赵国瑞　陆明　主编

计算机软件技术基础实验指导 习题及解答

——C++、数据结构、软件工程

天津大学出版社
TIANJIN UNIVERSITY PRESS

计算机软件技术基础实验指导 习题及解答

——C++、数据结构、软件工程

赵国瑞 陆明 主编

天津大学出版社

内 容 提 要

本书与天津大学出版社出版的《计算机软件技术基础——C++、数据结构、软件工程》(第2版)配套。全书内容分为两部分:第1部分"实验指导"包括Visual C++ 6.0的使用、12个精心设计的基本实验和11个综合上机练习。每个基本实验均包括实验目的、实验内容与要求、说明与提示等。每个综合上机练习均给出了详细说明和实验要求,练习内容涉及游戏、数学、数据管理等多个方面。第2部分"习题及解答"给出了C++、数据结构和软件工程各部分的习题及参考解答。本书习题是《计算机软件技术基础——C++、数据结构、软件工程》(第2版)一书中没有的。书后附有模拟试题及解答和ASCII码表。

本书除可与《计算机软件技术基础——C++、数据结构、软件工程》(第2版)一书配套使用外,也可作为各类人员学习、使用C++的参考书。对于报考计算机类研究生的读者还可作为C++程序设计的复习辅导材料。

图书在版编目(CIP)数据

计算机软件技术基础实验指导 习题及解答:C++、数据结构、软件工程/赵国瑞主编.—天津:天津大学出版社,2002.9

ISBN 7-5618-1635-9

Ⅰ.计… Ⅱ.赵… Ⅲ.①C语言-程序设计-高等学校-教学参考资料②数据结构-高等学校-教学参考资料③软件工程-高等学校-教学参考资料 Ⅳ.TP31

中国版本图书馆CIP数据核字(2002)第052768号

出版发行	天津大学出版社
出 版 人	杨风和
地　　址	天津市卫津路92号天津大学内(邮编:300072)
电　　话	发行部:022-27403647　邮购部:022-27402742
印　　刷	天津市宝坻区第二印刷厂
经　　销	全国各地新华书店
开　　本	185mm×260mm
印　　张	15.5
字　　数	405千
版　　次	2002年9月第1版
印　　次	2002年9月第1次
印　　数	1—5 000
定　　价	21.00元

前　言

计算机软件技术基础是一门实践性非常强的课程。要获得利用计算机、编写程序解决实际问题的能力，仅靠阅读教材和听课是不够的，还有一个重要的环节是亲自在计算机上实践。

很多同学反映，通过阅读教材和听课对内容能基本理解，但是，当自己编写解决实际问题的程序时却感到无从下手。对于老师反复强调、举例说明的某些语法规则和注意事项，在实际上机时仍会频繁违反，造成程序多处语法错误或运行异常。甚至对一个简单的程序，花上几个小时也调不通。出现这些现象实际上很正常。所有学习或从事计算机编程工作的人员都有类似的经历。要把书本上的知识真正转化成自己的技能，必须通过大量的实践。

本书作为与《计算机软件技术基础 —— C++、数据结构、软件工程》(第2版)配套使用的实验教材和学习辅导书，目的是使读者在实践过程中少走些弯路，多享受些成功的乐趣。

本书内容分为两部分。第1部分“实验指导”，首先介绍了 Visual C++ 6.0 开发环境及其使用方法，然后按照教材中各部分内容的顺序精心设计了12个基本实验，每个实验均包含实验目的、实验内容与要求和说明与提示，使读者通过这些实验可以深入理解和熟练掌握书中的内容。最后特别编写了11个综合上机练习。每个练习均涉及到数据结构选择、外部文件操作以及简单菜单设计、查找排序、统计计算等在实用程序中所面临的基本问题，以便检验读者在学习教材的全部内容后应用计算机解决小型实际问题的能力。每个综合上机练习可作为“大作业”由几个同学共同协作完成。第2部分“习题及解答”分别给出了各类不同形式的习题，包括单项选择题、阅读程序写运行结果题、填空完善程序题、简答题和编程序题，共三百余道。此外，书末还有两个附录。附录A给出了四套模拟试题及解答，附录B是ASCII码表。

参加本书编写的有陆明、王保旗、赵国瑞、汪大菊、葛卫民、张英杰、李鹰，最后由赵国瑞审阅第1部分，陆明审阅第2部分。

由于编者水平有限，错误和不当之处敬请读者批评指正。编者的电子邮件地址为 guoruizhao@163.net。

编者

2002年5月

目　录

第1部分　实验指导

第2部分　习题及解答

第1部分 实验指导

第 1 章 Visual C++6.0的使用

Visual C++6.0 是 Microsoft 公司于 1998 年推出的基于 Windows 9x 和 Windows NT 的优秀集成开发环境。面向对象的程序设计方法与可视化开发环境的完美结合,使 Visual C++6.0 深受广大软件开发者的喜爱。Visual C++ 不仅是 C++ 语言的集成开发环境,而且还是开发 32 位 Windows 应用程序的强大工具。它所提供的一系列功能丰富的编程工具,使编写在 Windows 环境下运行的应用程序十分简捷、快速;同时,基于类库的设计方法又使编写面向对象的程序变得更为简单。另外,与 Visual Basic 相比,用 Visual C++ 开发出的应用程序的代码效率更高。

Visual C++6.0 软件包中包含了许多独立的组件,例如,编辑器、编译器、连接器、调试器以及各种各样的辅助开发工具。在 Visual C++ 的软件包中还包含着一个名为 Developer Studio (开发者工作室)的集成应用程序。Developer Studio 是访问 Visual C++ 提供的开发工具的主要界面。它把 Visual C++ 的各种工具结合在一起,集成为一个有机的整体,通过一个由窗口、菜单、对话框、工具栏以及快捷键组成的和谐系统,使用户可以比较轻松地监控应用程序的创建过程。

如果计算机上已经安装了 Visual C++6.0,可通过用鼠标单击"开始|程序|Microsoft Visual Studio 6.0|Microsoft Visual C++6.0"启动 Visual C++6.0(图 1.1)。此时,屏幕上显示图 1.2 所示的 Visual C++6.0 的主窗口。由于设置不同,看到的窗口与图 1.2 所示的窗口可能有些差别。

下面简要介绍在 Visual C++6.0 中开发应用程序的过程。

1.1 控制台应用程序的生成

为使大家能集中精力完成有关 C++ 语言方面的练习,而不去考虑编写 Windows GUI (Graphic User Interface,图形用户界面)程序时的一些复杂情况,下面先介绍在 Visual C++ 下生成简单的控制台(console)应用程序,即生成字符方式应用程序的过程。

在 Visual C++ 中,可以使用编写字符方式程序时常用的技术编写控制台应用程序。具体地说,在程序中可以使用 C++ 的 iostream 类库的成员函数读取输入的数据和显示输出等。对于一个编写好的 C++ 源程序,要想在 Visual C++ 环境下运行,需要经过以下步骤。

1.1.1 生成项目

现在的应用程序,尤其是 Windows 应用程序,一般由多个文件组成,其中包括源文件、头文

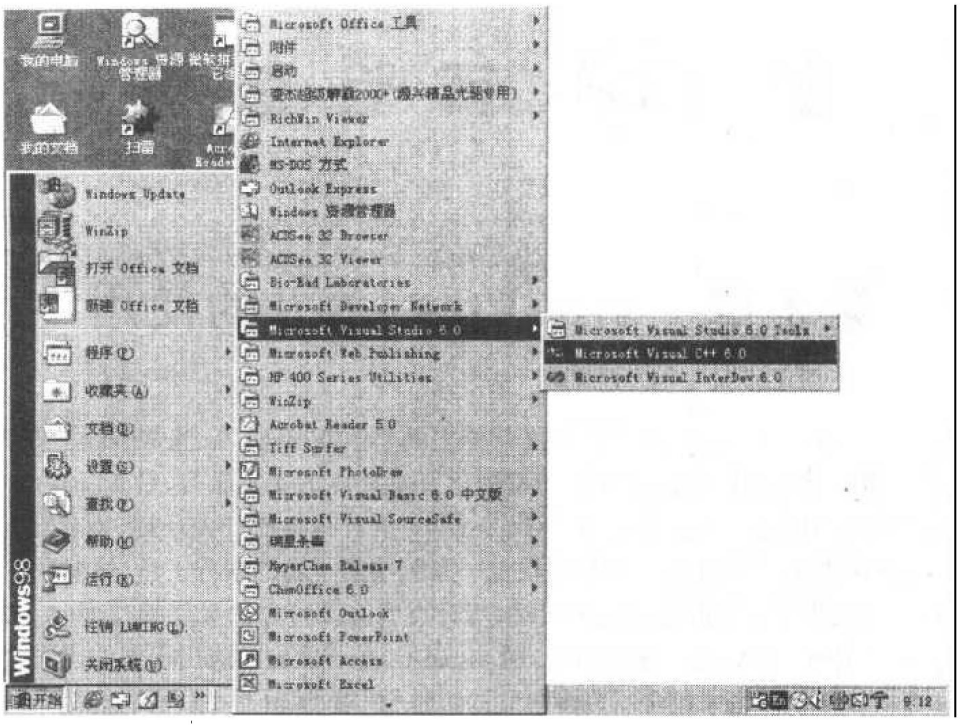

图 1.1 启动 Visual C++6.0

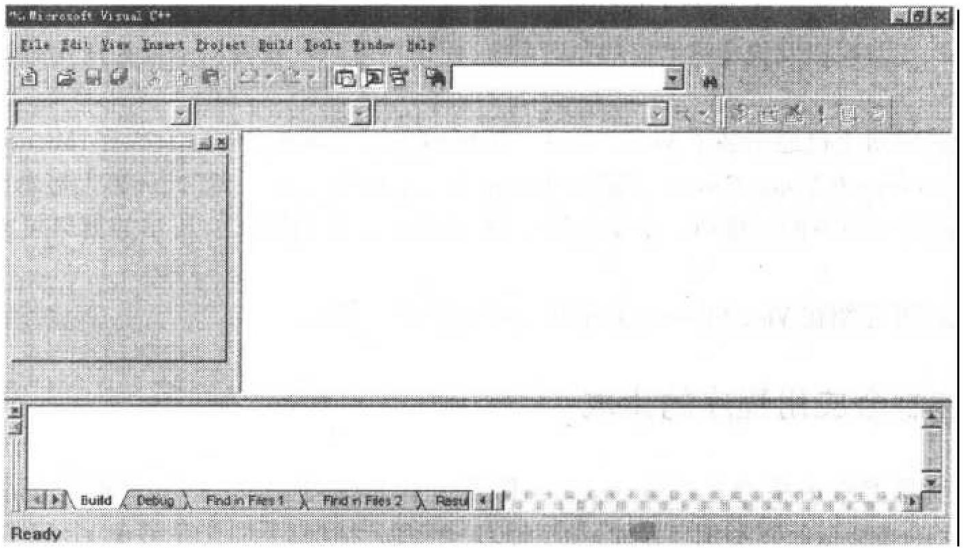

图 1.2 Visual C++6.0 的主窗口

件、资源文件等。为使组成程序的所有文件能够形成一个有机的整体,引入了项目的概念,即把一个应用程序作为一个项目。项目能够自动地将其包含的文件进行分类、管理,这就大大减轻了程序员的负担。

在 Visual C++ 中,项目又置于项目工作区(Workspace)的管理之下。一个项目工作区可以管理多个项目,甚至是不同类型的项目。这些项目之间相互独立,但共用一个项目工作区的环

境设置。Visual C++ 中,文件、项目以及项目工作区之间的关系如图 1.3 所示。

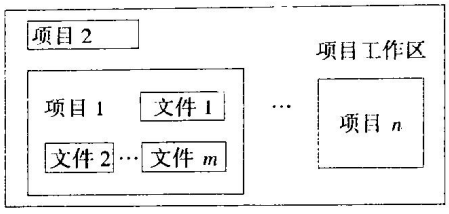

图 1.3　文件、项目、项目工作区之间的关系

为简单起见,本章介绍的所有程序实例都使用了只包含一个项目的项目工作区。现有一个名为 Welcome 的 C++ 程序。该程序的程序代码如下:

```
//Welcome. cpp: Welcome 程序的 C++ 源代码
# include < iostream. h >
char Name[16];
void main()
{
    cout << "Enter your name:";
    cin. getline (Name, sizeof(Name));
    cout << " \ nWelcome, " << Name << " \ n";
    cout << " \ nPress Enter to continue ...";
    cin. get();
}
```

为生成以上程序所需要的项目文件,可按以下步骤工作。

①执行 Visual C++ 主窗口中的"File|New"菜单命令,或按 Ctrl + N,打开 New 对话框,如图 1.4 所示。

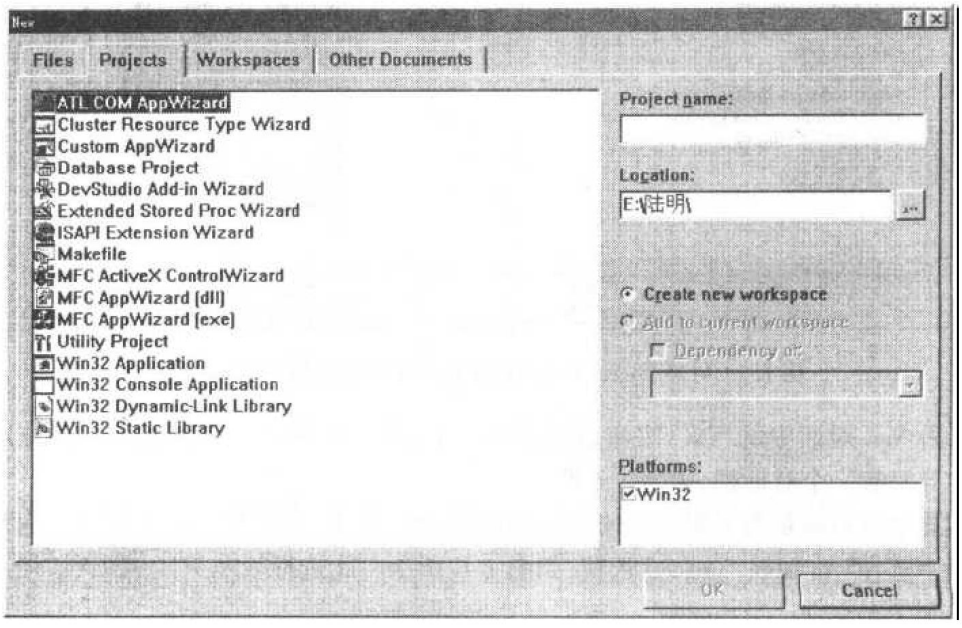

图 1.4　New 对话框

②选择 New 对话框中的 Projects 标签，以便生成新项目。在生成新项目时，Developer Studio 将自动生成项目工作区，并将新项目加入其中。

③在 Projects 标签的项目类型列表框中选择“Wind32 Console Application”（Win32 控制台应用程序）。

④在“Project Name:”文本框中输入项目名 Welcome，指示 Developer Studio 把新项目（以及该项目所在的项目工作区）取名为 Welcome。

⑤在“Location:”文本框中指定保存项目文件的文件夹（即项目文件夹）的路径。可以直接采用该文本框中给出的缺省文件夹路径，也可以单击文本框右侧带省略号（...）的按钮，搜索不同位置的项目文件夹，还可以直接输入项目文件夹的路径。如果指定的项目文件夹不存在，Developer Studio 会自动生成这个项目文件夹，同时还会在项目文件夹中生成一个或几个存放输出文件的子文件夹。

⑥在 Platforms 区中，检查是否选择了“Win32”。Win32 是 Windows 9x 和 Windows NT 下的 GUI 应用程序和控制台应用程序的 32 位 API 接口（Application Programming Interface）。选择 Win32 表示应用程序由 Win32 支持、可以调用 Win32 函数、能够作为 32 位程序在 Windows 9x 和 Windows NT 上运行。（经过以上步骤后，New 对话框中 Projects 标签的设置情况如图 1.5 所示。）

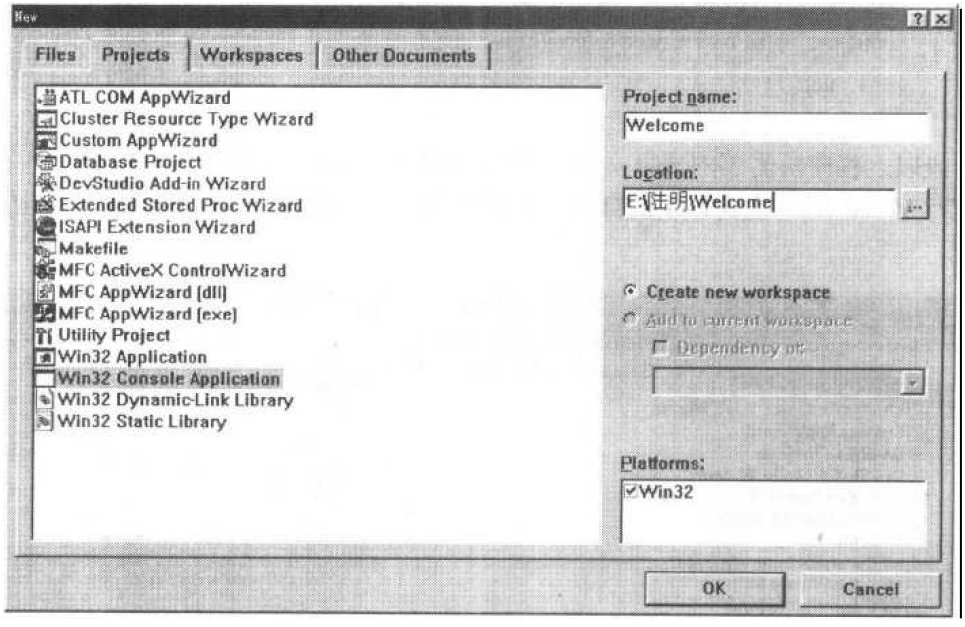

图 1.5 针对 Welcome 程序设置的 New 对话框的 Projects 标签

⑦单击 New 对话中的“OK”按钮，这时 Developer Studio 将显示 Win32 控制台应用程序向导。该向导只有一个对话框，如图 1.6 所示。

⑧本练习的目的是为了学习如何自己生成源程序文件，所以在 Win32 控制台应用程序向导对话框中选择“An empty project”选项，如图 1.6 所示。此选择表示只生成项目，但不向其中加入任何源代码文件。当然，也可以根据需要选择其他选项。

⑨单击 Win32 控制台应用程序向导对话框底部的“Finish”按钮，此时屏幕上出现 New

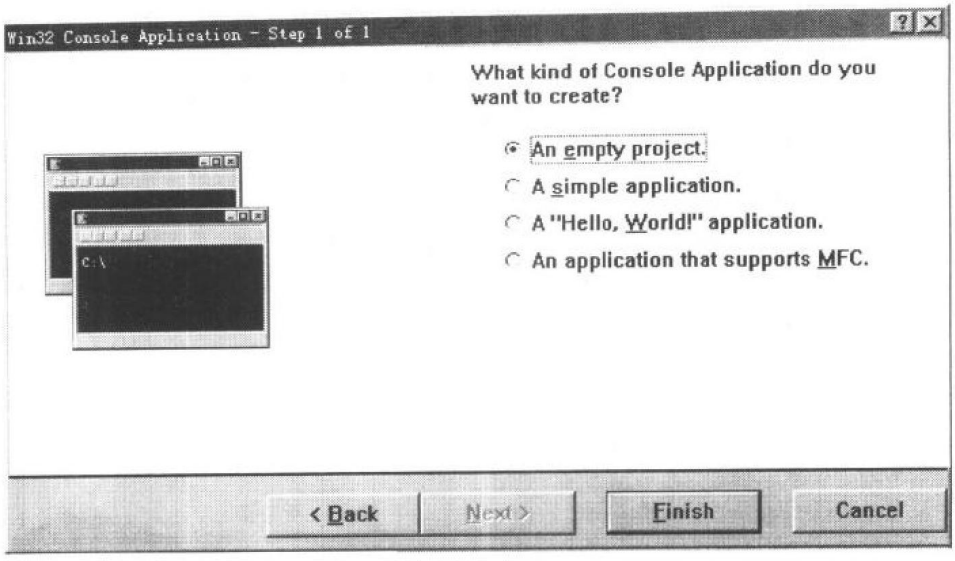

图 1.6 Win32 控制台应用程序向导

Project Information 对话框,在该对话框中显示了将要建立的新项目的基本信息,如图 1.7 所示。

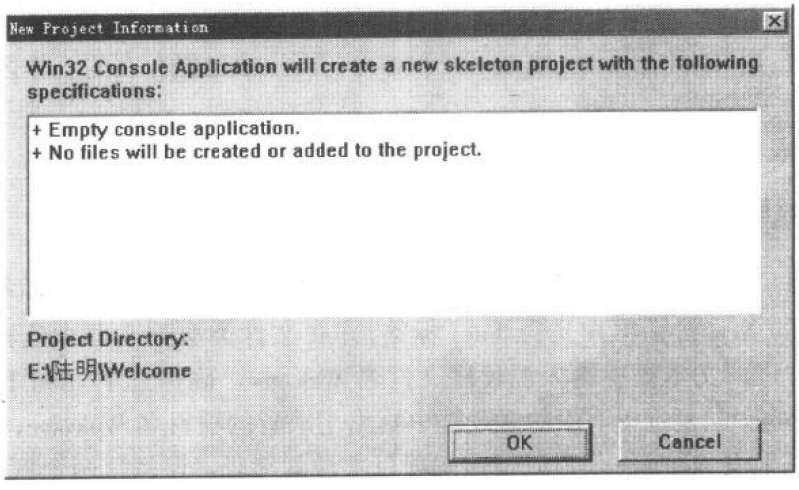

图 1.7 Welcome 项目的 New Project Information 对话框

⑩单击 New Project Information 对话框下面的"OK"按钮,屏幕显示将切换到 Visual C++ 6.0 的主窗口。此时,Developer Studio 已经生成并打开了新的项目工作区 Welcome。在 Welcome 项目工作区中只包含了一个项目,名称也是 Welcome。与项目有关的信息会在工作区(Workspace)窗口中显示,如图 1.8 所示。如果屏幕上没有出现工作区窗口,可执行"View | Workspace"菜单命令将其打开。

在工作区窗口中包含了两个标签,即 ClassView 和 FileView(如果生成的是一个 Windows GUI 程序,那么在工作区窗口中还会增加一个 ResourceView 标签)。它们显示了与项目有关的各种信息。关于在各标签中显示的信息的意义将在后续章节介绍。

可以用"File | Save Workspace"菜单命令保存当前工作区;用"File | Close Workspace"菜单命令

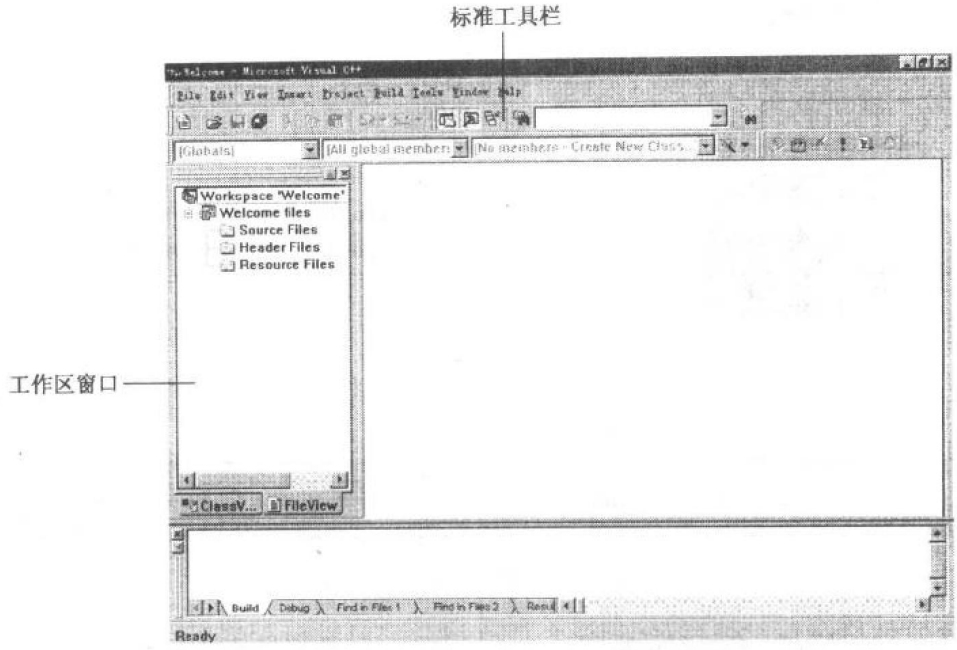

图 1.8 Welcome 项目的工作区窗口

关闭当前工作区;用"File | Open Workspace"菜单命令打开一个已经存在的工作区。工作区文件的扩展名是.dsw。

至此,为生成项目而需要做的工作已经全部完成。

1.1.2 生成和编辑源程序文件

在生成项目后,下一步是生成和编辑源程序文件 Welcome.cpp。为此,执行"File | New"菜单命令;在打开的 New 对话框中选择 Files 标签,并在文件类型列表框中选择"C ++ Source File"项,然后在"File"文本框中输入源程序文件名 Welcome。确保选中"Add To Project:"复选框,在"Add To Project:"复选框下面的组合框中给出了缺省的项目名 Welcome,在"Location:"文本框中显示的是生成 Welcome 项目时指定的项目文件的路径。保持这两个框中的内容不变,完成的 New 对话框的 Files 标签如图 1.9 所示。

单击 New 对话框中的"OK"按钮后,Developer Studio 将生成一个名为 Welcome.cpp 的源程序文件,并将此源文件加入到 Welcome 项目中,同时在文本编辑器窗口将该源文件打开。可以在文本编辑器窗口中输入编写好的 Welcome 源程序的代码,结果如图 1.10 所示。

在 Welcome 程序运行时,程序中最后一条语句的作用是暂停程序的执行,等待用户按下 Enter 键。这样做的目的是使用户在关闭程序窗口前能够看到程序的输出结果。如果程序是在 Developer Studio 中运行,这条语句是多余的。因为 Developer Studio 会自动安排在程序执行后保持程序窗口的打开状态,直到用户按下任意键为止。如果程序是在 Developer Studio 外运行或在 Developer Studio 调试器中运行,则这条语句是必要的。因为在此情况下,程序执行结束后程序窗口会自动关闭。

源代码输入完毕后,可执行"File | Save"菜单命令或单击标准工具栏中的"Save"按钮,保存

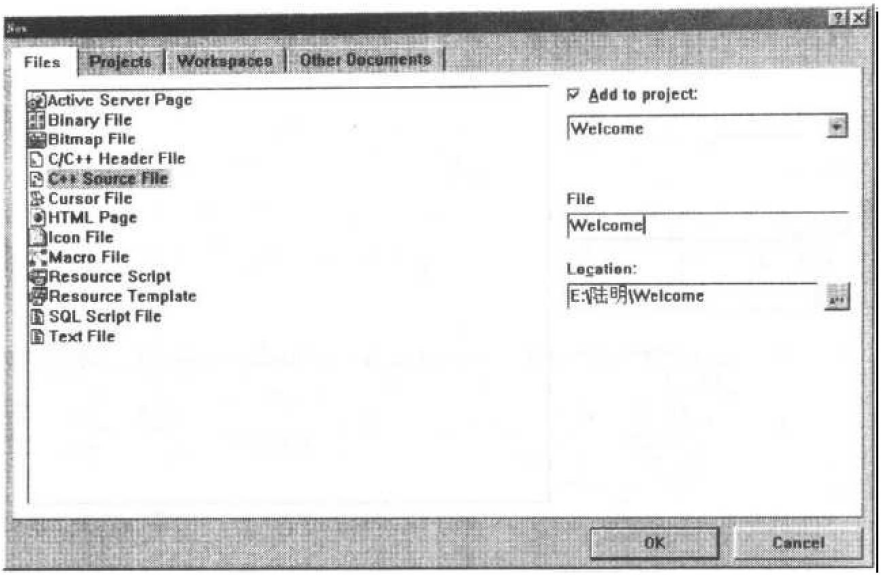

图 1.9　完成的 New 对话框的 Files 标签

输入的内容。

1.1.3　查看类及文件

在工作区窗口的 FileView 标签中，以层次图方式显示了当前工作区中的项目以及项目中包含的文件。可单击各分支顶部节点旁的“+”(或“-”)标记，以展开或折叠层次图的分支。展开的层次图如图 1.10 所示。

工作区窗口中的 FileView 标签

图 1.10　输入 Welcome 程序代码

在 FileView 标签的层次图中，双击其中的某个文件名，可以将这个文件打开或在编辑窗口中显示这个文件的内容。这是一种在包含多个源文件的项目中查看某个指定文件的简单方法。

在工作区窗口的 ClassView 标签中也显示了一个层次图，如图 1.11 所示。Globals 文件夹中列出了在 Welcome 程序中定义的所有全局符号。对于一个包含类的 C++ 程序，在 ClassView 标签的层次图中还将显示该程序定义的所有类及其成员。如果双击 ClassView 标签中的某个符号（如 Welcome 程序的 main()或 Name），Developer Studio 会显示定义该符号的源文件，并将插入点放在定义处。

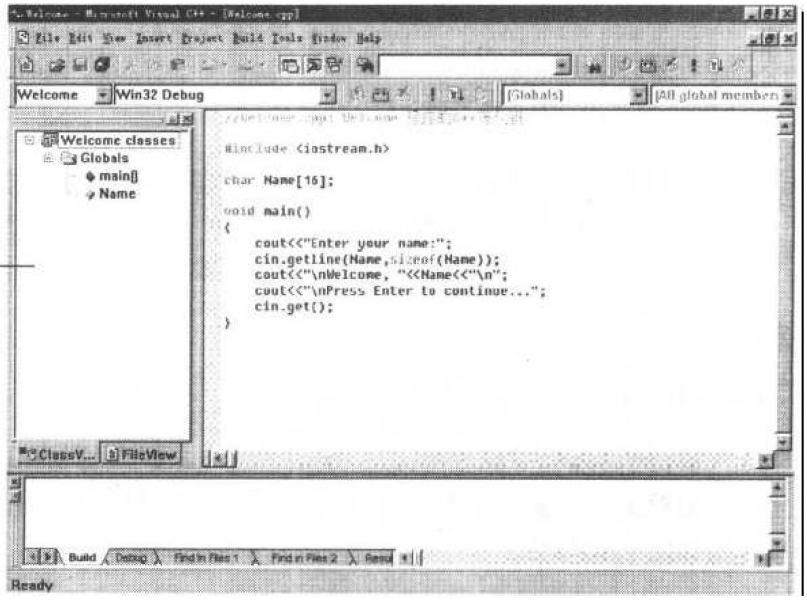

图 1.11 工作区窗口中的 ClassView 标签

1.1.4 访问联机帮助

如果想更全面地了解 Visual C++ 和 Windows 的工作机制，或想在开发过程中获得帮助，应启动 Visual C++ 6.0 的联机帮助。

Visual C++ 6.0 的帮助系统做得非常详尽。访问 Visual C++ 帮助系统的方法主要有两种：一是从“Help”菜单中选择执行 Contents、Search 或 Index 命令；二是在文本编辑窗口中把光标定位在一个需要查询的单词处，然后按 F1 键。打开的 Visual C++ 6.0 的联机帮助窗口如图 1.12所示。

Visual C++ 6.0 的帮助系统很像微软的 Internet Explorer，可以在此浏览以 HTML 文件组成的静态帮助文件。此外，帮助系统还提供了全文搜索功能。当用户在搜索标签中输入要搜索的单词时，帮助系统将在所有的帮助文件中进行全文搜索。同时，Visual C++ 6.0 的帮助系统通过“活动子集”下拉列表框还提供了很好的文件分类，用户可以缩小搜索范围。

在使用联机帮助前，必须保证已正常装入了 MSDN（Microsoft Developer Network）。

图 1.12　Visual C++ 的联机帮助窗口

1.1.5　改变项目配置

Developer Studio 中的项目可以产生两种可执行代码：一种是 Win32 Debug(调试版)；另一种是 Win32 Release(发布版)。

在调试版中存在着大量的符号信息。这些信息用于记录程序中函数和变量的名字及其在源代码中的位置。通过这些符号信息，调试器可以将源代码中的每一行与可执行代码中的相应指令联系起来，以便于用户检测程序中的错误。调试版的体积较大，运行速度通常比发布版要低。它主要在开发过程中使用。

发布版只包含了经过代码优化的可执行代码，其中没有符号信息。正因如此，它不能用于调试程序。当认为程序可以为用户使用时，可以提供发布版。

在系统生成可执行代码之前，可以改变影响文件编译方式的项目设置。通常情况下，系统为项目创建的缺省项目配置中，所指定的设置对于大多数项目而言是足够的，但如果需要对项目设置进行更改，可以执行“Project | Setting...”菜单命令，在打开的 Project Setting 对话框中完成设置。

1.1.6　建立目标程序

下面说明如何建立 Welcome 程序的调试版。

如果 Welcome 项目还未打开，应执行“File | Open Workspace”命令将它打开。在建立目标程序时，Developer Studio 总是使用当前活动的项目配置。所以，在发出建立目标程序的命令之前，应保证 Welcome 项目的当前活动项目配置为 Win32 Debug。为此，可执行“Build | Set Active Configuration”菜单命令，在弹出的 Set Action Configuration 对话框中选择 Win32 Debug，然后单击“OK”按钮。也可以通过 Build 工具栏中的“Active Configuration”下拉列表框进行设置，如图 1.13

所示。如果窗口中没有出现 Build 工具栏，可在菜单栏或某个工具栏上单击鼠标右键，并从弹出的快捷菜单中选择 Build，打开 Build 工具栏。

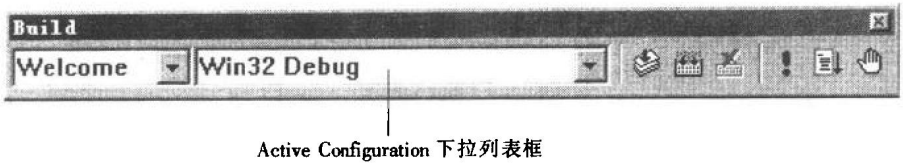

图 1.13 Build 工具栏

图 1.14 Build MiniBar 工具栏

要建立目标程序，可执行“Build | Build Welcome.exe”菜单命令，或单击 Build 工具栏或 Build MiniBar 工具栏中的 Build 按钮，如图 1.14 所示。

Developer Studio 在建立目标程序时将自动弹出 Output 窗口（如果它原来没有出现的话），并在此窗口中显示目标程序建立过程中的一些相关信息，包括错误和警告信息。当建立目标程序的工作完成后，无论成功与否，Developer Studio 都会发生“嘟”的声响，并在 Output 窗口中显示错误信息和警告信息的总数。假设在 Output 窗口中显示的是“Welcome.exe—0 error(s)，0 Warning(s)”，表示程序中没有错误和警告，目标程序的建立工作顺利完成。

在生成目标程序时，Developer Studio 会在项目文件夹中建立子文件夹 Debug，用来存放输出文件（Welcome.obj，Welcome.exe 等）。如果还要基于发布配置建立目标文件，则 Developer Studio 会在项目文件夹中建立子文件夹 Release，并在此文件夹中存放发布版的输出文件。由此可见，基于调试版和发布版生成的输出文件被分别存放在不同的地方，它们不会相互覆盖。如果要改变存放输出文件的文件夹，只要执行“Project | Setting...”菜单命令，并在打开的 Project Setting 对话框的 General 标签中指定中间输出文件（例如，Welcome.obj）和最终输出文件（例如，Welcome.exe）所在的文件夹即可。

1.1.7 运行程序

在顺利生成项目的可执行代码后，可通过执行“Build | Execute Welcome.exe”菜单命令直接在 Developer Studio 中运行它，也可以在 Developer Studio 之外用任何标准的 Windows 方法运行可执行文件 Welcome.exe。作为控制台应用程序，Welcome 程序将在控制台窗口中运行。控制台窗口与 Windows 中运行的 MS-DOS 程序的窗口相似，如图 1.15 所示。

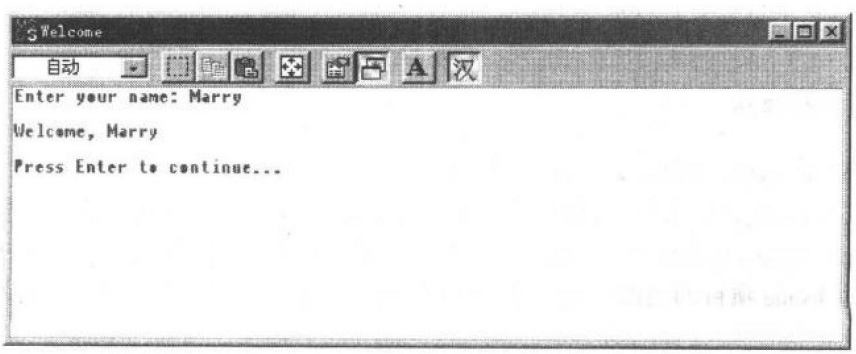

图 1.15 运行 Welcome.exe 程序

上述建立目标程序和运行程序也可以通过单击“Build”工具栏或“Build MiniBar”工具栏中的“!”工具按钮或按下 Ctrl + F5 键完成。

1.1.8 项目

(1)把程序分解成多个源文件

为便于编辑和编译大程序,可将大程序分解并存储于多个源文件中。至于如何分解,并没有确定的规则。根据经验,把程序中用于实现共同任务或具有共同特性的函数放到一个源文件中是最方便的。例如,如果程序中有一些显示菜单的函数,可把它们存放到一个名为 MENU.CPP 的文件中;如果程序中有一些专用的图形函数,可以把它们放到一个名为 GRAPHICS.CPP 的文件中。另外常把类的定义放到一个.h 文件中,而把该类成员函数的实现放在一个.cpp 文件中。

当把一个程序分解为多个单独的源文件(或模块)时,如果想在一个模块中调用在另外一个模块中定义的函数,那么,在调用前必须说明函数的原型。也可以为每一个今后要被其他模块使用的源文件建立一个头文件。例如,为 MENU.CPP 建立头文件 MENU.H,为 GRAPHICS.CPP 建立头文件 GRAPHICS.H,在头文件中说明函数原型并定义可被其他文件使用的常量标识符、宏和变量。这样,在使用这些模块的程序中,只需用 # include 预处理命令将相应的头文件包含进去,编译程序就可以知道定义在外部模块中的变量的类型并且可以建立起正确的函数调用。下面的程序被分解为 main.cpp、source1.cpp、source2.cpp 三个源文件。

main.cpp 文件的内容为:

```
void source1();          //函数原型说明
void source2();          //函数原型说明
void main(){
    source1();
    source2();
}
```

source1.cpp 的内容为:

```
# include <iostream.h>
void source1(){
    cout <<"I am source1." << endl;
}
```

source2.cpp 的内容为:

```
# include <iostream.h>
void source2(){
    cout << "I am source2." << endl;
}
```

为使一个应用程序的所有文件形成一个有机的整体,在 Visual C++ 中引入了“项目”(Project)这一概念。项目是应用程序开发的基本单位,实质是一些相互关联的源文件的集合。项目中的每个源文件(以.cpp 为扩展名)可被单独编译产生.OBJ 目标文件,通过对这些目标文件的连接,最终形成可执行的.EXE 文件。