控件和 COM 对象,或者进行数据库操作,那么您需要有专业版或企业版。本书附带的光盘包含了书中所有示例程序的源代码。请在将这些源代码文件安装到硬盘之前先阅读光盘根目录下的 Readme.txt 文件。Visual J++ 6.0 的 90 天专业试验版也包含在本书附带的光盘中。您所需要的只是学会编写这些漂亮的应用程序的决心。

欢迎提出意见

如果您有任何评论、建议或批评(衷心欢迎),或者关于本书的任何问题,那么作者非常希望能与您交流。本书中的示例程序是在 Visual J++ 6.0 开发期间完成的,所以其代码中有些部分可能因为系统功能的增强而显得不是十分必要或者不那么有效。欢迎您对此提出建议,以便在本书以后的版本中改进这些程序。您可以通过以下的电子邮件地址与作者联系: sdunn1@mci2000.com。

目录

致谢	1
引言	3
第一章 Visual J++ 6.0 的新特性	1
1.1 集成开发环境	1
1.2 Windows 基类(WFC)	8
1.3 Internet 支持	8
1.4 COM 支持	8
1.5 向导和生成器	9
1.6 数据访问	9
1.7 打包和部署	9
1.8 可扩展性	10
1.9 总结	10
第二章 Windows 基类	11
2.1 Java 是一种编程语言!	11
2.2 Windows 基类节约时间!	12
2.2.1 超越 J/Direct	12
2.2.2 WFC: 不仅仅是一个框架	12
2.3 WFC 数据包	13
2.3.1 com.ms.wfc.app	14
2.3.2 com.ms.wfc.core	14
2.3.3 com.ms.wfc.data 和 com.ms.wfc.data.ui	14
2.3.4 com.ms.wfc.html	15
2.3.5 com.ms.wfc.io	15
2.3.6 com.ms.wfc.ui	15

2.3.7	com.ms.wfc.util	16
2.3.8	数据包小结	16
2.4	第一个 WFC 应用程序:“Hello WFC”	17
2.4.1	步骤 1: 创建项目和窗体	17
2.4.2	步骤 2: 添加显示的第一行文本	18
2.4.3	步骤 3: 添加显示的第二行文本	20
2.5	总结	23

第三章 窗体 25

3.1	理解窗体的用途	26
3.2	创建和显示窗体	26
3.2.1	步骤 1: 创建项目和应用程序的主窗体	27
3.2.2	步骤 2: 创建启动屏幕窗体	27
3.2.3	步骤 3: 将代码添加到启动屏幕的窗体中	29
3.2.4	步骤 4: 添加启动屏幕窗体的方法和事件处理程序	32
3.2.5	步骤 5: 编写代码以显示启动屏幕	37
3.3	创建屏幕保护程序	38
3.3.1	步骤 1: 创建项目及其主窗体	39
3.3.2	步骤 2: 将成员变量添加到屏幕保护程序窗体中	40
3.3.3	步骤 3: 将功能添加到屏幕保护程序中,第 1 部分	41
3.3.4	步骤 4: 将功能添加到屏幕保护程序中,第 2 部分	43
3.3.5	步骤 5: 编译、打包及部署屏幕保护程序	47
3.4	创建对话框	48
3.4.1	步骤 1: 将 DWMarquee 项目移动到新的项目中	49
3.4.2	步骤 2: 将屏幕保护程序选项窗体添加到项目中	49
3.4.3	步骤 3: 为 ScreenSaverOptions 类创建事件处理程序	51
3.4.4	步骤 4: 将注册表方法添加到 ScreenSaverOptions 类中	53
3.4.5	步骤 5: 更新 ScreenSaver.java 以支持选项	56
3.5	创建一个可持续窗体类	60
3.5.1	步骤 1: 创建项目和窗体	60
3.5.2	步骤 2: 在注册表中保存窗体的状态	61
3.5.3	步骤 3: 从注册表中加载窗体的状态信息	62
3.5.4	步骤 4(可选): 将可持续窗体类添加到窗体模板列表中	63
3.6	子类化窗体	67
3.6.1	步骤 1: 创建基窗体	67
3.6.2	步骤 2: 创建子类化的窗体	69
3.7	创建工具窗口	73
3.7.1	步骤 1: 创建项目和工具窗口	74

3.7.2 步骤 2: 创建应用程序的父窗体	75
3.7.3 步骤 3: 添加调用工具窗口的代码	76
3.7.4 步骤 4: 将事件处理程序和成员变量添加到工具窗口中	77
3.7.5 步骤 5: 将功能代码添加到工具窗口中	78
3.8 总结	81
第四章 窗口附件	83
4.1 创建菜单	84
4.1.1 步骤 1: 创建项目和应用程序主窗体	84
4.1.2 步骤 2: 创建应用程序菜单	86
4.1.3 步骤 3(可选): 将启动屏幕添加到应用程序中	91
4.2 创建菜单事件处理程序	91
4.2.1 步骤 1: 添加 WFC Jot 的菜单事件处理程序	92
4.2.2 步骤 2: 将支持代码添加到 File 菜单的事件处理程序中	93
4.2.3 步骤 3: 添加 Edit 菜单事件处理程序的支持代码	99
4.2.4 步骤 4: 添加其他菜单事件处理程序的支持代码	102
4.3 创建环境菜单	104
4.3.1 步骤 1: 将 ContextMenu 控件添加到项目中	105
4.3.2 步骤 2: 为环境菜单分配菜单项目	105
4.3.3 步骤 3: 添加应用程序的 About 框	110
4.3.4 步骤 4(可选): 添加 Find And Replace 支持	112
4.4 创建工具栏	113
4.4.1 步骤 1: 在新项目中打开 WFC Jot 应用程序	114
4.4.2 步骤 2: 添加 ImageList 控件及其图像	114
4.4.3 步骤 3: 添加 ToolBar 控件及其按钮	115
4.4.4 步骤 4: 处理工具栏的事件	116
4.5 创建状态栏	119
4.5.1 步骤 1: 在新项目中打开 WFC Jot 应用程序	120
4.5.2 步骤 2: 添加 StatusBar 控件并创建其窗格	120
4.5.3 步骤 3: 添加支持状态栏的代码	121
4.6 总结	123
第五章 控件	125
5.1 什么是控件?	125
5.2 使用 Windows 基本控件	126
5.2.1 步骤 1: 创建项目并添加窗体	127
5.2.2 步骤 2: 将控件添加到 General Information 选项卡中	129

5.2.3 步骤 3: 将控件添加到 Player Skills 选项卡中	131
5.2.4 步骤 4: 添加支持代码	136
5.3 将帮助添加到应用程序中	145
5.3.1 步骤 1: 打开 HockeyPlayerScout 应用程序并添加帮助控件	146
5.3.2 步骤 2: 添加 Help 按钮的事件处理程序及其代码	147
5.3.3 步骤 3: 添加支持 F1 键和“What’s This”帮助的代码	147
5.4 创建可滚动的 PictureBox 控件	150
5.4.1 步骤 1: 创建项目及其窗体	150
5.4.2 步骤 2: 修改构造函数	151
5.4.3 步骤 3: 添加滚动条的事件处理程序及其支持代码	152
5.5 使用 TreeView、ListView 和 ImageList 控件	154
5.5.1 步骤 1: 创建项目并添加新的窗体	155
5.5.2 步骤 2: 将 TreeView、ListView、ImageList 及其他控件添加到窗体中	155
5.5.3 步骤 3: 使用 J/Direct Call Builder 添加方法	159
5.5.4 步骤 4: 将支持代码添加到应用程序中	159
5.6 将拖放支持添加到应用程序中	169
5.6.1 步骤 1: 创建项目及其窗体	170
5.6.2 步骤 2: 创建窗体控件的事件处理程序	171
5.6.3 步骤 3: 添加拖放支持代码	171
5.7 将 ActiveX 控件导入到 WFC 应用程序中	174
5.7.1 步骤 1: 创建项目并设计窗体	175
5.7.2 步骤 2: 添加 ActiveX 控件	176
5.7.3 步骤 3: 添加事件处理程序及其代码	178
5.8 总结	180
第六章 MDI 应用程序	181
6.1 什么是 MDI?	181
6.2 创建 MDI 应用程序	182
6.2.1 步骤 1: 创建项目和 MDI 容器窗体	182
6.2.2 步骤 2: 创建子窗体	185
6.2.3 步骤 3: 打开文件并处理 MRU 文件列表	185
6.2.4 步骤 4: 添加 About 框	192
6.3 检索和存储 MDI 应用程序数据	194
6.3.1 步骤 1: 创建项目和 MDI 容器窗体	194
6.3.2 步骤 2: 创建子窗体	196
6.3.3 步骤 3: 将代码添加到 MDI 容器窗体中	199
6.3.4 步骤 4: 将支持代码添加到子窗体中	207
6.4 从 SDI 应用程序创建 MDI 应用程序	217

6.4.1 步骤 1: 创建项目	218
6.4.2 步骤 2: 将事件处理程序移动到 MDI 窗体中	218
6.4.3 步骤 3: 调整子窗体中的代码	226
6.5 总结	232
第七章 数据库	233
7.1 早期的数据库	233
7.2 ADO 就是答案!	233
7.3 使用 Data Form Wizard	234
7.3.1 步骤 1: 创建项目并显示 Data Form Wizard	235
7.3.2 步骤 2: 指定数据库类型及其名称	235
7.3.3 步骤 3: 指定窗体类型	237
7.3.4 步骤 4: 选择 Master 和 Detail 记录源及字段	238
7.3.5 步骤 5: 选择记录源关系及控件	240
7.3.6 步骤 6: 完成向导, 检查窗体并运行示例	241
7.4 将控件绑定到数据库中	243
7.4.1 步骤 1: 创建项目并添加窗体	244
7.4.2 步骤 2: 将控件添加到窗体中	244
7.4.3 步骤 3: 将 DataSource 控件与数据库联系起来	247
7.4.4 步骤 4: 将控件绑定到数据库中	249
7.4.5 步骤 5: 为按钮添加事件处理程序和代码	250
7.5 访问没有数据绑定的数据	254
7.5.1 步骤 1: 打开 HockeyScout 项目并准备代码	255
7.5.2 步骤 2: 添加 DataSource 控件	256
7.5.3 步骤 3: 创建 PlayerSelect 对话框	256
7.5.4 步骤 4: 添加访问数据库的代码	258
7.6 使用 DataGrid 控件	269
7.6.1 步骤 1: 将 PlayerExplorer 示例移动到新的项目中	270
7.6.2 步骤 2: 更改 TreeView 控件并添加 DataSource 和 DataGrid 控件	271
7.6.3 步骤 3: 将代码添加到示例中	271
7.7 总结	275
第八章 WFC 控件开发	277
8.1 子类化控件	278
8.1.1 步骤 1: 创建控件项目	279
8.1.2 步骤 2: 创建 ClassInfo 类和 Value 属性	279
8.1.3 步骤 3: 定义 NonNumberEntered 事件	281

8.1.4 步骤 4: 将代码添加到 GetValue 方法中	282
8.1.5 步骤 5: 使用 Class Outline 忽略方法	283
8.1.6 步骤 6: 将控件添加到窗体中	286
8.2 自定义控件	287
8.2.1 步骤 1: 创建控件项目	288
8.2.2 步骤 2: 将代码添加到构造函数中	288
8.2.3 步骤 3: 创建属性和事件	289
8.2.4 步骤 4: 忽略 Control 类的方法	291
8.2.5 步骤 5: 将 ClockTestForm 窗体添加到项目中	294
8.3 组合控件	295
8.3.1 步骤 1: 使用控件模板创建控件项目	296
8.3.2 步骤 2: 将控件添加到 UserControl 中	296
8.3.3 步骤 3: 创建控件属性	299
8.3.4 步骤 4: 创建 DefaultState 属性对话框	303
8.3.5 步骤 5: 将 DefaultStateEditor 值编辑器添加到项目中	305
8.3.6 步骤 6: 将公共方法添加到 AddressProvider 类中	306
8.3.7 步骤 7: 添加测试控件的窗体	308
8.4 WFC 到 ActiveX 的转换	309
8.4.1 步骤 1: 打开控件项目	310
8.4.2 步骤 2: 将控件类注册为 COM 类	310
8.4.3 步骤 3: 将控件打包到 COM DLL 中	310
8.4.4 步骤 4: 注册 COM DLL	311
8.4.5 步骤 5: 创建包含控件的 Visual Basic 项目	311
8.5 总结	313
第九章 COM 组件	315
9.1 通过对象的代码重用	315
9.2 广泛使用的组件	315
9.3 分布式 COM	316
9.4 不可见的 COM 组件	317
9.4.1 步骤 1: 使用 COM 模板创建项目	317
9.4.2 步骤 2: 将方法添加到类中	318
9.5 WFC COM 客户应用程序	321
9.5.1 步骤 1: 创建 WFC 应用程序项目	321
9.5.2 步骤 2: 设计窗体的用户界面	321
9.5.3 步骤 3: 导入 COM 组件	327
9.5.4 步骤 4: 添加支持代码	328
9.6 常见的用户界面组件	330

9.6.1 步骤 1: 创建项目	331
9.6.2 步骤 2: 将对话框添加到项目中	331
9.6.3 步骤 3: 将方法添加到 COM 类中	331
9.6.4 步骤 4: 编译 COM 客户应用程序	333
9.6.5 步骤 5: 导入 FormReuse COM 组件	333
9.7 数据库 COM 组件	335
9.7.1 步骤 1: 创建 COM 项目	336
9.7.2 步骤 2: 设计 CustomerSearchDlg	336
9.7.3 步骤 3: 将支持代码添加到 CustomerSearchDlg 中	339
9.7.4 步骤 4: 将代码添加到 CustomerSearch COM 类中	347
9.7.5 步骤 5: 编译 CustomerSearchEngine 客户应用程序	348
9.8 第三方 COM 组件	352
9.8.1 步骤 1: 创建 WFC 应用程序项目	353
9.8.2 步骤 2: 导入 Microsoft Word COM 组件	353
9.8.3 步骤 3: 将支持代码添加到应用程序中	354
9.9 总结	357
第十章 Visual J++ 应用程序的高级功能	359
10.1 示例环境	359
10.2 前期规划	361
10.3 开发计划	363
10.4 开发过程	364
10.4.1 准备	364
10.4.2 Order.java 对话框用户界面设计	365
10.4.3 COM 组件设计	367
10.4.4 Order.java 窗体: 代码的重要部分	368
10.4.5 MDIMain 窗体: 主要代码	387
10.4.6 应用程序的登录对话框	396
10.4.7 Previous Orders 窗口	399
10.4.8 Order Details 对话框	401
10.4.9 Password Change 和 About 对话框	401
10.5 遍历应用程序	403
10.6 总结	405
10.7 展望	405

第一章

Visual J++ 6.0 的新特性

Microsoft Visual J++ 6.0 与它以前的版本相比有很大不同。它现在不仅支持一个全新的“集成开发环境”(IDE),而且还提供了许多新工具和新特性来帮助您创建 Windows 应用程序和 Java applet。现在 Microsoft 已经将 Visual J++ 的一些旧特性进行了升级,使之能够适应“快速应用程序开发”(RAD)的模式。本章主要介绍 Visual J++ 的新特性。主要目的是向您介绍新的 Visual J++ 环境,以及使用本书中的示例时需要理解的内容。

1.1 集成开发环境

在 Visual J++ 6.0 中,开发环境已经从 Developer Studio(最初来源于 Visual C++)更改为一个以 Visual Basic 5.0 IDE 为基础的新的开发环境。您可以“多文档界面”(MDI)模式(默认模式),如图 1-1 所示,或者以“单文档界面”(SDI)模式,如图 1-2 所示,来显示新的环境。这两种环境的主要区别是:MDI 模式允许您将工具窗口停放在环境中,而 SDI 模式是将所有的窗口和工具栏都作为浮动的窗口来显示。使用 MDI 模式的主要优点是通过对窗口停放在环境中,您可以获得比 SDI 模式更大的屏幕空间。

提示 您可以通过在 Options 对话框中更改 SDI Environment 选项来更改 IDE 模式。通过在 Option 对话框中的树结构中定位到 Environment 节点,然后选定 General,您可以找到该选项。

您也可以使用标签链接 Visual J++ 的窗口来节省空间,如图 1-3 所示。当用标签链接两个或多个窗口时,它们将叠放在一起,只有最上层的窗口可以看见。标签链接的窗口占据相同的窗口框架,通过单击框架底部的标签来访问每一个窗口。如果您想在特定的时间显示指定的窗口和工具栏,而不显示其他窗口和工具栏,那么 Visual J++ 允许您定义窗口布局。Visual J++ 中有一些预先定义好的窗口布局,用来在执行任务(如进行

调试或使用 Forms Designer(窗体设计器))时显示适当的窗口。您可以通过从 View 菜单中选择 Define Window Layouts 命令来添加或删除窗口布局。

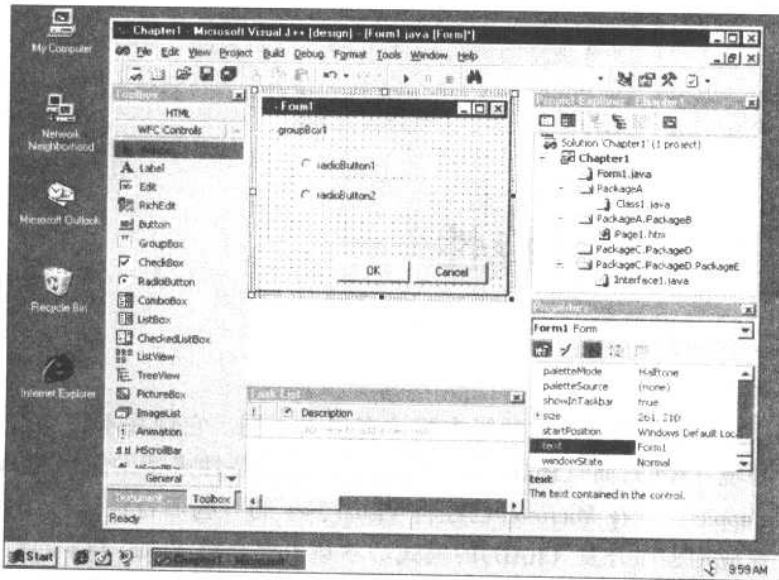


图 1-1 以 MDI 模式显示的 Visual J++ (默认情况)

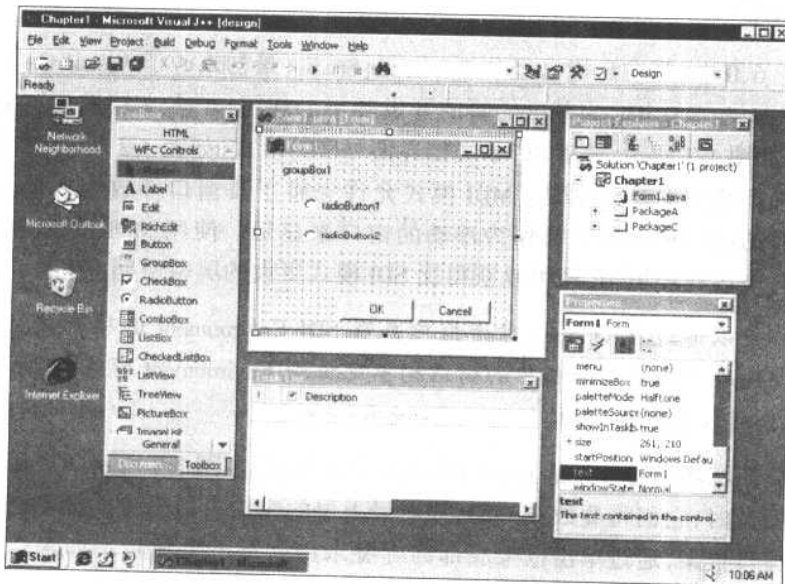


图 1-2 以 SDI 模式显示的 Visual J++

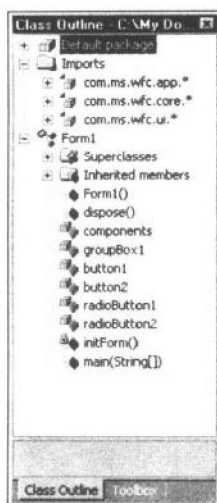


图 1-3 标签链接的窗口

提示 要将一个窗口标签链接到另一个窗口,您可以将要链接的窗口拖到要链接到的窗口的左上角。当显示该窗口的标签或轮廓时释放鼠标键。

Project Explorer(项目浏览器)是一个新的 Visual J++ 工具,它可以帮助您定位和控制以 Visual J++ 目录为基础的项目。Project Explorer 可以在两种视图——数据包视图(Package View)和目录视图(Directory View)——中显示项目目录和条目。图 1-4 显示了这两种视图的样子。

您可以使用 Project Explorer 的环境菜单来添加、删除项目中的文件。删除的文件仍保留在项目的目录结构中,但不再被认为是项目的一部分。虽然该特性打破了基于目录的项目模式,但它使您可以更容易地维护项目代码,确定正确的代码进行编译。

Visual J++ 允许您同时打开和使用多个项目。Project Explorer 会将项目分组成一些解决方案,使您可以容易地打开和使用相关的项目。Visual J++ 会自动为您创建解决方案文件。

Visual J++ 其中一个最重要的增加内容就是 Forms Designer 数据包。该数据包包括 Forms Designer(窗体设计器)、Toolbox(工具箱)和 Property Browser(属性浏览器),如图 1-5 所示。Forms Designer 允许您设计窗口及其控件的布局。Format 菜单提供了一些工具,帮助您调整、对齐和锁定窗体中的控件。集成的菜单编辑器可以快速地將菜单添加到窗体中,以及将控件添加到环境菜单中。(要了解创建和管理菜单的详细信息,请参阅第四章“窗口附件”的有关内容。)

Toolbox 和 Forms Designer 一起使用,其中显示了可以添加到窗体中的组件。Toolbox 可以帮助您将组件组织到 Toolbox 的组中,以管理相关的组件。