21 世纪高职高专规划教材

# Visual C++ 程序设计

王明福　主　编
余苏宁　副主编

高等教育出版社

## 内 容 提 要

本书从实际应用的角度介绍了 Visual C ++6.0软件包的使用方法和编程技巧。通过开发计算器、学生档案管理程序、绘图程序、多媒体点播系统、公众聊天室和桌面时差时钟等程序,详细介绍了包括菜单、对话框、常用控件、工具栏等在内的界面设计,MFC 库的使用和扩展以及对文件、多媒体、数据库、网络通信和多线程等编程技术的具体操作技巧。

本书改变传统写法,采用‘ 项目 ’驱动的编写方式,把知识点融入到实际项目的开发中,通过项目的不断扩展逐步引入新的知识点,通俗易懂,可操作性强。适合高等院校‘ Visual C ++ 程序设计 ’课程教学用书,尤其对高职高专院校计算机专业和从事 Visual C ++ 的编程开发人员,更是一本很难得的好书。

书中所有程序全部运行通过,所有程序源代码及示例相关文档均可以从高等教育出版社网站上下载,网址为 :http ://cs. hep. com. cn 或 http ://www. hep. edu. cn。

# 前　言

Microsoft Visual C++6.0 是微软公司最新推出的开发工具包 Visual Studio 6.0 中功能最强大的开发工具，是一种面向对象程序设计语言，可以大大提高软件设计能力以及开发速度。基于对当前教材的深入了解及教学实践需要，我们着手编写了 21 世纪高职高专规划教材《Visual C++程序设计》。本书从实际应用的角度介绍了 Visual C++6.0软件包的使用方法和编程技巧。通过开发计算器、学生档案管理程序、绘图程序、多媒体点播系统和公众聊天室等程序，详细介绍了包括菜单、对话框、常用控件、工具栏等在内的界面设计方法、MFC 库的使用和扩展以及对文件、多媒体、数据库、网络通信和多线程等编程技术的具体操作使用技巧。

本书改变传统教材的编写方法，具有如下特点：

1. 采用“项目”驱动的编写方式，引入案例教学和启发式教学方法，便于激发学习兴趣。

本书的编写思想是把知识点融入到实际项目的开发中去，立足于“理论够用，操作熟练，重在实践”的要求，力求做到通俗易懂，循序渐进，适合以“案例入门，改造拓宽，项目综合”的学习知识体系模式展开教学。每一章都尽量从实际中的典型实例开始，引出问题，由问题牵引，然后逐步展开，在讲述实例或开发项目的过程中，将本章的知识融入。通过对问题的深化或功能扩充，来拓宽知识的广度和深度，最后达到学习知识、培养能力的目的。这种将知识点融入到实际项目开发中的编写方式，可读性、可操作性强，非常适合高职高专的学生阅读和使用。

2. 在案例或项目的选择上，遵循“易学”、“有趣”和“有用”的原则，这样有利于激发学生的求知欲望。

本教材所选案例（或项目）基本包含了“Visual C++程序设计”的基本概念和设计技巧，由浅入深、循序渐进、逐步拓宽知识点。兼顾了理论知识的系统性和完整性，考虑到了独立和相关的平衡，其总目标是强调综合应用开发能力的培养。换言之，既能实践循序渐进的教学方法，也有利于开展“项目综合”的教学模式，符合教学规律。

3. 一切以实用为目的，注重知识应用的先进性和前沿性，具有高职教育特色。

本教材着眼于 IT 产业飞速发展的需要，将多媒体编程技术、数据库技术和网络通信技术纳入教材内容。本书不追求面面俱到，而是大胆舍去不用或根本就不实用的内容，适合“理论够用，重在实践”的高职高专教学的特点。

4. 本书注重 C 语言程序设计系列的个性和共性，考虑到两个方面的平滑过渡及其中的异同点。一是从面向过程的程序设计到面向对象程序设计的平滑过渡及异同点，二是在 DOS 环境下与 Windows 环境下程序的平滑过渡及异同点。反映在章节内容的安排上，第 1 章介绍了 Visual C++编程基础，其目的就是让读者了解 DOS 程序与 Windows 程序的差别以及 MFC 应用程序的结构；第 2 章介绍 MFC 的界面设计与资源管理，其目的是让读者掌握用 VC 开发平台编写 MFC 应用程序的一些基本操作；从第 3 章开始才是本教材的主要内容。

本书由深圳职业技术学院计算机系王明福、余苏宁两位老师编写。其中第 2 章由余苏宁老

师编写，其余各章由王明福老师编写，最后由王明福负责全书的统稿。此外，李亮、鸟云高娃和王梅等老师在百忙之中对本书的编写思路提出了许多宝贵意见，并承担了部分章节的文字录入与审校工作；同时，本书也得到了计算机系软件专业全体老师的大力支持，提出了许多建设性意见。在此，我们深表感谢。

由于编者水平有限，加之时间仓促，书中难免有错漏之处，敬请广大读者批评指正，在此表示感谢。编者 E – mail 地址是 wmf@ oa. szpt. net。

编　者

2003 年 5 月于深圳

# 目　录

# 第1章

# Visual C++ 编程基础

## 本章导读

Windows 下编程的工具多种多样，但均遵循 Windows 的工作原理。作为一名程序员，必须熟练掌握与编程有关的 Windows 的工作原理。选择能为用户提供一种功能强大同时又易于掌握的应用程序开发工具。

本章通过介绍 Windows 编程的基础知识，编写一句代码的 MFC（Microsoft Foundation Class）应用程序，让读者了解：

- Windows 编程基础 消息驱动、资源管理等
- MFC 类基础
- 利用向导建立一个应用程序框架

---

## 1.1 Windows 编程概念

传统的 MS－DOS 程序主要采用顺序的、关联的、过程驱动的程序设计方法，是面向程序而不是面向用户的。Windows 程序设计是基于事件驱动的，程序的运行不是由事件的顺序来控制，而是由能触发的事件来控制，它是一种面向用户的程序设计方法。其中消息驱动机制是 Windows 程序设计的精髓。

### 1.1.1 事件与消息

Windows 花费大量时间等待用户的动作以便做出响应，所以这种系统也叫事件驱动的系统。例如，当用户按下一个键、移动鼠标或单击鼠标按钮时，计算机通知 Windows 系统已经发生了一个事件以及事件的种类、发生的时间和发生的位置（如坐标值）。

事件以如下三种方式产生：

（1）通过输入设备，如键盘和鼠标，产生硬件事件。

（2）通过屏幕上可视的对象，如菜单、工具栏按钮、滚动条和对话框上的控件。

（3）来自 Windows 内部，例如，让一个后面的窗口显示到前面来。

当 Windows 捕获一条事件后,它会编写一条消息,将相关信息放入一个数据结构中,然后将包含此数据结构的消息发送给需要消息的程序。Windows 消息是在 Windows.h 文件中用宏定义的常数。消息常数名通常为 WM_XXX,例如,WM_WUIT、WM_CHAR 等。

Windows 将消息放入目标应用程序的消息队列中,在消息队列中所有消息都处于等待状态,直到应用程序准备处理它。

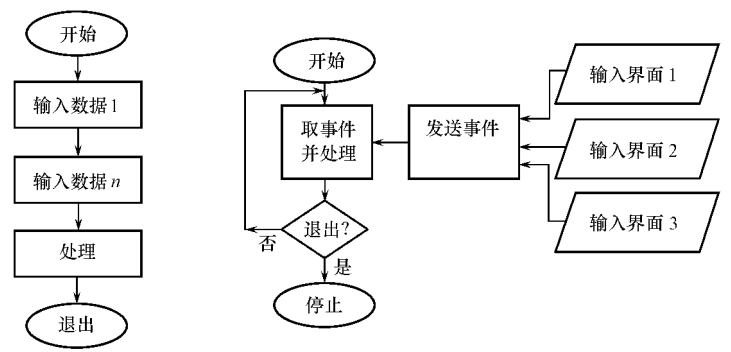
### 1.1.2 消息驱动

消息驱动也称事件驱动,Windows 程序设计是基于事件驱动的,Windows 对消息有一套完善严格的定义,并在其产生时将其发送给所有相关的应用程序,这些消息用于驱动应用程序运行以实现一定的功能。

例如,当用户单击鼠标左键时,将发送 WM_LBUTTONDOWN 消息;而双击时,则发送 WM_LBUTTONDBLCLK 消息。除提供的标准消息外,程序员可以自定义所需的消息。总之,消息驱动是 Windows 程序设计的精髓。

传统的 MS-DOS 程序是一系列预先定义好的操作序列的组合,具有一定的开头、中间过程和结束,也就是程序直接控制程序事件和过程的顺序。它的基本模型如图 1.1(a)所示。

事件驱动的程序设计不是由事件的顺序来控制,而是由事件的发生来控制,而这种事件的发生是随机的、不确定的,并没有预定的顺序,这样就允许程序的用户用各种合理的顺序来安排程序的流程。它是一种面向用户的程序设计方法,它在程序设计过程中除了完成所需功能之外,更多地考虑了用户的各种输入,并根据需要设计相应的处理程序。它是一种被动式的程序设计方法,程序开始运行时,处于等待用户输入事件状态,然后取得事件并做出相应反应,处理完毕又返回并处于等待事件状态,如图 1.1(b)所示。




(a) 过程驱动程序模型　　(b) 事件驱动程序模型

图 1.1 过程驱动与事件驱动模型之比较

### 1.1.3 消息响应函数

消息响应函数是用于处理特定消息的一些代码。收到消息的应用程序会做些什么,取决于应用程序本身。程序员可以编写相应的处理函数以处理消息。例如,当用户点击某菜单项时,希望程序弹出一个口令对话框,那么,只要在相应的消息处理函数中编写弹出一个口令对话框的代

码即可。

对于特定的消息有许多标准或典型的处理。例如,WM _ PAINT 消息(在窗口重新绘制内容时发送)的处理函数需要采取步骤,重新构造显示在窗口的图像,重新绘制可见的文本、图形,等等。

### 1.1.4 资源管理

程序员设计任何应用程序均将涉及到诸如菜单、对话框、消息框以及按钮等标准格式数据。Windows 将这些数据保存在资源文件中,程序员可通过编辑工具编辑、修改这些资源文件,使其提供所需的菜单或按钮,并将其放入设计的程序之中。

DOS 下的内存管理常常是程序设计的瓶颈问题,而在 Windows 下编程时,只需要简单地申请所需内存即可。

### 1.1.5 设备独立性

设备独立性或称设备无关性,是 Windows 95/98/2000 的突出特点之一。在编程时,程序员不必关心诸如打印机、鼠标、键盘、显示器、声卡、显示卡和 CDROM 等多种设备的类型及其驱动程序。Windows 95/98/2000 提供了图形设备接口之类的各种抽象接口,使得在程序中可以通过调用标准函数与硬件交互,这些函数通过设备环境的数据结构,由 Windows 操作系统将其映射到相应的物理设备,而程序员则无需了解其提供的操作设备的各种指令。

## 1.2 MFC 基础

MFC 应用程序框架生成的应用程序使用了标准的结构,很好地解决了因程序结构不同,导致维护程序和程序员之间交流的障碍。

Visual C++ 已集成了 MFC 库、Visual C++ 资源编辑器、AppWizard 和 ClassWizard,明显减少了应用程序编码的时间。AppWizard 为整个应用程序生成框架代码,同时 ClassWizard 为消息处理程序生成原型和函数体。

在本节先介绍 MFC 类库,然后介绍程序框架中的一些要素,接下来介绍 MFC 消息映射,最后介绍程序的运行过程,使读者对 MFC 有一个大概的了解。

### 1.2.1 MFC 类库简介

MFC(Microsoft Foundation Class)类库是用来编写 Windows 应用程序的 C++ 类集,封装了大部分 Windows API 函数,使用 MFC 类库和 Visual C++ 提供的高度可视化的应用程序开发工具,可使应用程序的开发变得简单,而可靠性和可重用性得到很大提高。

MFC 库包括 CObject 类及其派生类以及其他类(可以参考 MSDN 来了解 MFC 类库层次图)。了解了 CObject 类及其派生类,就等于了解了 MFC 的一大半。

在 MFC 类库中,有这样一些重要的类(都是 CObject 的派生类):

(1) 应用程序类 CWinApp 属于 Application Architechture(应用程序体系结构)。一个 MFC

项目对应一个此类的对象。

(2) CWnd 类及派生类 属于 Windows Support 部分,用户看到的 Windows 界面都是由这个类的对象所形成。它包括这样几部分:

① Frame Windows 包括用于生成框架窗口的 CFrameWnd 及其派生类,用于生成分隔栏窗口的 CSplitterWnd。

② Views 包括 CView 及其派生类,用于生成视图窗口。

③ Dialog 包括 CDialog 及其派生类,用于生成对话框。

④ Control Bars 包括 CControlBar 及其派生类,用于生成状态栏和工具栏等。

⑤ Property Sheets 包括 CPropertySheet 及其派生类,用于生成属性表。

⑥ Controls 包括各种控件类,比如,CEdit 用于生成编辑框,CListBox 用于生成列表框等。

(3) CDocument 及其派生类 和 CWinApp 同属于 Application Architechture 范畴。用于提供应用程序数据的存储和加载,常和 CView 类一起工作,合在一起称为文档/视图结构。

(4) File Services 包括 CFile 各类,提供文件服务,这是一种底层的服务,一般可以用一些高层的服务代替(比如文档的序列化等)。

(5) Graphical Drawing 包括 CDC(Class of Device Context,设备环境类)等类,提供图形绘制功能。当希望在视图窗口中绘图时,需要使用该 CDC,以提供绘图环境。

(6) Graphical Drawing Objects 绘图环境,可以提供绘图对象,比如画笔(CPen)、刷子(CBrush)等来进行绘制。

(7) Menus 包括 CMenu 类,封装了 Windows 中菜单的数据结构。

(8) ODBC Database 和 DAO Database Support 包括 CDatabase 和 CDaoDatabase 等类,提供数据库服务。

(9) Internet Services 包括 CInternetSession 等各类,提供网络服务。

此外,还有一些非 CObject 类及派生类,也是必须了解的。比如:

(10) Simple Value Types 包括 CPoint、CRect、CSize、CString、CTime 和 CTimeSpan。

(11) Internet Service API 包括 CHttpServer 等各类,用于提供底层的网络服务,对这些类的理解最好能结合实例进行。

### 1.2.2 MFC 应用程序框架

应用程序框架包含用于生成应用程序所必需的各种面向对象的组件的集合。在 Visual C++6.0 中,MFC AppWizard 能方便地生成应用程序框架,然后可以在此基础上进行进一步的编辑工作。MFC AppWizard 生成的应用程序包括如下一些要素:

(1) WinMain() 函数 Windows 要求应用程序必须有一个 WinMain() 函数。但在程序中看不到 WinMain(),因为它隐藏在应用程序框架中。

(2) 应用程序类 也称 CMyHelloApp。该类的每一个对象代表一个应用程序。程序中默认定义一个全局 CMyHelloApp 对象,即 theApp。CWinApp 基类决定 theApp 的大多数行为。

(3) 应用程序启动 启动应用程序时,Windows 调用应用程序框架内置的 WinMain() 函数,WinMain() 寻找由 CWinApp 派生出的全局构造的应用程序对象。在 C++ 程序中,全局对象在主程序执行之前构造。

(4) 成员函数 CMyHelloApp::InitInstance() 当 WinMain() 函数找到应用程序对象时,它调用伪成员函数 InitInstance(),这个成员函数调用所需的构造并显示应用程序的主框架窗口。必须在派生的应用程序类中重载 InitInstance(),因为 CWinApp 基类不知道需要什么样的主框架窗口。

(5) 成员函数 CWinApp::Run()。函数 Run() 隐藏在基类中,但是它发送应用程序的消息到窗口,以保持应用程序的正常运行。在 WinMain() 调用 InitInstance() 之后,便调用 Run()。

(6) CMainFrame 类 类 CMainFrame 的对象代表应用程序的主框架窗口。当构造函数调用基类 CFrameWnd 的成员函数 Create() 时,Windows 创建实际窗口结构,应用程序框架把它连接到 C++ 对象,函数 ShowWindows() 和 UpdateWindow() 也是基类的成员函数,必须调用它们来显示窗口。

(7) 文档与视图类 这一部分比较复杂,会在后面的章节中单独提出详细讲解。

(8) 关闭应用程序 如果用户通过关闭主框架窗口类关闭应用程序,这个操作就将激发一系列事件的发生,包括 CMainFrame 对象的析构、从 Run() 中退出、从 WinMain() 中退出和 CMyHelloApp 对象的析构。

### 1.2.3 MFC 消息映射及处理

在 MFC 中,管理消息的方式通常是这样的:当发生某一个消息(比如用户移动了鼠标和按下键盘等),该消息进入消息队列,操作系统根据消息提供的信息决定由哪个应用程序来处理,该应用程序依照一定的方式查找应用程序中各个类的消息映射(一组宏,这些宏用来确定某个消息及相应的处理程序的对应关系),找到处理程序,然后由处理程序执行。

MFC 程序要处理消息的种类如下:

(1) Windows 消息 这类消息以 WM_开头,但 WM_COMMAND 除外。通常由窗口和视图来处理。这些消息常常带有参数,用于决定处理该消息的方式。

(2) 由控件和其他子窗口发送给父窗口的 WM_COMMAND 消息 这些消息中包括 EN_CHANGE 通知码。例如,当用户在编辑框中键入文本或进行修改时,就会向系统发送一个带 EN_CHANGE 通知码的 WM_COMMAND 消息。

(3) 来自于用户界面对象的 WM_COMMAND 消息 如表 1.1 所示。这些用户界面对象包括菜单、工具栏按钮和快捷键。系统处理这些用户界面对象消息的方式和处理前面的消息有所不同,当这种类型对象接受某个消息时,它将处理该消息的权利优先提供给其他对象。

表 1.1 用户界面对象发送消息处理顺序表

| 对 象 | 处理次序 |
| --- | --- |
| MDI 框架窗口<br>(CMDIFrameWnd) | 1. 活动的 CMDIChildWnd 2. 本框架窗口<br>3. 应用程序(CWinApp 对象) |
| 文档窗口<br>(CFrameWnd,CMDIChildWnd) | 1. 活动视图 2. 本框架窗口<br>3. 应用程序(CWinApp 对象) |
| 视图 | 1. 本视图 2. 和视图相关的文档 |
| 文档 | 1. 本文档2. 和该文档相关的文档模板 |

| | |
|---|---|
| 对话框 | 1. 本对话框2. 负责该对话框的窗口<br>3. 应用程序(CWinApp 对象) |

### 1.2.4　程序的运行过程

如图 1.2 所示，当运行用户应用程序时，程序中的框架首先获得控制权，然后依次执行下述功能：

（1）做部分初始化工作。

（2）构造应用程序的惟一应用类的对象，构造应用类对象时要调用其构造函数。

（3）调用 WinMain()函数（此函数也隐藏在应用框架内部）。注意，调用函数 WinMain()时已将其 hPrevInstance 参数强行置为 NULL，这一点与 SDK 有区别。

（4）从 WinMain()函数返回后，删除应用程序的惟一应用类的对象，删除时要调用其析构函数。

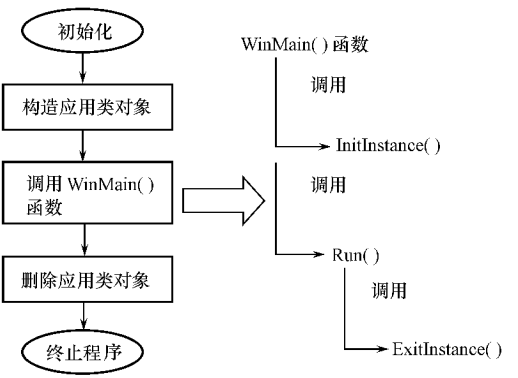



图 1.2　应用程序的运行过程

其中调用 WinMain()函数又进一步细分为：

① 获得指向惟一应用类对象的指针。

② 进行一些内部初始化，若失败，则转至第(6)步。

③ 进行应用程序初始化，也就是调用应用类的 InitApplication()函数进行初始化。如果失败则转至第(6)步，若成功则继续下一步。

④ 进一步初始化应用程序，即调用应用类 InitInstance()函数进行初始化。由于 MFC AppWizard 自动生成的应用程序框架中的应用类重载了 MFC 中应用类基类的 InitInstance()函数，所以这时调用的就是应用程序框架中应用类对象的 InitInstance()函数。如果初始化失败，则调用应用类的 ExitInstance()函数，然后转至第(6)步，若初始化成功则继续下一步。

⑤ 调用应用类的 Run()函数，其主要功能就是运行消息循环，不断获取消息和处理消息（翻译和分派），若有用户行为如鼠标移动等消息，则处理这些消息；若没有，则进行空闲消息处理。一直到用户关闭应用程序窗口，产生 WM _ QUIT 消息时，Run()函数才调用类的 ExitInstance()函数，准备退出。

（5）终止应用程序。

（6）进行退出应用程序前的收尾工作，如删除注册的窗口类并释放其内存等。

（7）返回。

## 1.3　第一个 MFC 应用程序

用 Visual C++6.0 的 MFC 编写 Windows 应用程序，是一种“填空式”的编程方法，一般有 3 个步骤：

（1）创建工程　用 Visual C++6.0 的 MFC AppWizard 生成应用程序的工程文件，也就是创建应用程序的基本框架。

(2) 可视化设计 用 Visual C++ 自带的工具软件 Winzards,制作 Windows 风格的图形用户界面和各种控件,如矩形按钮、滚动条和单选按钮等。

(3) 编写程序代码 根据目标程序的要求,用 MFC ClassWizard 添加消息响应函数,然后用 Visual C++ 提供的文本编辑器和 C++ 程序设计语言在函数中编写代码。

### 1.3.1 MyHello 应用程序

在开始编写 MyHello.exe 程序之前,首先观察一下它的外观和所具有的功能:

(1) 例程 MyHello.exe 运行结果如图 1.3 所示。

(2) 程序主窗口显示字符串"Hello,我们开始 V C++ 编程了!"。

图 1.3 MyHello 程序运行结果

(3) 单击 MyHello.exe 程序主窗口右上角关闭按钮,退出 MyHello.exe 程序。

对 MyHello 程序进行目标分解,用 Visual C++6.0 开发,由以下两步完成:

① 用 Visual C++6.0 的 MFC AppWizard,创建应用程序的基本窗口框架。

② 编写显示字符串"Hello,我们开始 V C++ 编程了!"的代码。

### 1.3.2 创建工程

在生成 MyHello.exe 程序的工程文件和其他文件之前,首先应在硬盘上创建一个新目录,比如 d:\MYVC,以便存放程序所生存的文件。

创建 MyHello 工程的步骤如下:

(1) 启动 Visual C++6.0。从"File"菜单中选择"New"。此时,Visual C++ 将显示一个"New"对话框。

(2) 在"New"对话框中选择"Project"标签,然后选择"MFC AppWizard(exe)"类型,告诉 Visual C++ 即将创建一个 EXE 程序。

(3) 在"New"对话框的"Project name"文本编辑框中输入"MyHello",单击位于"Location"框右边的小按钮,再从下拉的对话框中选择"d:\MYVC"目录,使新创建的工程文件放置在"d:\MYVC"目录之下。

以上几个步骤分别指定了 MyHello.exe 程序的工程类型、工程名字和工程位置,此时"New"对话框如图 1.4 所示。


图 1.4 指定工程类型、工程名字和工程位置

(4)单击‘OK’按钮。此时 Visual C++将显示如图 1.5 所示的‘MFC AppWizard – Step 1’对话框。

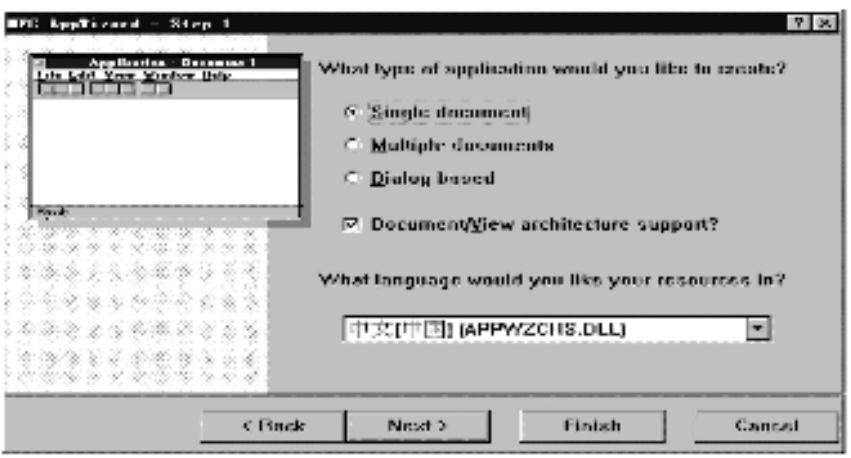



图 1.5 MFC AppWizard – Step 1 设置应用程序类型

(5)在‘MFC AppWizard – Step 1’对话框中:

①“Single document”选项表示单文档界面,简称 SDI,这种类型应用程序的主窗口只能容纳一个文档,如 Windows 自带的记事本。

②“Multiple documents”选项表示多文档界面,简称 MDI,这种类型应用程序容许同时打开多个文档,这些文档可以层叠于主窗口。Microsoft Office 产品就属于 MDI 应用程序。

③“Dialog based”选项表示生成基于对话框的应用程序。

在本例中选择‘Single document’,创建一个基于单文档界面的应用程序。然后选择资源语言,本例中选择‘简体中文’。

(6)单击‘Next’按钮,Visual C++6.0 显示‘MFC AppWizard – Step 2 of 6’对话框,如图 1.6