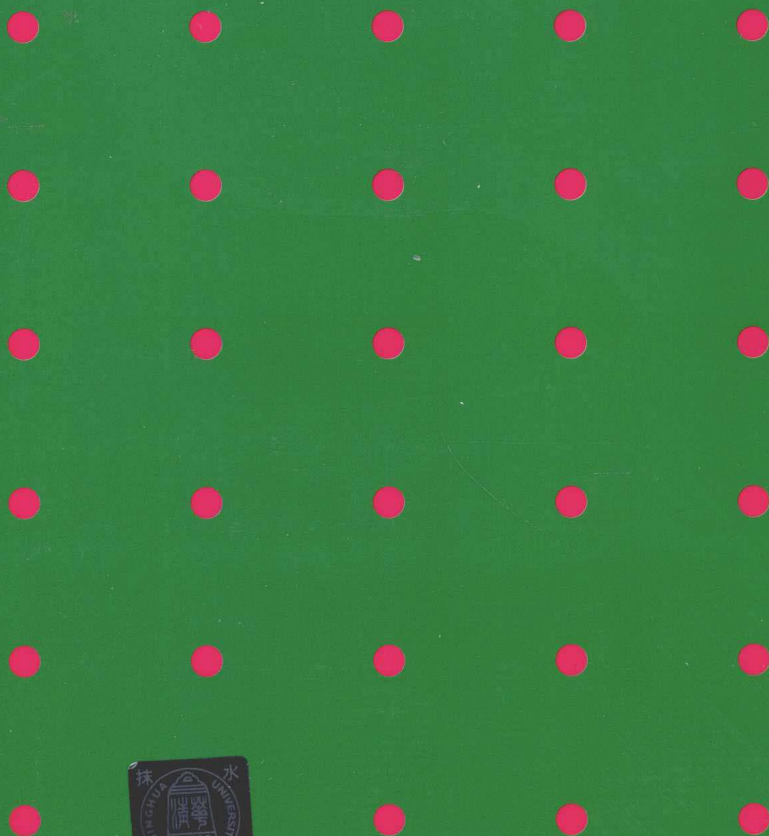


普通高校本科计算机专业特色教材精选·算法与程序设计

MFC Windows应用程序设计 习题解答及上机实验(第3版)

任哲 李益民 车进辉 编著



清华大学出版社



普通高校本科计算机专业特色教材精选 · 算法与程序设计

MFC Windows应用程序设计 习题解答及上机实验 (第3版)

任 哲 李益民 车进辉 编著



清华大学出版社
北京

内 容 简 介

本书是《MFC Windows 应用程序设计(第3版)》的配套教材。书中对《MFC Windows 应用程序设计(第3版)》的习题做了详细的解答并提供了一些上机实验题目,力图通过这些练习和训练使读者能较好地理解和掌握 MFC Windows 应用程序框架及使用 MFC 类库编程的基本方法。在书后的附录 Visual C++ 开发环境介绍部分,简明扼要地介绍了创建应用程序框架的方法及使用调试开发工具的方法,为读者快速地使用该开发工具提供了帮助。

本书适合作为高等学校和高等职业学校计算机专业的教学用书,同时也适合其他有一定 C++ 基础的读者,它可以在较短的时间内使读者了解和掌握开发 MFC Windows 应用程序的特点和设计方法。本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

MFC Windows 应用程序设计习题解答及上机实验/任哲,李益民,车进辉编著.--3版.--北京:清华大学出版社,2013.5

普通高校本科计算机专业特色教材精选·算法与程序设计

ISBN 978-7-302-31229-1

I. M… II. ①任… ②李… ③车… III. ①C语言—程序设计—高等学校—教学参考资料
②Windows 操作系统—程序设计—高等学校—教学参考资料 IV. ①TP312 ②TP316.7

中国版本图书馆 CIP 数据核字(2013)第 002190 号

责任编辑:汪汉友

封面设计:傅瑞学

责任校对:李建庄

责任印制:沈 露

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦 A 座 邮 编:100084

社 总 机:010-62770175 邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质量反馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

课件下载: <http://www.tup.com.cn>, 010-62795954

印 装 者:北京鑫海金澳胶印有限公司

经 销:全国新华书店

开 本:185mm×260mm 印 张:7.75

字 数:192 千字

版 次:2004 年 6 月第 1 版 2013 年 5 月第 3 版

印 次:2013 年 5 月第 1 次印刷

印 数:1~3000

定 价:19.00 元

产品编号:047601-01

出版说明

INTRODUCTION

在我国高等教育逐步实现大众化后，越来越多的高等学校将会面向国民经济发展的第一线，为行业、企业培养各级各类高级应用型专门人才。为此，教育部已经启动了“高等学校教学质量和教学改革工程”，强调要以信息技术为手段，深化教学改革和人才培养模式改革。如何根据社会的实际需要，根据各行各业的具体人才需求，培养具有特色显著的人才，是我们共同面临的重大问题。具体地说，培养具有一定专业特色的和特定能力强的计算机专业应用型人才则是计算机教育要解决的问题。

为了适应 21 世纪人才培养的需要，培养具有特色的计算机人才，急需一批适合各种人才培养特点的计算机专业教材。目前，一些高校在计算机专业教学和教材改革方面已经做了大量工作，许多教师在计算机专业教学和科研方面已经积累了许多宝贵的经验。并将他们的教研成果转化为教材的形式，向全国其他学校推广，而这对于深化我国高等学校的教学改革是一件十分有意义的事情。

清华大学出版社在经过大量调查研究的基础上，决定组织出版一套“普通高校本科计算机专业特色教材精选”。本套教材是针对当前高等教育改革的新形势，以社会对人才的需求为导向，主要以培养应用型计算机人才为目标，立足课程改革和教材创新，广泛吸纳全国各地的高等院校计算机优秀教师参与编写，从中精选出版确实反映计算机专业教学方向的特色教材，供普通高等院校计算机专业学生使用。

本套教材具有以下特点：

1. 编写目的明确

本套教材是在深入研究各地各学校办学特色的基础上，面向普通高校的计算机专业学生编写的。学生通过本套教材，主要学习计算机科学与技术专业的基本理论和基本知识，接受利用计算机解决实际问题的基本训练，培养研究和开发计算机系统，特别是应用系统的基本能力。

2. 理论知识与实践训练相结合

根据计算学科的3个学科形态及其关系,本套教材力求突出学科的理论与实践紧密结合的特征,结合实例讲解理论,使理论来源于实践,又进一步指导实践。学生通过实践深化对理论的理解,更重要的是使学生学会理论方法的实际运用。在编写教材时突出实用性,并做到通俗易懂,易教易学,使学生不仅知其然,知其所以然,还要会其如何然。

3. 注意培养学生的动手能力

每种教材都增加了能力训练部分的内容,学生通过学习和练习,能比较熟练地应用计算机知识解决实际问题。既注重培养学生分析问题的能力,也注重培养学生解决问题的能力,以适应新经济时代对人才的需要,满足就业要求。

4. 注重教材的立体化配套

大多数教材都将陆续配套教师用课件、习题及其解答提示,学生上机实验指导等辅助教学资源,有些教材还提供能用于网上下载的文件,以方便教学。

由于各地区各学校的培养目标、教学要求和办学特色均有所不同,所以对特色教学的理解也不尽一致,我们恳切希望大家在使用教材的过程中,及时地给我们提出批评和改进意见,以便我们做好教材的修订改版工作,使其日趋完善。

我们相信经过大家的共同努力,这套教材一定能成为特色鲜明、质量上乘的优秀教材,同时,我们也希望通过本套教材的编写出版,为“高等学校教学质量和教学改革工程”做出贡献。

清华大学出版社

前言

PREFACE

目前,各高等学校已经把 C/C++ 列为理工科专业学生的必修课程,这门课程使学生初步了解和掌握了面向对象程序设计的思想和方法,这无疑为学生学习和掌握 MFC 打下了良好的基础。如果在此基础上开设 MFC 课程,除了可以使学生掌握 Windows 应用程序设计的基本方法之外,又可以使学生更进一步深刻、全面地理解面向对象程序设计的思想,把握住计算机程序设计方法的发展方向,从而为今后进一步提高计算机程序设计能力打下坚实基础。

本书作为《MFC Windows 应用程序设计(第 3 版)》一书的配套教材,除了提供了《MFC Windows 应用程序设计(第 3 版)》教材中各章习题的解答外,还设计了一些课后上机实验,学生通过这些必要的练习可以较快地了解 MFC 的框架并掌握使用 MFC 类库进行 Windows 应用程序设计的一般方法。

为使学生可以较快地熟悉 Visual C++ 开发环境中的众多开发工具,书后还有选择地介绍了 Visual C++ 部分开发工具的使用方法。

本书在编写中参考了大量的相关文献,并引用了其中的一些例题和内容,在此对这些文献的作者表示诚挚的谢意。

参加本书编写工作的有任哲、李益民、车进辉。由于作者水平有限,恳请读者对书中的缺点和错误提出批评和指正,欢迎读者与作者联系。电子邮箱:renzhe71@sina.com.cn

作 者

2013 年 2 月

目 录

CONTENTS

第 1 章	Windows 应用程序基础知识习题解答及上机实验	1
1.1	习题解答	1
1.2	上机实验	6
第 2 章	Windows 应用程序的类封装习题解答及上机实验	13
2.1	习题解答	13
2.2	上机实验	13
第 3 章	MFC 的程序框架习题解答及上机实验	17
3.1	习题解答	17
3.2	上机实验	19
第 4 章	图形习题解答及上机实验	23
4.1	习题解答	23
4.2	上机实验	25
第 5 章	MFC 的通用类习题解答及上机实验	27
5.1	习题解答	27
5.2	上机实验	30
5.2.1	简单通用类的应用	30
5.2.2	群体类的应用	32
第 6 章	Windows 应用程序界面的设计习题解答及上机实验	35
6.1	习题解答	35
6.2	上机实验	36
6.2.1	实验 1	36
6.2.2	实验 2	37

第7章 鼠标和键盘习题解答及上机实验	39
7.1 习题解答	39
7.2 上机实验	49
第8章 资源习题解答及上机实验	53
8.1 习题解答	53
8.2 上机实验	62
第9章 MFC 的文件处理机制习题解答	65
第10章 控件习题解答	67
第11章 对话框习题解答及上机实验	69
11.1 习题解答	69
11.2 上机实验	74
第12章 进程与线程的管理习题解答	79
第13章 动态链接库习题解答	81
第14章 组件对象模型(COM)基础习题解答	83
第15章 ActiveX 应用基础习题解答	85
第16章 用 MFC 设计数据库应用程序习题解答	87
第17章 异常和异常处理习题解答	89
第18章 Visual C++ .NET 简介习题解答	91
附录 A Visual C++ 开发环境简介	93
A.1 Visual C++ 的用户界面	93
A.2 在 Visual C++ 中创建项目	94
A.2.1 设置工程项目的基数据	94
A.2.2 选择应用程序的类型	95
A.2.3 设置数据库支持	96
A.2.4 复合文档支持	96
A.2.5 设置应用程序特色	97

A. 2. 6	MFC 类库支持和注释	98
A. 2. 7	确认文件和类名称	99
A. 3	Visual C++ 的类辅助设计工具	100
A. 3. 1	使用 ClassWizard 对类进行操作	100
A. 3. 2	使用 ClassView 对类进行操作	103
A. 3. 3	使用 WizardBar 对类进行操作	107
A. 4	Visual C++ 的资源编辑器	107
A. 4. 1	菜单资源编辑器	107
A. 4. 2	对话框资源编辑器	108
A. 4. 3	加速键资源编辑器	108
A. 4. 4	图标资源编辑器	109
A. 5	Visual C++ 的主要调试工具	110
A. 5. 1	建立调试环境	110
A. 5. 2	调试窗口	110
A. 5. 3	用设置断点的方法调试应用程序	111
A. 5. 4	使用单步运行的方式调试应用程序	111
A. 5. 5	使用调试宏调试应用程序	112

Windows 应用程序基础知识

习题解答及上机实验

1.1 习题解答

1-1 什么是 Windows API 函数?

答: 用来开发 Windows SDK 应用程序的软件开发工具包是用 C 语言编写的一个大型函数库。这个库中的函数叫做 API 函数。

1-2 查看 windows.h 文件, 看一下 Windows 系统的句柄是什么数据类型的?

答: 整型。

1-3 试说明以下是什么句柄?

(1) HWND

(2) HINSTANCE

(3) HDC

答: HWND: 窗口句柄。

HINSTANCE: 应用程序实例句柄。

HDC: 图形设备环境句柄。

1-4 什么是事件? 试举例说明之。

答: 能触发程序作出相应反应的因素或动作叫做“事件”。例如在键盘上按下了一个键, 鼠标的单击或双击, 应用程序窗口的显示和销毁, 等等。

1-5 如何显示和更新窗口?

答: 调用函数 ShowWindow 显示窗口, 调用函数 UpdateWindow 更新窗口。

1-6 什么是消息循环?

答: 在创建了窗口的应用程序中, 应用程序将要不断地从消息队列里获取消息, 并将消息指派给指定的窗口处理函数来处理, 然后再回来从消息队列获取消息, 这个不断重复的工作过程叫做消息循环。

1-7 Windows 应用程序的主函数有哪 3 个主要任务?

答:注册窗口类、创建应用程序的窗口和建立消息循环。

1-8 说明 Windows 应用程序的主函数、窗口函数与 Windows 系统之间的关系。

答:Windows 应用程序的主函数和窗口函数都是系统调用的函数,主函数是在应用程序启动时由系统首先调用的函数,而窗口函数是主函数在消息循环中获得消息并把消息派送给系统之后,由系统调用的用来处理消息的函数。

1-9 在创建新 Win32 Application 工程时,在 Win32 Application Step-1 of 1 对话框中选中 A typical “Hello World!” Application 选项后并单击 Finish 按钮后,系统可以自动创建一个 Windows SDK 的示例程序,请运行该程序并分析这个应用程序的代码。

答:从程序设计向导产生的代码中可以看到,这是向导为用户产生的一个 Windows 应用程序框架代码,该程序仅有的业务便是在该程序运行之后,在窗口界面上显示一个字符串“Hello World!”。

目前,能分析和理解的就是.cpp 文件中的部分代码,现将其代码列写如下:

```
#include "stdafx.h"           //包含了头文件 stdafx.h,而该文件中的主要内容如下:
//-----
//Windows 头文件,该文件定义了大量的数据类型别名
//#include <windows.h>

//C 运行库头文件
//#include <stdlib.h>
//#include <malloc.h>
//#include <memory.h>
//#include <tchar.h>
//本程序的资源头文件
//#include "resource.h"
//-----

#define MAX_LOADSTRING 100

//以下定义了程序的全局变量:
HINSTANCE hInst;               //当前程序句柄
TCHAR szTitle[MAX_LOADSTRING]; //窗口标题栏题目字符数组
TCHAR szWindowClass[MAX_LOADSTRING];

//本程序的各个函数声明:
ATOM                MyRegisterClass(HINSTANCE hInstance);
BOOL                InitInstance(HINSTANCE, int);
LRESULT CALLBACK    WndProc(HWND, UINT, WPARAM, LPARAM);
LRESULT CALLBACK    About(HWND, UINT, WPARAM, LPARAM);

//入口函数(主函数)
int APIENTRY WinMain(HINSTANCE hInstance,
                     HINSTANCE hPrevInstance,
                     LPSTR     lpCmdLine,
```

```
int nCmdShow)

{
    MSG msg;
    HACCEL hAccelTable;

    //字符串初始化
    LoadString(hInstance, IDS_APP_TITLE, szTitle, MAX_LOADSTRING);
    LoadString(hInstance, IDC_BB, szWindowClass, MAX_LOADSTRING);
    MyRegisterClass(hInstance);

    //调用 InitInstance 初始化应用程序实例:
    if (!InitInstance (hInstance, nCmdShow))
    {
        return FALSE;
    }

    hAccelTable = LoadAccelerators(hInstance, (LPCTSTR)IDC_BB);

    //消息循环:
    while (GetMessage(&msg, NULL, 0, 0))
    {
        if (!TranslateAccelerator(msg.hwnd, hAccelTable, &msg))
        {
            TranslateMessage(&msg);
            DispatchMessage(&msg);
        }
    }

    return msg.wParam;
}

//
//注册窗口函数 MyRegisterClass()
//
ATOM MyRegisterClass(HINSTANCE hInstance)
{
    WNDCLASSEX wcex;

    wcex.cbSize = sizeof(WNDCLASSEX);

    wcex.style = CS_HREDRAW | CS_VREDRAW;
    wcex.lpfnWndProc = (WNDPROC)WndProc;
    wcex.cbClsExtra = 0;
    wcex.cbWndExtra = 0;
    wcex.hInstance = hInstance;
```

```

    wcex.hIcon          = LoadIcon(hInstance, (LPCTSTR)IDI_BB);
    wcex.hCursor        = LoadCursor(NULL, IDC_ARROW);
    wcex.hbrBackground = (HBRUSH)(COLOR_WINDOW+1);
    wcex.lpszMenuName   = (LPCSTR)IDC_BB;
    wcex.lpszClassName = szWindowClass;
    wcex.hIconSm        = LoadIcon(wcex.hInstance, (LPCTSTR)IDI_SMALL);

    return RegisterClassEx(&wcex);
}

//
//初始化应用程序实例函数 InitInstance(HANDLE, int)
//
BOOL InitInstance(HINSTANCE hInstance, int nCmdShow)
{
    HWND hWnd;

    hInst = hInstance;    //Store instance handle in our global variable

    hWnd = CreateWindow(szWindowClass,
        szTitle,
        WS_OVERLAPPEDWINDOW,
        CW_USEDEFAULT,
        0,
        CW_USEDEFAULT,
        0,
        NULL, NULL, hInstance, NULL);

    if (!hWnd)
    {
        return FALSE;
    }

    ShowWindow(hWnd, nCmdShow);
    UpdateWindow(hWnd);

    return TRUE;
}

//
//窗口函数 WndProc(HWND, unsigned, WORD, LONG)
//
LRESULT CALLBACK WndProc(HWND hWnd,
    UINT message, WPARAM wParam, LPARAM lParam)

```

```
{
    int wmId, wmEvent;
    PAINTSTRUCT ps;
    HDC hdc;
    TCHAR szHello[MAX_LOADSTRING];
    LoadString(hInst, IDS_HELLO, szHello, MAX_LOADSTRING);

    switch (message)
    {
        case WM_COMMAND:
            wmId=LOWORD(wParam);
            wmEvent =HIWORD(wParam);
            //以下代码用于窗口菜单条上两个菜单项的消息处理:
            switch (wmId)
            {
                case IDM_ABOUT:
                    DialogBox(hInst, (LPCTSTR)IDD_ABOUTBOX,
                        hWnd, (DLGPROC)About);
                    break;
                case IDM_EXIT:
                    DestroyWindow(hWnd);
                    break;
                default:
                    return DefWindowProc(hWnd, message,
                        wParam, lParam);
            }
            break;
        case WM_PAINT:                                //窗口用户区绘制消息
            hdc =BeginPaint(hWnd, &ps);
            RECT rt;
            GetClientRect(hWnd, &rt);
            //下面这行代码根据资源文件绘制了字符串"Hello World!"
            DrawText(hdc, szHello, strlen(szHello), &rt, DT_CENTER);
            EndPaint(hWnd, &ps);
            break;
        case WM_DESTROY:
            PostQuitMessage(0);
            break;
        default:
            return DefWindowProc(hWnd, message, wParam, lParam);
    }
    return 0;
}
```

```
//"关于"对话框中的消息循环
//(这段代码需要在对话框程序设计部分学习理解)。
LRESULT CALLBACK About (HWND hDlg, UINT message,
WPARAM wParam, LPARAM lParam)
{
    switch (message)
    {
        case WM_INITDIALOG:
            return TRUE;

        case WM_COMMAND:
            if (LOWORD(wParam) == IDOK || LOWORD(wParam)
                == IDCANCEL)
            {
                EndDialog(hDlg, LOWORD(wParam));
                return TRUE;
            }
            break;
    }
    return FALSE;
}
```

以上这些便是. cpp 文件中目前还能理解的代码,而其余包括字符串的显示等代码都需要在以后的学习中逐渐理解。

1-10 什么叫做可变代码?为什么要对可变代码进行隔离?试说明 Windows 所采用的代码隔离方法。

答:在一个设计良好的完整应用程序中,有些代码形成了程序的框架,它们与程序的具体任务关联较弱,通常不经修改或很少修改就可以应用于其他程序,因此具有一定的通用性且变化较小,称作框架代码或不可变代码。相对于程序框架代码,那些为完成程序具体功能而编写的代码叫做可变代码,因为当用户对程序的功能有新的要求时,这些代码必须要发生相应的变化时,故称为可变代码,也称业务代码。

为提高程序的可维护性,防止因可变代码的变化而引起框架代码的变化,从而波及整个程序结构,故在设计一个应用程序时,应尽量减少可变代码与框架代码之间的关联,即应采取措施对可变代码进行隔离。

在 Windows 程序中,多分支结构的各个 case 中的代码段就是可变代码段,Windows 把这些代码段都分别编写成了函数,并采用函数指针与这些函数进行关联,从而有效地把可变代码段与框架代码进行了隔离。

1.2 上机实验

实验内容:

在 Visual C++ 6.0 中创建 Win32 Application 工程。

实验目的：

- (1) 熟悉在 Visual C++ 6.0 中创建 Win32 Application 的过程。
- (2) 熟悉 Visual C++ 6.0 可以创建的 3 种 Win32 Application。
- (3) 在 Visual C++ 6.0 中查看帮助文件。

实验步骤：

- (1) 选择【开始】|【程序】|Microsoft Visual Studio 6.0|Microsoft Visual C++ 6.0 命令启动 Visual C++ 6.0, 见图 1-1。

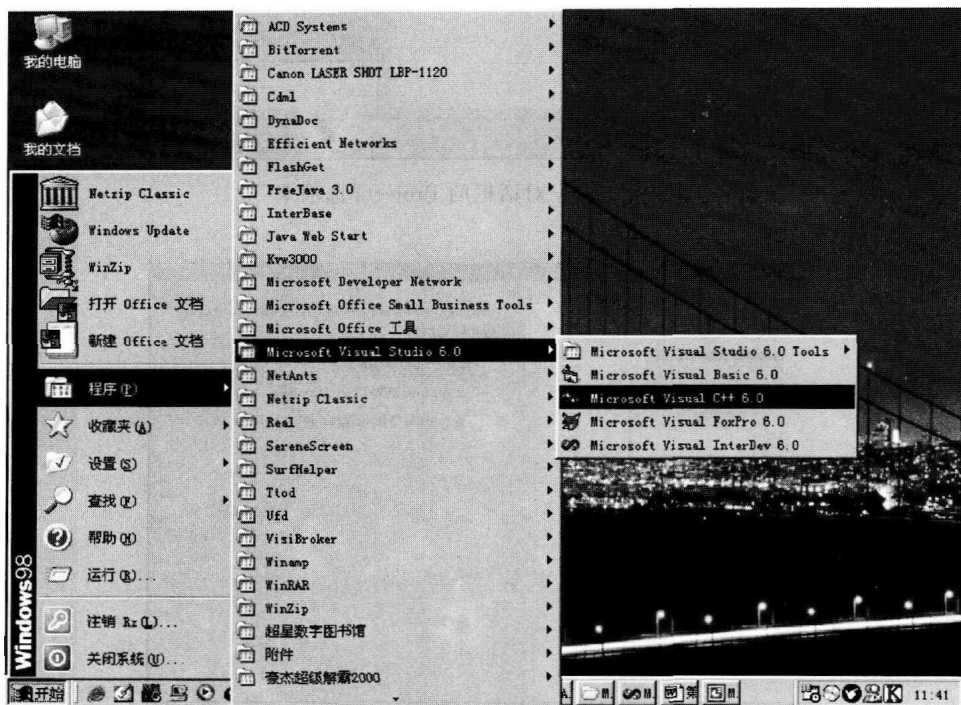


图 1-1 启动 Visual C++ 6.0

- (2) 选择 Visual C++ 6.0 的菜单 File|New 命令打开 New 对话框, 在这个对话框中选择 Projects 选项卡, 见图 1-2。

- (3) 在图 1-2 左侧的列表框中选中 Win32 Application 选项, 在 Project name 编辑框中填写工程名称(例如, MyPrj), 在 Location 编辑框中选择存放工程文件的路径, 其余使用默认选项, 见图 1-2。最后, 单击 OK 按钮, 打开 Win32 Application-Step 1 of 1 对话框, 见图 1-3。

- (4) 在图 1-3 所示的对话框中, 有 3 种工程类型可以供选择:

- An empty project;
- A simple Win32 application;
- A typical "Hello World!" application.

选择 An empty project 选项, 可以创建一个没有任何文件的空工程, 用户需要自己向工程中添加所需要的文件。

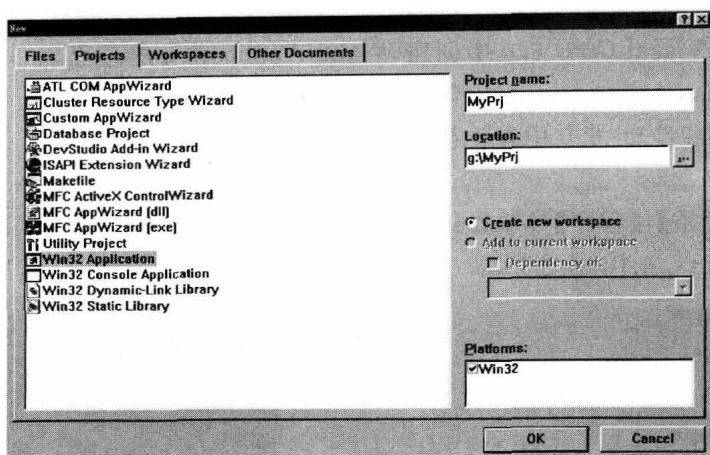


图 1-2 New 对话框的 Projects 选项卡

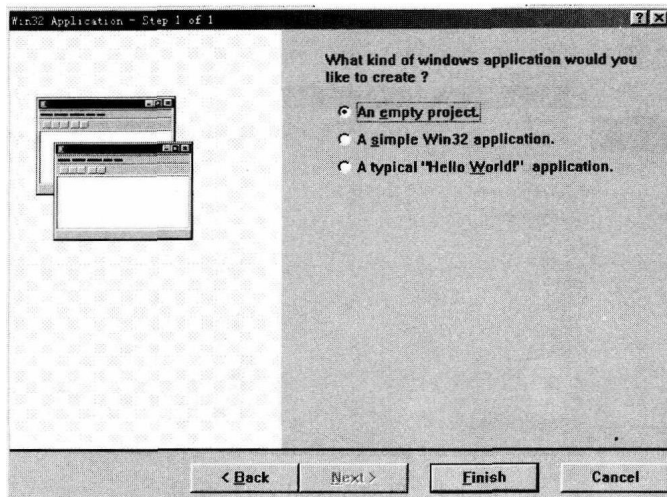


图 1-3 Win32 Application-Step 1 of 1 对话框

选择 A simple Win32 application 选项,可以创建一个由系统自动生成必要代码的工程。

选择 A typical “Hello World!” application 选项,可以创建一个完整的 Win32 示例程序,这个程序在启动运行后会在窗口的用户区显示一个字符串“Hello World!”。

(5) 选择 An empty project 选项,创建一个空 Win32 工程。

(6) 选择菜单 File|New 命令打开 New 对话框,在这个对话框中选中 Files 选项卡,在卡中左侧的窗口中选中 C++ Source File,在确认 Add to project 复选项被选中且在它下面的下拉列表框中的工程名为当前工程名后,在 File 文本框中填写要创建文件的名称(例如,MyPrj)。最后单击 OK 按钮,打开文件,见图 1-4。

(7) 在打开的空文件中输入如下代码。

```
#include<windows.h>
```

```
//编写 Windows 程序必须要包含的头文件
```