第一部分

使用 Visual C++

第1章 用Appwizard创建一个新的应用程序

在用Visual C++开发环境编制应用程序时，必须首先运行Appwizard。它为应用程序生成相应的开头文件，并使文件与Class Wizard兼容。AppWizard显示一系列对话框，用户按要求输入各项参数，Visual C++据此生成相应的开头文件。

Appwizard允许你指定许多只适用于本程序和项目的选项。例如，你可指定本应用程序是单文档、多重文档或基于对话的界面，也可选择DLE支持和ODBC支持，另外也可以立即获取系统所提供的内部功能如File菜单上的Open(打开文件)，Save As(存文件)和Print(打印)命令，甚至，你还可以决定到底是创建一个外部的make文件还是内部的make文件。

程序和项目选项定义完后，相应的开头文件就包括了创建Windows应用程序所需要的一切要素：源文件、头文件、资源文件、项目文件，等等。Visual C++的源文件撑起了程序中类的骨架(原始版本)，类的具体实现是根据你在Appwizard对话框中所确定的选项来完成的。在用内部功能创建项目的同时，应用程序的框架也生成了。

1.1 创建一个Appwizard项目

Appwizard显示一系列对话框，当中列出了表示应用程序特性的一些选项。可以将对话框序列视为一条创建路径，它因应用程序而异，所以有些对话框并不是在创建每个应用程序时都会显示。你可以在这些对话框中循环移动，向前或向后都可以，只要还未创建应用程序，你就可以随时改变任一选项。

创建一个新的项目须遵循以下步骤：

1. 启动Visual C++

2. 从File菜单上，选取New命令(或按热键CTRL+N)，这时New对话框就会出现。

3. 在New对话框中，选择Project。

New对话框允许用户选择好几种资源类型——code/TEXT(代码/文本)，Project(项目)，Resource Script(资源正本)，Binary File(二进制文件)，Bitmap File(位图文件)，Icon File(图标文件)，或者Cursor File(光标文件)。

4. 选择OK。此时会显示New Project对话框。

5. 在Project Type(项目类型)表中，选择MFC AppWizard (exe)。

项目类型表允许用户选择好几种项目类型：MFC(Microsoft基本类库)，AppWizard (exe)，MFC Appwizard (dll)，Application(应用程序)，Dynamic-Link Library(动态连接库)，Console Application(控制台应用程序)以及Static Library(静态库)。

6. 在Project Name(项目名)框中输入项目名。

Visual C++自动将你输入的名字作为目录名写入New Subdirectory(新子目录)框中。另外，它还自动在Directory(目录)框的上面显示项目文件的路径。Appwizard会记

住你在 Project Name 框中输入的名字，并据此派生出该应用程序的文件和类的缺省名。

提示 如果一个非“Appwizard 项目”类型被选中（例如应用程序或静态库），New Subdirectory 域将会呈现空白，因为 Visual C++认为项目所在的目录即源文件所在目录，而该目录是现存的，故无须用户输入其名称。当然，也可以直接在 New Subdirectory 框中输入目录名或在 Project Name 框中输入完整的路径名。另外，如果你选择了非 Appwizard 的项目类型，Project Files（项目文件）框将被显示，这样你就可以将现存的文件加到新项目中去了。

7. 为项目指定目标平台。

利用所提供的列表框为项目选择任一个可用的平台。

注意 缺省平台是 Win32。如果要选择其它的目标平台，则必须安装 Visual C++相关的交叉开发版本。

8. 在 Directory 框中为项目指定路径。

利用列表框搜索所选驱动器上的各层目录。

9. 改变 New Subdirectory 中的目录名。这一步是可选的。

New Subdirectory 中包含了项目的文件。AppWizard 自动将项目名作为 NewSubdirectory 中的名称。

10. 选择 Create。

Appwizard 体系结构任选项如图 1.1 所示。

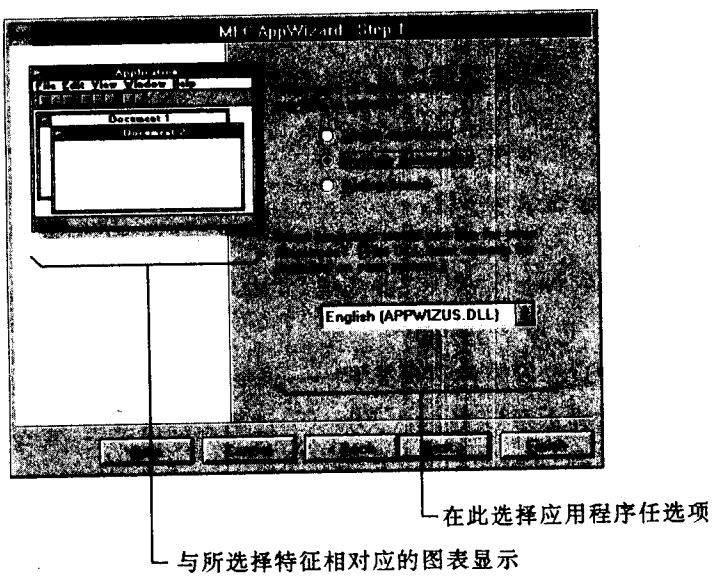

图 1.1 AppWizard 的体系结构任选项

选择一种体系结构类型和资源语言必须遵循以下步骤：

1. 从以下三种类型中择一：

Single document（单文档）。SDI（单文档界面）体系结构允许用户一次只能用一个文

档工作。Windows 中的 Notepad 就是一个 SDI 应用程序的例子。

Multiple documents（多文档）。MDI（多重文档界面）允许用户打开多个文档，每一个文档都对应一个窗口。Windows 中的 File Manager（文件管理器）就是一个 MDI 应用程序的例子。

Dialog-Based（基于对话）。一个基于对话的体系结构会显示一个简单的对话框以便用户输入。MFC Trace Options 就是一个该结构的应用程序实例。

注意 当选择 Appwizard 的特征时，Appwizard 对话框左边会显示与所选特征相对应的图表。例如，如果你选择了多重文档体系结构，将会在名为“Application”（应用程序）的窗口中显示两个文档。

2. 为资源文本选择一种语言：

■英语
■法语
■德语
■日语

注意 英语是资源文本的缺省语言。如果想选择其它语言，必须确认系统中已安装了 *LANGUAGE.DLL* 文件。

3. 选择 Next 以显示下一个 AppWizard 对话框。

如果应用程序的体系结构是单文档或多文档，AppWizard 数据库支持任选项将会被显示。

1.2 选择数据库和 OLE 任选项

接着显示的两个对话框让你决定你的项目是否以及在何种程度上支持 ODBC（开放的数据库连接性）和 OLE。

选择数据库支持任选项必须遵循以下步骤：

1. 从以下数据库支持任选项中择一：

None 该选项拒绝任何一个支持 ODBC 的库。如果应用程序没有使用数据库，选择该选项将会生成一个更小的程序。

Only include header files 该选项提供最低限度的数据库支持，即它只包括所有的数据库头文件和连接库。选择了该选项后，AppWizard 不会为你创建任何与数据库相关联的类，你必须自己动手创建。

A database view, without file support 该选项包括所有的数据库头文件和连接库。另外，Appwizard 还会创建一个记录视图和记录集供你查看记录。选择该选项会拥有文档支持但却不支持顺序（串行）支持。

Both a database view and file support 与上一个选项唯一不同处，是它不仅有文档支持还拥有顺序（串行）支持。

关于 MFC 数据库支持的更深入的信息，将会在《用 MFC 和 Win32 编程》一书的第 6 章中进一步阐述。

2. 如果你的应用程序包括数据库视图，必须定义数据源并选择一张表。

■选择 Data Source（数据源）以显示 SQL Data Source 对话框。注意必须选择那些已经通过 ODBC 管理器在机器上注册的数据源。

■在 Select a Table 对话框中，找到你想要的表格名称，该表格的内容正是你要加到记录集中去的，双击表名即可选中该表格。

■关闭 Select a Table 和 Data Source 对话框。

3. Choose Next 显示下一个 AppWizard 对话框。

如果应用程序的体系结构是单文档或多重文档，AppWizard OLE 任选项将接着显示。

AppWizard 为各种不同的 OLE 应用程序类型产生相应的支持代码。选择任一个 OLE 任选项都会使标准 OLE 资源有效并把额外的 OLE 命令附加到应用程序菜单条上。

选择 OLE 任选项必须遵循以下步骤：

1. 选择 OLE 复合文档支持类型：

None 这是缺省值，在此情形下，AppWizard 创建的应用程序是不带 OLE 支持的。

Container 该选项表明应用程序将包含连接和嵌入式对象（OLE）。

Mini Sever 该选项表明应用程序只允许创建嵌入式对象。

Full Server 该选项表明应用程序既可以独立运行，又能支持 OLE。除此之外，它创建的对象还能被复合文档所容纳。

Both container and Server 该选项表明应用程序既能包容 OLE，又能作为服务器使用。

有关 MFC OLE 支持的更深入的信息，将在《用 MFC 和 Win32 编程》一书的第 5 章进一步阐述。

2. 选择是否支持 OLE 自动支持功能：

Yes 该选项允许创建支持 OLE 方法和特性的对象，并且将文档类当作一个可编程对象，这样用户可随意选择使用任一种自动功能。作为缺省情形，AppWizard 创建的应用程序不带 OLE 支持。

No 该选项表明 AppWizard 创建的应用程序不带有 OLE 自动支持功能。

3. 选择 Next 命令以显示下一个 AppWizard 对话框。

AppWizard 应用程序任选项将会被显示。

1.3 选择应用程序和项目任选项

依据应用程序的体系结构，可在各种不同的选项中择一。

选择应用程序任选项应遵循以下步骤：

1. 首先要指定应用程序的基本特征，从下面的选项中择一即可：

Dockable Toolbar AppWizard 为每一个工具条产生相应代码。工具条包含了许多按钮，它们的功能分别是：创建新文档、打开和存储文档文件、剪除、拷贝、剪贴、打印文本、显示 About Box 和支持用热键 Shift.F1 寻求帮助。另外，该选项还能增加相应的菜单命令以显示和隐藏工具条。

Initial Status Bar AppWizard 为每一个状态条产生相应代码。状态条包含了与键盘上某些键相对应的自动指示器，这些键是 CapsLock NumLock 以及 ScrollLock。同时，还有

一消息行显示有关菜单命令和工具条按钮的帮助信息。另外，该选项还能增加相应的菜单命令以显示和隐藏状态条。

Printing and Print Preview AppWizard 通过调用 MFC 中的CView 类的成员函数产生相应的代码去处理打印，打印设置以及打印预显等功能。另外，它还在 File 菜单上增加相应的命令去处理这些功能。

About Box AppWizard 创建一个对话框以显示软件版本以及版权启事。该对话框还扼要地显示有关软件生产厂家和作者的信息。如果应用程序是基于对话的体系结构，Visual C++将在主对话框的控制菜单上增加 About 命令。如果应用程序是 SDI 或 MDI 的体系结构，那么 About 框会自动生成。

Context Sensitive Help AppWizard 产生一系列文件以提供与上下文有关的帮助。Help 编译器随 Visual C++系统提供。

Use 3D Controls AppWizard 用三维阴影建立可视界面。

MRU List Files 系统中有一张表用来记录最近使用过的文件，该选项设置表中文件的数目。

Dialog Title 对于基于对话的应用程序，要输入对话标题条的名称。

2. 如果应用程序的体系结构是 SDI 或 MDI，并且你感到有必要调整刚才已选定的某些选项，可选择 Advanced 命令，此时会显示 Advanced Options 对话框：

- 选择 Document Template Strings（文档样板字符串）并根据需要改变字符串。
- 如果应用程序是 SDI 结构，选择 Main Frame 并改变标题和主框架风格。如果有必要，可接着选择 Use Splitter Window，这一步是可选的。
- 如果应用程序是 MDI 结构，选择 Main Frame 并改变标题和主框架风格。接着选择 MDI Child Frame 并改变 MDI 子框架的风格。作为可选的步骤，可接着选择 Use Splitter Window。
- 选择 Close。

3. 选择 Next 以显示下一个 AppWizard 对话框。此时 AppWizard 项目任选项将会被显示。

选择项目任选项应遵循以下步骤：

1. 首先从以下所列设置中择一：

Generate Source File Comments AppWizard 在源文件中适当的地方插入相应注释以帮助完成编程工作。具体包括一系列指示器以及文件 README. TXT。指示器告诉应在何处加上你自己的代码，而 README. TXE 则详细描述每一个产生的文件。这一选项我们推荐用户使用。

Type of Makefile 作为缺省情形，AppWizard 产生一个 Visual C++的 make 文件，它与 Visual C++以及 NMAKE 兼容。如果想生成一个 NMAKE 文件，它能够直接被编辑，但只能被当作外部项目来运行，那么请选择 External Makefile。

MFC Library Linkage Microsoft 的基类既能动态链接又能静态链接。

2. 选择 Next 命令以显示下一个 AppWizard 对话框。此时，AppWizard 类汇总任选项将会被显示。

1.4 结束 AppWizard 过程

AppWizard 首先显示它即将创建的类，头文件以及组成文件的名称。类名允许作不同程度的修改。不过请记住，颜色呈灰暗显示的文本框中的信息是不能改变的。

按以下方法修改类名：

在 Classes 对话框上部的列表框中选取类名。此时，其余的四个框便会显示相关类名以及组成文件。可以根据应用程序的需要改变类信息的名称。若程序是 SDI 或 MDI 结构，还允许改变视图中的基类。

按以下步骤创建应用程序：

1. 选择 Finish 命令以显示 AppWizard' s New Project Information 汇总。

2. 选择 OK 命令表示你已确认以前所选的各选项都是正确的。于是，AppWizard 便会根据这些选项来生成应用程序的源文件。如果想改变某些选项，请选择 Cancel 命令以关闭 New Project Information 对话框。这一动作将使你有机会重新访问刚才已接触过的那些对话框。

1.5 AppWizard 所创建的文件

AppWizard 总是会创建一张基本文件表，这与你所作的选项毫无关系。不过不要认为你刚才的许多工作是无意义的，AppWizard 会用你在 Project Name 框中所指定的名称为大多数文件和类派生出相应名称。有些文件名截取项目名的前五个字母。

毫无疑问，你想检查一下新创建的源代码文件，为此 AppWizard 在新项目目录中创建了一个文本文件 README. TXT，它详细地描述了每一个文件的内容和用途。

如果你想更多地了解关于 AppWizard 所创建的文件，请参看《用 MFC 和 Win32 编程》中“AppWizard：Files Created”一节。

第2章 项目的使用

项目提供了用户所创建的程序和库的有关信息。具体来说，可以概括为以下三个方面：

■用来编制程序的源文件名称和位置。

■编制程序时所用到的工具设置，例如编译器和链接器选项。

■Visual C++工作区的外观和布局。

有两种途径可以创建新项目：

■选择项目类型为 AppWizard (MFC)，这样，AppWizard 会自动用适当的类去建立相应文件并把它们加到项目中去。具体请参看第 1 章“用 AppWizard 创建一个新的应用程序”。

■选择其它的项目类型，这样，你就必须自己创建所有文件并有选择地将它们加到项目中去。

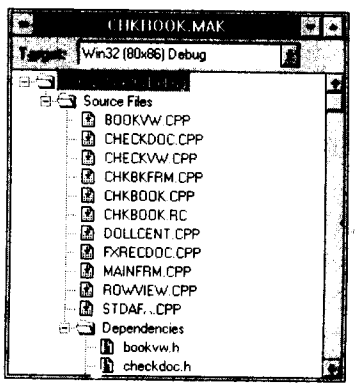

图 2.1 项目窗口

项目窗口显示一张图以描述用来创建项目的各源文件间的关系，如图 2.1 所示。不过，这里表现的是逻辑关系而非物理关系，故并不反映文件在硬盘上的布局，窗口标题为×××.MAK，可通过该窗口访问项目内各组成成份的信息。

提示 当你处于项目窗口中时，在许多情形下只要单击右鼠标按钮就会在屏幕上显示一份热键菜单，上面列出了一些频繁使用的命令。当然这些命令与光标当前所处的位置有关。例如，当光标落在某个源文件上时，热键菜单就会显示 Properties 命令，还有另外一些在 Project 菜单上可用的命令，比如 Build 和 Compile。

2.1 项目类型

创建项目时都会为它定义一个类型。项目类型规定了所生成目标的种类并决定应用程序中各选项的缺省值。

Visual C++2.0 允许选择以下六种项目类型中的一种：

MFC AppWizard (exe)

应用程序是用 MFC 开发的完全的图形界面。Visual C++会自动选用适当的类去创建骨架文件并把这些文件加到项目中。文件扩展名是. EXE。

MFC AppWizard (dll)

这是用 MFC 来开发的函数库。Visual C++会自动选用适当的类去创建骨架文件并把这些文件加到项目中。文件扩展名是 .DLL。

Application

应用程序是用 Windows NT Win32 API 函数或用 MFC 创建的完全的图形界面。文件

扩展名是 .EXE。

Dynamic_ Link Library

该函数库是用 Windows NT Win32 API 函数开发的，可以被 32 位的 Windows 应用程序在运行时动态调用。文件扩展名是 .DLL。

Console Application

应用程序是用 Console APIs（控制台应用程序编程接口）开发的，并以字符方式支持控制台流的操作。Visual C++也可使用标准输入/输出函数所提供的控制台 I/B 流操作，例如函数 **printf** () 和 **scanf** ()。文件扩展名是 .EXE.

Static Library

标准库是根据项目所含的对象文件和其它库文件直接创建的。库中包括所有项目所包含的对象文件，所生成的对象文件以及项目所使用的库。文件扩展名是 .LIB。

2.2 使用项目

用户可根据需要在项目中增加和移动文件，也可增加或删除目标，并更新文件间的依赖关系。

创建一个项目

创建一个项目就意味着创建下列两个文件：

- Make 文件。该文件的扩展名是 .MAK。它包含建立项目目标所必须具备的一切条件：所有的命令、宏定义、任选项等 。
- 项目设置文件。该文件的扩展名是 .VCP。它包含 Visual C++的环境设置，例如窗口的大小和位置，插入点的位置，项目断点的状态以及 Watch 窗口的内容等等。

用户不能直接修改这两个文件。

创建一个项目须遵循以下步骤：

1. 从 File 菜单上选取 New 命令。此时会显示 New 对话框。
2. 从列表框中选择项目。
3. 选择 OK。此时会显示 New Project 对话框。
4. 在 Project Name 文本框中输入项目名称。此时 New Subdirectory 文本框中会自动显示同样的名称。当然也可以根据需要改变此子目录名。Visual C++会自动为项目文件创建新的子目录。
5. 在下拉式列表框 Project Type 中选择应用程序类型。
6. 如果有必要的话，选择子目录所在的驱动器和父目录。
7. 选择 Create。

如果应用程序类型不是 MFC AppWizard，那么会接着显示 Project Files 对话框。否则，请参看第 1 章中“创建一个 AppWizard 项目”一节。

8. 将文件加到项目中去。具体请看本节中的“添加和移动文件”内容。
9. 选择 Close。

当关闭 Project Files 对话框时，Visual C++会浏览项目中的所有文件以更新文件间

依赖关系，并在项目窗口中显示文件以及它们之间的依赖性，正如图 2.1 所示。

Visual C++自动从新加入的项目文件中递归搜索“**#inculde**”字样。尖括号括起的〈*incl.h*〉和引号引起的“*incl.h*”都能被识别。所有的源文件（.c，.cpp 和 .cxx）以及资源文件（.RC）都将被浏览。Visual C++会将它找到的所有包含文件都加入到一个依赖（Dependencies）子组中去。该子组中的文件是不能由用户来增加或删除的。它们带有扩展名 .H，.HXX，.INC，.FON，.CUR，.BMP，.ICO 以及 .DLG。

Visual C++也引用下列两个排它文件：

SYSINCL.DAT　该文件包含一张系统包含文件的缺省表。它是由启动程序 MSVC.EXE 安装到系统中的，位于 Visual C++所在的目录中。

MSVCINCL. DAT 该文本文件可由用户创建并放在 Windows 目录下。它列出了用户想排除的一些补充文件。在追加项时必须借助该文件，因为刚才所提及的 SYSINCL. DAT 在下列几种情形下有可能被重写：当用户重新安装 Visual C++时或用 Setup 改变设置、更新设置时；如果你用 Visual C++文本编辑器创建该文件，记住先退出 Visual C++环境再重新启动它以确保文件有效。

这些表中所包含的文件不应经常修改。Visual C++无论在何时更新文件间依赖性，这些文件都被排除在外，依赖子组中也不会显示它们。如果你想改变任一张表中的文件，请选择 Project 菜单上的 Rebuild All 命令以确保能成功创建你所选定的目标。如果仅仅选择 Build 命令，依赖子组不会有所变化，Visual C++将会报告目标已过时。

使用热键菜单

热键菜单上面列出了一些命令以帮助用户在项目窗中快速完成当前选择。当然也可以借助于主菜单条来完成，只是速度慢一些。

将鼠标箭头移到项目窗口中并单击鼠标右按钮即可在屏幕上显示热键菜单。接下去就可以选择你想要的命令。

添加和移动文件

可以使用菜单命令来添加和移动文件。也可以用鼠标将要添加的文件直接拖到项目窗口中。

添加文件须遵循以下步骤：

1. 从 Project 菜单上选取 Files 命令，此时会显示 Project Files 对话框。

2. 如果项目中不止一个元组，从 Add Files To This Group 下拉式列表框中选择相应元组。

3. 从 List Files of Type 下拉式列表框中选择文件类型。该类型的文件名显示在 File Name 列表框中。

4. 如果有必要的话，选择要查看的驱动器和目录名。

5. 用下列方法添加文件：

■单个文件：从 File Name 列表框中选择文件名并选中 Add 命令。

■单个文件：双击 File Name 框中的文件名。

■File Name 列表框内的所有文件：选择 Add All 按钮。

重复以上步骤直至所有要添加的文件都被加到项目中去。当 Project Files 对话框被关

闭时，Visual C++会自动搜索所有文件以找出它们之间的依赖关系，并把包含文件添加到对应的依赖子组中。注意依赖子组中的文件是不能直接修改的。

当然，也可以用鼠标将要添加的文件图标直接拖到项目窗口中。例如，可以从 File Manager（文件管理器）中将文件拖到项目中。

注意 以上所提供的方法仅仅改变项目的文件表，并不改变文件在硬盘上的物理位置。特别要指出的是，将文件从 File Manager 中拖到项目窗口内仅仅是在项目的文件表中添加上该文件，并不是把文件加到项目所在的目录中。

从项目中移动文件须遵循以下步骤：

1. 从 Project 菜单上选取 Files 命令，此时会出现 Project Files 对话框。

2. 如果项目中不止一个元组，从 Add Files To This Group 下拉式列表框中选择所要的元组。

3. 如果有必要的话，选择要查看的驱动器和目录名。

4. 从 Files In Group 列表框中选择要移动的文件，然后选择 Remove。

重复以上步骤直至所有要移动的文件都已移动完毕。当 Project Files 对话框被关闭时，Visual C++会自动搜索所有文件以找出它们之间的依赖关系，并从依赖子组中移走那些无用的包含文件。当然，依赖子组中的文件用户是不能直接修改的。

如果要想从项目中快速移走文件，只要先选中文件，再从 Edit 菜单上选择 Delefe (Del) 命令。

另外，也可以在项目窗口中按下 CTRL 或 SHIFT 键，选择文件。

保存项目

用 Save 或 Save As 命令保存项目时首先要激活项目窗口，因为 File 菜单上的命令只对当前活动窗口有效。

保存项目的步骤如下：

1. 激活项目窗口。

2. 从 File 菜单上选取 Save 命令或按热键 CTRL+S。

工具条：

Visual C++会自动保存项目，而无须用户再做任何工作。

以一个新名称保存一个已有项目的步骤如下：

1. 激活项目窗口。

2. 从 File 菜单上选取 Save As 命令，或按热键 F12。此时会显示 Save As 对话框。

3. 在此对话框中，输入新的文件名，并选择所要的驱动器、目录和文件类型。

4. 选择 OK。

如果想把所有的文件都存到一个项目中，请从 File 菜单上选择 Save All 命令。这样，Visual C++会将所有修改过的文件都存到一个项目中，无论它们原来是否包含在同一个项目中。

打开一个已有项目

打开项目时 Visual C++会自动按最近一次的使用状态恢复环境设置，并打开浏览信

息文件。

打开一个已有项目的步骤如下：

1. 从 File 菜单上选取 Open（CTRL+O）命令。

此时会显示 Open 对话框。List Files of Type 下拉式列表框内的缺省值是 Main Files，它包含×××. MAK 文件。如果你只想查看×××. MAK 文件，请在列表框中选择 Projects。

2. 选择项目所在的驱动器和目录。

3. 从 File Name 列表框中选择. MAK 文件再选择 OK，或者双击文件名。

此时会显示一个标题为×××. MAK 的项目窗口，正如图2.1所示，它以图表形式显示项目中的文件。

如果所打开的项目是在2.0以前的版本中创建的，系统会询问你是否将原来的 MAKe 文件转换成与当前版本兼容的 MAKe 文件。如果需要转换，Visual C++会显示 Save As 对话框，这样你就可以用一个新名称来保存原项目。对话框中显示的缺省名是以前版本中的项目名，如果使用该名称，那么以前的项目文件会被新的项目所覆盖。如果你不想重写旧文件，必须重新设置中间文件和结果文件所在的目录。具体细节请看本章2.4节中的“为结果文件指定所在目录”。

关闭项目

要想关闭一个打开的项目，有以下三种方法：

■激活该项目窗口后从 File 菜单上选择 Close 命令。

■ 打开另一个项目。

■ 创建一个新项目。

更新项目内文件间的依赖关系

在编辑完一些带有“**#include**”字样的源文件之后，必须更新项目内文件间的依赖关系，即把包含文件加到相应的依赖子组中去。

要想更新项目中所有文件的依赖性，须从 Project 菜单上选取 Update All Dependencies 命令。

若想更新项目中单个文件的依赖性，步骤如下：

1. 在项目窗口中选中欲更新的文件或直接打开该文件。

2. 从 Project 菜单上选取 Update Dependencies 命令。

Visual C++自动浏览项目文件以递归搜索“**#include**”字样。尖括号括起的〈*incl.h*〉和引号引起的“*incl.h* 都能被识别。另外，它也引用下列两个排它文件：

SYSINCL. DAT　该文件包含一张系统包含文件的缺省表。它是由启动程序 MSVC. EXE 安装到系统中的，位于 Visual C++系统所在的目录中。

MSVCINCL. DAT　该文本文件可由用户创建并放在 Windows 目录下。它列出了用户想排除的一些补充文件。在追加项时必须借助该文件，因为刚才所提及的 SYSINCL. DAT 在下列几种情形下有可能被重写：当用户重新安装 Visual C++时或用 Setup 改变设置、更新设置时，如果你用 Visual C++文本编辑器创建该文件，记住先退出 Visual C++环境，

再重新启动它以确保文件有效。

记住这些表中所包含的文件是不应被经常修改的。Visual C++无论在何时更新文件间依赖性，这些文件都不包括在内，依赖子组中也不会显示它们。如果想改变任一张表中的文件，请选择 Project 菜单上的 Rebuild All 命令以确保能成功创建你所选定的目标。如果仅仅选择 Build 命令，依赖子组不会有所变化，而 Visual C++将会报告目标已过时。

使用目标

一个内部目标实际上是从项目所得到的一个二进制结果文件。Visual C++依据输入源文件以及项目说明来生成目标。具体地说，它依据下列规范组合来创建目标：

■项目所在的平台

■二进制结果文件的类型（应用程序，静态库、动态连接库等）

■编译器，链接器的设置

■指定的源文件

你可在任何时候从项目窗口内的下拉式列表框中选择一个特定目标去创建。

也可用不同的选项设置去定义新的内部目标，这样就可形成一个项目的多种版本。当你在 Project Target 对话框中定义一个新目标时，可以为该目标重新指定选项设置，只要利用 Project Settings 对话框即可。例如，你可以重作优化选择。

定义一个新目标的步骤如下：

1. 从 Project 菜单上选取 Targets 命令。此时会显示 Targets 对话框。

2. 选择 New。此时会显示 New Target 对话框。

3. 在 Target Name 文本框中输入目标的描述名。

4. 如果新目标使用已定义的缺省设置，请选择 Use Default Settings 按钮。若有必要，可从下拉式列表框中选择目标类型。如果希望目标包含调试信息，请选择 Debug Build 检查框，否则清除该检查框。

另外，如果想在某个已有的目标基础上建立新目标，请选择 Copy Sottings 按钮，然后从下拉式列表框中选择基目标。

5. 选择 OK。

6. 在 Targets 对话框中，选择 OK。

新创建的目标名被加到 Target 下拉式列表框中。若想编制该目标，只要从列表框中选中它，然后再从 Project 菜单上选择 Rebuild All 命令。

删除一个目标定义的步骤如下：

1. 从 Project 菜单上选取 Targets 命令。此时会显示 Targets 对话框。

2. 从 Target 列表框中选中欲删除的目标名。

3. 选择 Delete。此时会显示一个确认消息框。

4. 选择 Yes 以确认删除。

5. 选择 OK。

如果目标名不能充分描述该目标的目的，你也可以改变它。

改变目标名的步骤如下：

1. 从 Project 菜单上选取 Targets 命令。此时会显示 Targets 对话框。

2. 从 Target 列表框中选中欲改名的目标。

3. 选择 Rename。此时会显示 Rename Target 对话框，框内标号为 Old Target Name 处显示原来的目标名。

4. 在 New Target Name 框中输入新名称。

5. 选择 OK。

2.3 在项目中使用元组

当创建一个新项目时，Visual C++会向项目中添加一个名为 Source Files（源文件）的元组。对于大多数项目来说，这一个元组已足够了。当然，也可以创建其它元组。元组可以用来从逻辑上显式识别项目层次结构中的文件组。例如，可以建立一个元组以包括所有的用户界面文件，也可包括其它与项目有关的文件：诸如项目说明，文档记录或测试组等。这些文件都用不同的图标表示，如表2.1所示。

当然，只能在项目一级上添加元组，而不能在元组中嵌套定义元组。

若想添加一个元组只须从 Project 菜单上选择 New Group 命令即可。另外，你可在任何时候重新命名已定义过的元组。

重新命名元组的步骤如下：

1. 从项目窗口中选中该元组。

2. 从 Edit 菜单上，选取 Properties 命令，或按热键 ALT+ENTER。也可用单击右鼠标按钮以显示热键菜单，再从菜单上选择 Properties 命令。

3. 在 Group Name 文本框中输入新的元组名。

注意 *依赖子组是 Visual C++系统自动创建的，它不能由用户来改变名称。*

移动一个元组的步骤如下：

1. 从项目窗口中选中该元组。

2. 从 Edit 菜单上选取 Delete（Del）命令。

移动一个元组意味着从项目文件列表框中移动该元组内的文件。你可以用鼠标将文件图标从一个元组拖到另一个元组中。除此之外，也可借助 File Manager（文件管理器）夹添加文件。

注意 *以上所提供的方法仅仅改变项目的文件表，并不改变文件在系统内的物理位置。特别要指出的是，将文件从 File Manager 中拖到项目窗口内仅仅是在项目的文件表中添加上该文件，并不是把文件加到项目所在目录中。*

把文件从一个元组移到另一个元组的步骤如下：

1. 从项目窗口内选中源项目元组。（按下 CTRL 或 SHIFT 键可以一次性选中多个文件。）

2. 用鼠标将文件从源元组中拖到目标元组中，也可拖到目标元组结点或目标元组中的文件结点上。或者从 Edit 菜单上选择 Cut 命令，按热键 CTRL+X 也可，然后在项目窗口中选中目标元组，再从 Edit 菜单上选择 Paste 命令（也可按 CTRL+V 键）。

拷贝文件到一个元组的步骤如下：

1. 从项目窗口内选中源项目元组。（按下 CTRL 或 SHIFT 键可以一次性选中多个文件。）

2. 按住 CTRL 键再用鼠标将文件从源元组中拖到目标元组中，也可拖到目标元组结点或目标元组中的文件结点上。或者从 Edit 菜单上选择 Copy 命令，按热键 CTRL＋C 也可，然后在项目窗口中选中目标元组，再从 Edit 菜单上选择 Paste 命令（也可按 CTRL＋V 键）。

从一个项目移动/拷贝文件或元组到另一个项目的步骤如下：

1. 从项目窗口中选中欲移动或拷贝的文件和元组。（按下 CTRL 或 SHIFT 键允许一次选中多个文件。）

2. 从 Edit 菜单上，选取 Cut 命令（或按 CTRL＋X 键）以移动文件或元组，或选取Copy 命令（或按 CTRL＋C 键）以拷贝文件或元组。

3. 关闭当前项目。

4. 打开目的项目。

5. 如果是移动或拷贝文件，请选中目的元组。否则选择目的项目以接受元组。

6. 从 Edit 菜单上，选取 Paste 命令，或按热键 CTRL＋V。

提示 如果按住 CTRL 键时单击一个选择项，该项的状态将会发生翻转。你可以用这种方法从一个多重选择中快速装卸一个文件。

如果选择项中包含有依赖子组中的文件，Visual C＋＋仅仅显式地移动或拷贝源文件。但在项目建立之前，它会更新文件间的依赖关系并把文件放到适当的依赖子组中。

2.4 在项目内设置任选项

对大多数项目而言，在项目一级上设置任选项已经足够了。你可以为不同的目标设置与其对应的任选项，这些目标既可以是系统自动创建的，也可以是自己创建的。一般来讲，每个平台至少需要两个目标：Debug 和 Release。

项目可以为每个目标显示其任选项的层次结构。在项目一级上设置的任选项适用于项目内所有的文件。当然，如果想用特定的选项编译某些文件，也可以特别指定只适用于这些文件的选项。例如，如果为项目设置的优化选项值是 Default，那么项目内所有文件都采用 Default 作为优化值。但你完全可以为某些文件重新指定优化值，或者索性指定某个文件无优化值。换言之，在文件一级上设置的选项会屏蔽掉在项目一级上指定的选项。

但有些类型的选项只能在项目一级上设置，例如连接属性。

总之，可以从下列两个角度来设置选项：

- 在项目一级上设置。该层次上指定的选项适用于项目内的所有动作和文件，除非对某些文件而言，被文件一级的选项所屏蔽。
- 在文件一级上设置。该层次上指定的选项只适用于特定的文件以及文件级上的动作，例如编译。

为结果文件指定所在目录

你可以为每个目标指定存放中间文件和结果文件的目录。通过把文件放在不同的目录中这种方法，可以在系统内同时保存同一文件的不同副本，这些副本是用不同的途径建立起来的，例如，目标文件可以有两种版本：调试和释放版本。

指定结果文件所在目录的步骤如下：

1. 从 Project 菜单上，选取 Settings 命令。此时会显示 Project Settings 对话框，如图2.2所示。

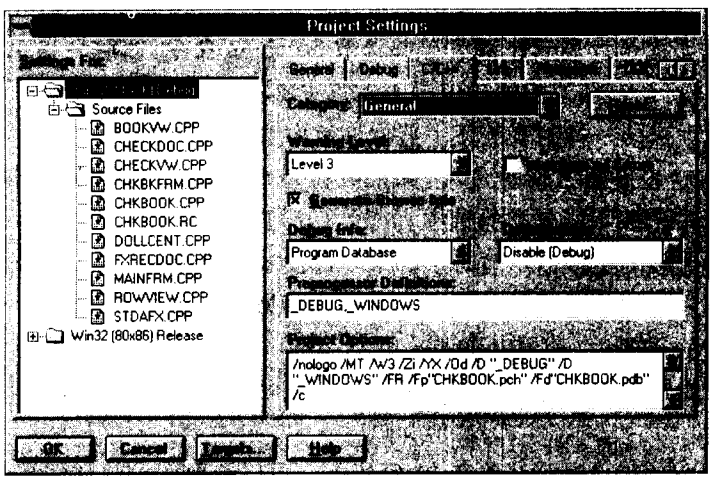

图 2.2 Project Settings 对话框

2. 从位于对话框左部的窗口中选取欲替其设置目录的结点。如果是项目结点，则既可以指定中间目录又可以指定目标目录。如果是文件结点，则只能指定中间目录。

3. 从对话框顶部的一行制表按钮中选取 General 按钮。该按钮包含了项目的部分选项。它说明项目如何使用 MFC 并指定中间和结果目录。

4. 在 Intermediate Files 文本框中为中间文件（例如.OBJ 文件）输入目录名。

5. 若是在项目一级上设置目录，则在 Output Files 文本框中输入目标文件（例如.EXE 文件）所在的目录名。

6. 选取 OK。

指定项目设置

记住，只能在项目被打开以后才能为其设置选项。

指定项目设置的步骤如下：

1. 从 Project 菜单上，选取 Settings 命令。此时会显示 Project Settings 对话框，如图2.2所示。

2. 从对话框左部的窗口中为目标选取项目一级的结点。当然也可以为目标选取多个项目级结点，并为多个项目设置公共选项。

3. 从对话框顶部的一行制表按钮中选取你欲设置的选项类型。

4. 在所选的制表按钮上设置选项。如果按钮是 C/C++和 Link，则可以根据需要从