



高等学校计算机教材

Visual C++ 6.0

网络编程教程

◎ 郑阿奇 主编



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

<http://www.phei.com.cn>

高等学校计算机教材

Visual C++ 6.0 网络编程教程

郑阿奇 主编

电子工业出版社

Publishing House of Electronics Industry

内 容 简 介

本书以 Visual C++ 6.0 为平台, 通过网络典型应用软件开发, 介绍网络编程。包括 MFC Socket 编程、Winsock API 编程、即时通信应用开发、Web 浏览器与服务器软件开发、文件传输软件开发和电子邮件软件开发。同时, 介绍相关网络知识、网络环境配置、搭建步骤配合, 便于理解和运行书中实例。程序之间实现互操作, 如客户端和服务器对接、接入第三方程序、程序之间整合集成成为套件等, 使书中每个例子不再是孤立的实体。将书中的实例程序与当前流行的产品软件进行比较, 从而激发读者从事实际应用性网络编程开发的热情。

本书可作为计算机及相关专业本、专科网络编程的教材或参考书。同时, 为广大计算机网络 DIY 玩家成为网络编程高手打好基础。

未经许可, 不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有, 侵权必究。

图书在版编目 (CIP) 数据

Visual C++ 6.0 网络编程教程 / 郑阿奇主编. —北京: 电子工业出版社, 2013.6
高等学校计算机教材
ISBN 978-7-121-20531-6

I. ①V… II. ①郑… III. ①C 语言—程序设计—高等学校—教材 IV. ①TP312

中国版本图书馆 CIP 数据核字 (2013) 第 111697 号

责任编辑: 张 慧

印 刷: 三河市鑫金马印装有限公司

装 订: 三河市鑫金马印装有限公司

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开 本: 787×1092 1/16 印张: 20.5 字数: 525 千字

印 次: 2013 年 6 月第 1 次印刷

印 数: 3 000 册 定价: 39.00 元

凡所购买电子工业出版社图书有缺损问题, 请向购买书店调换。若书店售缺, 请与本社发行部联系, 联系及邮购电话: (010) 88254888。

质量投诉请发邮件至 zltts@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线: (010) 88258888。

前 言

我们生活在信息时代，计算机和网络是这个时代的产物和标志。打开计算机上网，用户接触到各种各样的网络应用软件，如即时通信、文件下载、Web 应用、电子邮件……网络应用软件（又称为网络编程）自然成为焦点。

Visual C++ 6.0 仍然是目前最流行的软件开发和教学平台。本书以 Visual C++ 6.0 作为平台，介绍 Visual C++ 网络编程，具有如下特色。

（1）通过典型网络应用实例，由表及里、深入浅出地介绍网络编程。

（2）在介绍编程的同时适当地介绍相关网络知识并详细给出网络环境配置、搭建步骤，使读者能够很方便地开发本书实例。

（3）在本书程序之间实现互操作，如客户端和服务端对接、接入第三方程序、程序之间整合集成成为套件等，使书中每个例子不再是孤立的实体，而是整个互联网世界的一分子。

（4）联系实际，将书中开发实例与当下流行的网络软件产品进行比较，指出它们在基本原理上的相通之处，以及书中原型程序的局限性和完善改进的方向，从而激发读者从事实际应用性网络编程开发的热情。

本书从崭新的视角透视网络程序，用形象生动的语言介绍网络编程，软件与自然事物相类比，程序与网络环境相依存，编程技巧与探索实验并重，技术原理与哲理感悟交织，读者只要学过 C++ 语言，懂一点计算机和网络的基本知识，就可以顺利地学习本书。普通高校计算机及相关专业本专科学生可以将本书作为学习网络编程的教材或参考书。同时，我们期待本书能够成为广大计算机爱好者、网络爱好者、编程爱好者、软件发烧友和计算机网络 DIY 玩家的好朋友，为他们中的更多人成为高手打好基础。

本书提供同步教学课件和网络应用实例的所有源代码文件，网址为 www.hxedu.com.cn。

本书由南京师范大学郑阿奇主编，参加本书编写的还有梁敬东、顾韵华、王洪元、刘启芬、丁有和、曹弋、徐文胜、殷红先、张为民、姜乃松、彭作民、高茜、陈冬霞、钱晓军、朱毅华、时跃华、周何骏、赵青松、周淑琴、陈金辉、李含光、王一莉、徐斌、王志瑞、孙德荣、周怡明、刘博宇、郑进、刘毅、刘有春等。

由于作者水平有限，不当之处在所难免，恳请读者批评指正。

作者 E-mail: easybooks@163.com。

编 者
2013 年 3 月

目 录

第 1 章 开发平台和网络编程基础知识	1
1.1 Visual C++6.0 开发平台	1
1.1.1 创建 Visual C++6.0 项目工程	1
1.1.2 Visual C++6.0 可视化设计	3
1.1.3 一个简单的 Visual C++6.0 小程序	6
1.2 网络编程基础知识	13
1.2.1 网络协议	13
1.2.2 网络应用编程界面	16
1.2.3 网络程序工作机理	18
1.2.4 本书编程的协议环境	20
第 2 章 MFC Socket 编程	23
2.1 MFC 及其 Socket 类	23
2.1.1 MFC 简介	23
2.1.2 MFC 中的 Socket 类	24
2.2 Socket 类网络编程	27
2.2.1 客户端—服务器方式 (C/S 模式)	27
2.2.2 CAsyncSocket 类编程基础	29
2.2.3 CAsyncSocket 类程序的指针实现	46
2.2.4 CSocket 类编程	55
2.3 Socket 程序的互通	64
2.3.1 不同版本 Socket 程序的互通	64
2.3.2 接入第三方 Socket 程序	67
第 3 章 Winsock API 编程	69
3.1 Winsock API 编程原理	69
3.1.1 通行的操作	69
3.1.2 Winsock API 函数详解	71
3.1.3 TCP 与 UDP	75
3.2 TCP 编程	75
3.2.1 TCP 通信流程	76
3.2.2 TCP Socket API 程序设计	76
3.2.3 Winsock API 程序与 MFC Socket 程序的等价性	86
3.3 UDP 编程	86
3.3.1 UDP 通信流程	87
3.3.2 UDP Socket API 程序设计	87

3.3.3	UDP 进程通信演示	95
第 4 章	即时通信应用开发	99
4.1	IM 软件的体系结构	99
4.1.1	互联网中继通信原理	99
4.1.2	P2P 方式架构的系统	100
4.2	C/S 结构的聊天室应用	101
4.2.1	聊天室功能效果展示	101
4.2.2	聊天室的开发	102
4.3	P2P 架构的简单聊天工具	121
4.3.1	软件使用效果展示	121
4.3.2	P2P 通信规约	122
4.3.3	聊天工具的开发过程	124
4.3.4	P2P 方式通信的特性	142
4.4	原型程序与 IM 产品	146
4.4.1	本程序与腾讯 QQ 软件的类比	146
4.4.2	IM 产品的增强功能与技术	148
4.4.3	即时通信发展新趋势	148
第 5 章	HTTP 编程与万维网开发	150
5.1	HTTP 原理	150
5.1.1	万维网的工作过程	150
5.1.2	超文本传输协议	151
5.1.3	统一资源定位符 URL	155
5.2	浏览器开发	156
5.2.1	MFC 对浏览器开发的支持	156
5.2.2	定制开发自己的浏览器	160
5.3	Web 服务器的开发	174
5.3.1	项目框架的建立	174
5.3.2	Web 服务器界面总控	179
5.3.3	Web 服务流程的实现	181
5.3.4	HTTP 协议的实现	186
5.3.5	协议实现的辅助代码	192
5.4	自制浏览器访问 Web 服务器	196
5.4.1	Web 资源准备	196
5.4.2	访问 Web 服务器	198
5.4.3	相对路径下的资源访问	200
第 6 章	FTP 编程与资源访问	202
6.1	FTP 应用基础	202
6.1.1	FTP 简介	202

6.1.2	FTP 的特性	203
6.1.3	FTP 工作原理	204
6.1.4	FTP 命令和应答	206
6.1.5	FTP 网络环境搭建和使用	208
6.2	制作 FTP 上传下载器	213
6.2.1	WinInet 类对 FTP 的支持	213
6.2.2	设计软件界面	214
6.2.3	编程实现	215
6.2.4	测试 FTP 客户端	222
6.3	FTP 服务器的实现	223
6.3.1	项目框架的建立	224
6.3.2	FTP 服务器界面总控	228
6.3.3	FTP 服务流程的实现	234
6.3.4	FTP 协议的实现	240
6.3.5	FTP 实现辅助代码	249
6.4	自制 FTP 客户端与服务器对接	264
6.4.1	FTP 上传下载器的改造	264
6.4.2	自制客户端访问服务器	266
第 7 章	电子邮件应用编程	269
7.1	邮件系统原理	269
7.1.1	概述	269
7.1.2	邮件客户端配置	271
7.1.3	邮件收发环境	276
7.2	基于 MAPI 的邮件客户端开发	280
7.2.1	开发邮件程序的接口 MAPI	280
7.2.2	邮件客户端程序开发	281
7.2.3	网络邮件收发实验	293
7.3	基于 POP3 的邮件接收程序	298
7.3.1	POP3 原理	298
7.3.2	用 POP3 协议实现邮件接收	303
7.3.3	用 POP3 邮件程序接收信件	318

开发平台和网络编程基础知识

1.1 Visual C++ 6.0 开发平台

Microsoft Visual C++（简称 Visual C++、MSVC、VC++或 VC），是微软公司的 C++ 开发工具，具有集成开发环境，可提供编辑编译 C、C++等语言。VC++整合了便利的排错工具，特别是整合了视窗应用编程接口（Windows API）、三维动画 DirectX API 等。初学 VC 入门的最佳环境是 Visual C++ 6.0。

下面以编写一个简单的演示程序为例，使读者初步熟悉 Visual C++ 6.0 环境下 VC 编程开发的基本操作。

1.1.1 创建 Visual C++ 6.0 项目工程

选择菜单命令“文件”→“新建”，弹出如图 1.1 所示的“新建”对话框。

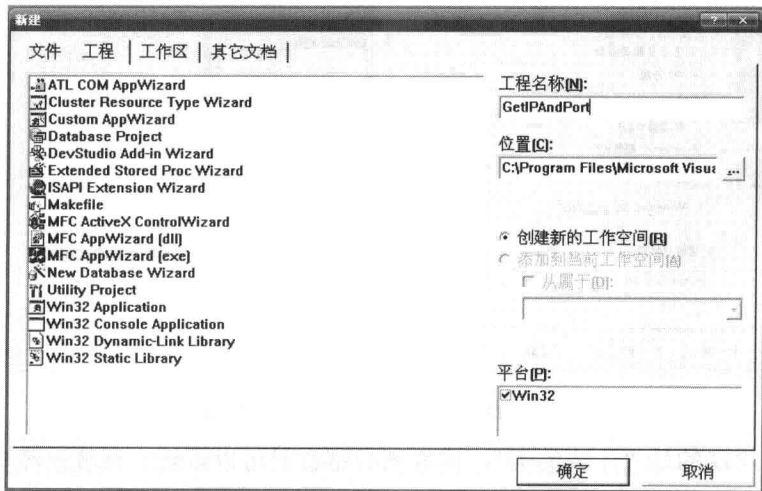


图 1.1 “新建”对话框

在“工程”选项卡左边的列表中选中“MFC AppWizard[exe]”，在右边“工程名称”栏填写项目名为“GetIPAndPort”（我们即将做的这个软件是用来演示程序如何获得用户输入的 IP 和端口号的——这也是几乎所有网络程序都具有的功能，所以取这个名字）。

单击“确定”按钮，弹出“MFC 应用程序向导”对话框（如图 1.2 所示），接下来将在这个对话框的指引下轻松完成创建 VC 工程的工作。

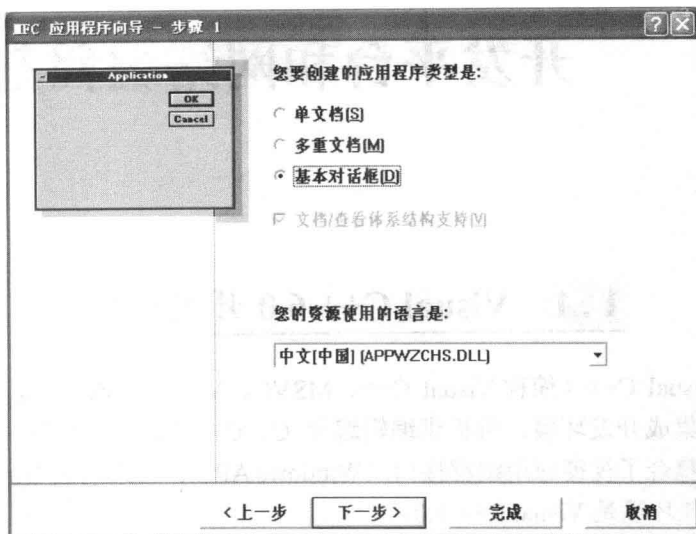


图 1.2 “MFC 应用程序向导”对话框

选中“基本对话框”单选按钮（这个程序很简单，不用文档），单击“下一步”按钮。

接下来的“步骤 2”和“步骤 3”界面（如图 1.3 所示）都采用系统默认设置，连续单击“下一步”跳过。

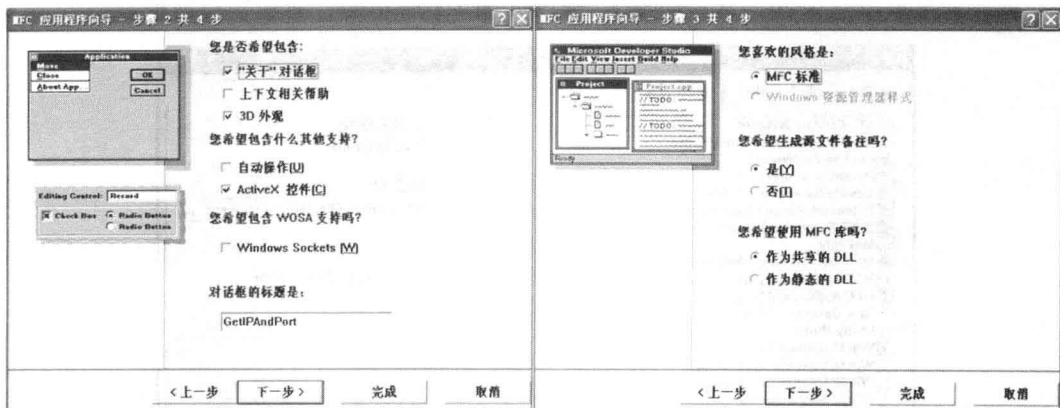


图 1.3 “步骤 2”和“步骤 3”默认设置

最后一步出现的是“生成的类”，读者稍经留意就可以看到，系统已经自动为程序建立了两个类——CGetIPAndPortApp 和 CGetIPAndPortDlg（如图 1.4 所示）。其中，CGetIPAndPortApp 类代表应用程序本身，CGetIPAndPortDlg 代表程序的主界面对话框。细心的读者可能会发现，这两个看似冗长的类名中间部分“GetIPAndPort”就是我们刚才

取的工程名！没错，以后大家就会发现一个规律：VC 在一开始创建工程时都会默认生成两个类，名称形如 CXXXXApp 和 CXXXXDlg（其中“XXX”部分就是用户指定的工程名），这样的命名法则是为了方便用户理清程序的类结构。



图 1.4 生成的类

单击“完成”按钮，出现如图 1.5 所示的信息窗口，其中显示用户要创建的这个工程的信息。



图 1.5 “新建工程信息”窗口

单击“确定”按钮，一个 VC 工程就创建完成了。

1.1.2 Visual C++ 6.0 可视化设计

开发环境工作区主界面将呈现的模样如图 1.6 所示。

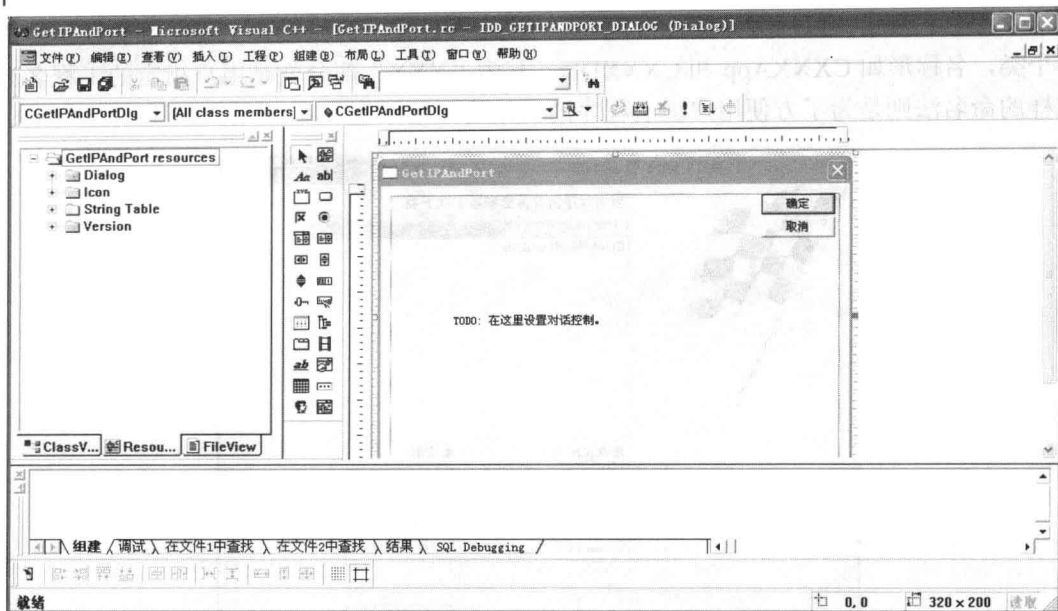


图 1.6 开发环境工作区主界面

主工作区大致分为三部分，最左边是供用户浏览程序结构的，包括多个选项卡界面，常用的是 ClassView（类视图）、ResourceView（资源视图）和 FileView（文件视图），如图 1.7 所示。

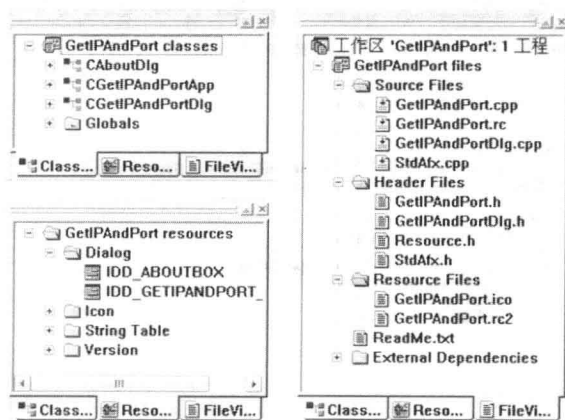


图 1.7 三个常用的视图

类视图用树状结构展示了工程中所有 C++ 类及其层次结构，可通过“查看”→“建立类向导”，在“MFC ClassWizard”窗口中设置对应的类，包括为其添加新的事件消息，重写某些方法的实现代码等。

资源视图分类列出了程序的所有资源，其中常用的是 Dialog 资源，这种资源是每一个具有 GUI 的程序都有的，双击资源 ID 号可以打开对应的界面设计工作区，就可以设计程序的图形界面了。

文件视图以树形目录结构列出了程序中包含的所有代码文件，VC 将它们分成三大类：

头文件(.h)、源文件(.cpp)和资源文件。头文件主要是对程序中用到的各种变量、常量、函数和类进行定义和声明；源文件是程序的主体部分，是各个函数、类的具体实现代码；资源文件定义程序运行要用到的各种资源，如图片、动画、声音和视频等，这对于一些功能丰富的软件（如多媒体类应用）是必不可少的。

VC 在创建工程时默认会生成两个类（本例是 CGetIPAndPortApp 和 CGetIPAndPortDlg），图 1.7 文件视图中的头文件 GetIPAndPort.h 和源文件 GetIPAndPort.cpp 对应 CGetIPAndPortApp 类，它们共同构成了该类的源代码实现；同理，GetIPAndPortDlg.h 和 GetIPAndPortDlg.cpp 对应 CGetIPAndPortDlg 类，从类视图图中也可以看到这两个类。

除此之外，读者会发现类视图里多了一个 CAboutDlg 类，它是 VC 自动创建的，称为“关于 GetIPAndPort”对话框，用于显示程序的版本信息。

在本例的三个类中，CAboutDlg 类和 CGetIPAndPortDlg 类都有各自的对话框界面资源。资源视图中 Dialog 目录下有它们的 ID 号（对应分别是 IDD_ABOUTBOX 和 IDD_GETIPANDPORT_DIALOG），双击 ID 号可以打开其对话框的界面设计工作区，如图 1.8 所示。

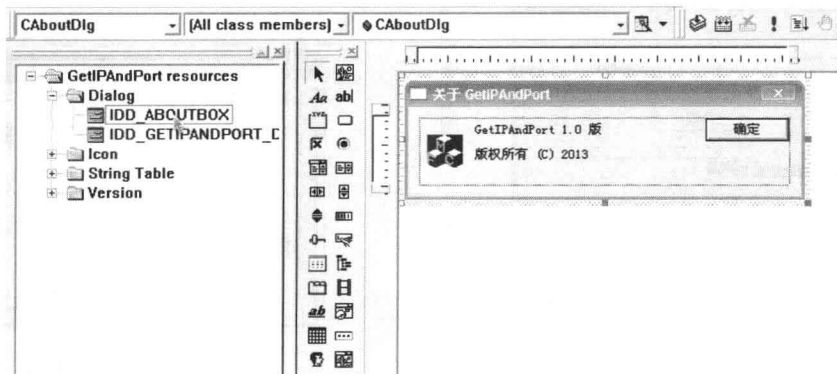


图 1.8 “关于 GetIPAndPort”对话框的设计工作区

工作区中显示的是“关于 GetIPAndPort”对话框的默认界面，读者可以在此基础上自己设计或重新布局。

类视图、资源视图和文件视图三者密切配合，将程序代码有机地组织成为一个结构精巧的 C++ 项目。通过这三者可以清楚地看到程序中每个类的对应代码实体，利用 C++ 写的类不再是抽象的代码，而是成为看得见摸得着的现实存在，这就是可视化设计的魅力！

对话框界面的设计布局如 Visual Basic 一样方便：开发环境界面中央就是主工作区，在这里可以打开任意多个程序文件（源文件或头文件）以及对话框界面设计工作区；只要将左边工具箱中的控件直接拖曳到工作区中就可以设计出自己想要的程序界面。VC 环境下的工具箱和 VB 的一样，都包含了一般 Windows 程序通用的界面元素（按钮、文本框、静态标签、列表框、滚动条等），初学编程者一般都喜欢选择 Visual Basic，用过 VB 的人对这些常用控件肯定再熟悉不过了，因此这里不作过多介绍。只是针对本书所介绍的网络编程，有一个控件需要特别提一下，那就是 **IP 地址控件**（如图 1.9 所示）。它在界面上的显示效果如图 1.10 所示。

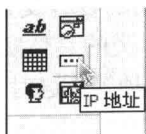


图 1.9 IP 地址控件

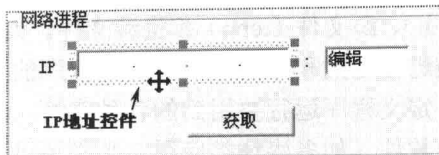


图 1.10 地址控件的显示效果

稍后将会介绍这个控件的具体用法，读者会看到它是个很实用的控件。

选择工具箱中的控件布局程序界面，可以看到 VC 界面设计环境的使用极其方便，丝毫不比 VB 逊色。在布局界面的时候可以使用工具栏中提供的功能调整各个控件的大小、对齐方式，如图 1.11 所示。首先选择“关于”按钮控件，再选择“退出”按钮控件，然后单击开发环境左下角工具栏中的“使大小相同”按钮就可以使先选择的按钮与后选择的按钮大小一样。

最终设计出的程序界面效果如图 1.12 所示。

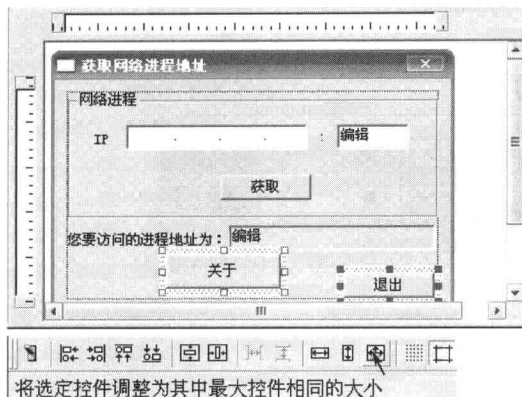


图 1.11 调整控件的大小

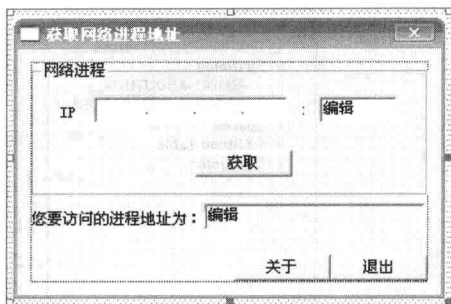


图 1.12 程序界面

在网络编程中，程序使用 IP 和端口来标志网络上的其他程序，以实现程序之间的通信（进程通信）。因此，获取并正确处理对方进程的 IP 和端口（通常由用户通过界面输入指定）就成为网络程序的通用功能。下面的小程序用来演示这个最基本的功能，不过，它只负责获取和处理 IP、端口号，并不实际发起网络连接。

1.1.3 一个简单的 Visual C++ 6.0 小程序

这个程序的界面已经设计好了，但要使程序完成一定的功能，还必须为其编写代码。写代码之前，首先要定义程序中需要用到的一些变量。在 VC 中，很多变量都不是孤立的，而是与某个界面元素（即控件）绑定，这样用户在界面上的输入就可以很容易地传递给程序中相应的变量进行了。例如，为了在程序代码中获得用户输入的 IP，需要为 IP 地址控件关联一个变量。选择“查看”→“建立类向导”命令，打开如图 1.13 所示的“MFC ClassWizard”对话框。

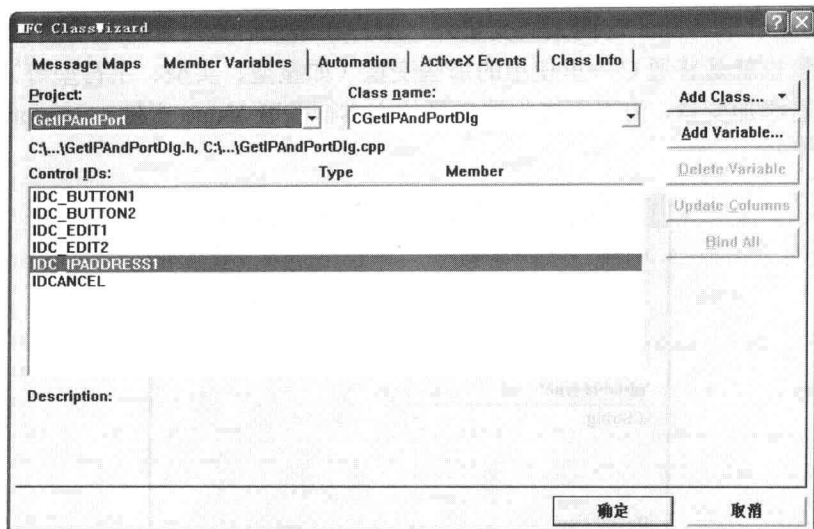


图 1.13 打开 MFC 类向导对话框

切换到“Member Variables”选项卡，在“Control IDs”列表中选中“IDC_IPADDRESS1”（IP 地址控件的 ID 号），单击“Add Variable”按钮，出现“Add Member Variable”对话框（如图 1.14 所示），变量命名为 m_ip，变量类别设置为“Control”。

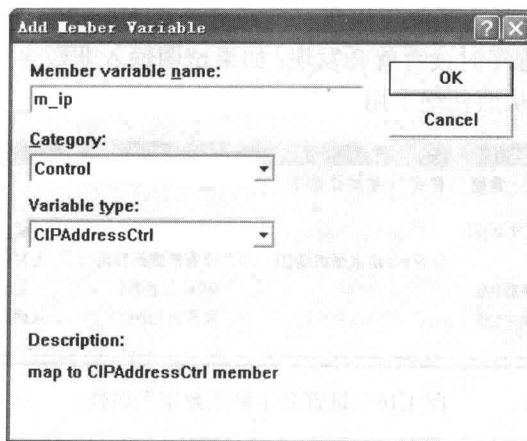


图 1.14 添加 Control 型变量

“Control”表示控件变量，它是微软对实现 Windows 程序的图形用户界面元素的一些类（如 CEdit、CButton）的总称，这些类大多是由同一个叫作窗口（CWnd）的 C++ 类衍生过来的。CWnd 及其庞大的衍生类家族封装了 Windows 的 GUI，作为 MFC 的重要组成部分，是微软公司对基础 C++ 语言的扩充，使其具有了强大的图形化界面功能。与 CWnd 家族中的类关联的变量即**控件变量**，可以通过这样的变量获取和控制它所关联控件的几乎一切的行为，从而对程序界面进行灵活的编程和定制。

而那些只是获取用户输入值的变量称为**值类型**（Value）变量，这种变量与我们以前学习 C++ 语言课程时接触的那些变量差不多，所不同的只是它们都有各自关联的控件，用于获取和保存特定控件接收的用户输入。

除了 Control 和 Value 这两大类变量外，用户在编程时也可以根据需要自定义临时变量，这种变量也就是普通 C++ 里使用的那些变量（如整型、实型、字符型等）。

下面接着添加变量，为用于接收端口号的文本框关联 Value 变量 m_strport，如图 1.15 所示。

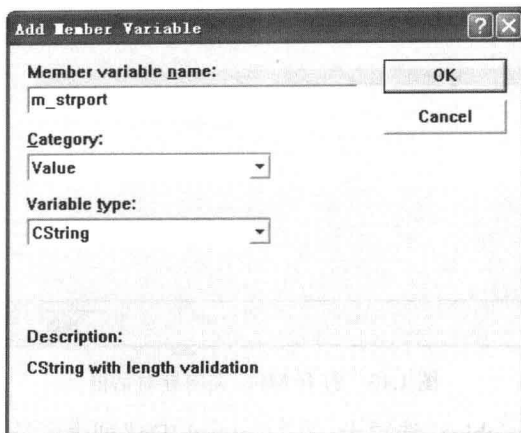


图 1.15 添加 Value 型变量

右击该文本框，在如图 1.16 所示的“Edit 属性”对话框的“样式”选项卡中勾选“数字”，之所以这样设置，是为了限定用户只能在这个文本框中输入数字形式的端口号，这样设置之后，后面运行程序时读者就会发现：如果试图输入非数字字符（中文、英文字母），文本框一概不会响应，从而杜绝了用户的非法输入。

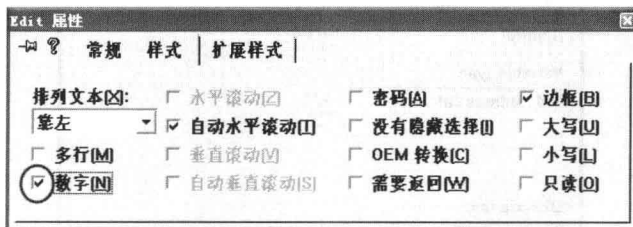


图 1.16 设置文本框“数字”属性

本程序还有一个文本框是用于显示程序获取的 IP 和端口的，为它关联 Control 型变量 m_showIpAndPort，并且设置“只读”属性为 True（作为显示信息窗口的文本框一般都设为只读模式），如图 1.17 所示。

至于将控件关联的变量究竟是设成为 Control 还是 Value 型，则应视具体需要而定。一般来说，如果希望对控件的行为进行某种控制，设置变量为 Control 型较好；如果仅想获得控件接收的用户输入值，设置为 Value 型也就足够了；有时候对控件的行为有较高的控制要求同时又希望能够非常方便地获取到控件的输入值，也可以对一个控件同时关联定义这两类变量。

现在，所有的变量都关联好了，界面控件也都进行了恰当的设置和布局，就剩下编码工作了。程序界面上的“获取”按钮是实现本程序功能的关键，编程工作主要就是为这个按钮添加事件处理程序。

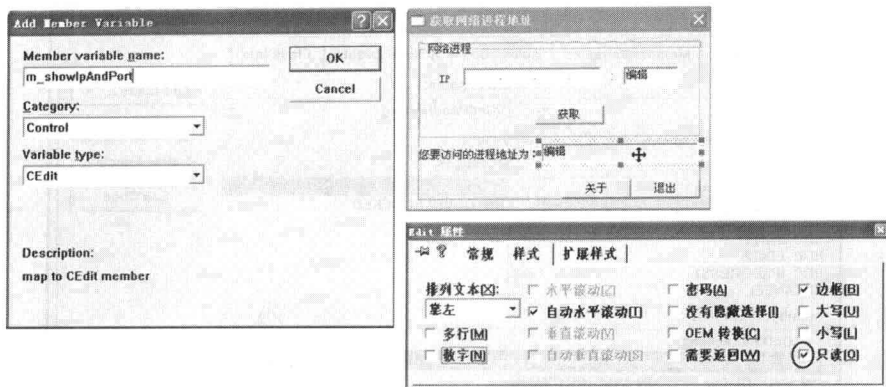


图 1.17 设置显示框关联变量和属性

打开“MFC ClassWizard”对话框,如图 1.18 所示,在“Message Maps”选项卡的“Object IDs”列选中“IDC_BUTTON1”(“获取”按钮控件的 ID 号),在右边“Messages”列选择“BN_CLICKED”,单击“Add Function”按钮,弹出如图 1.19 所示的“Add Member Function”对话框。

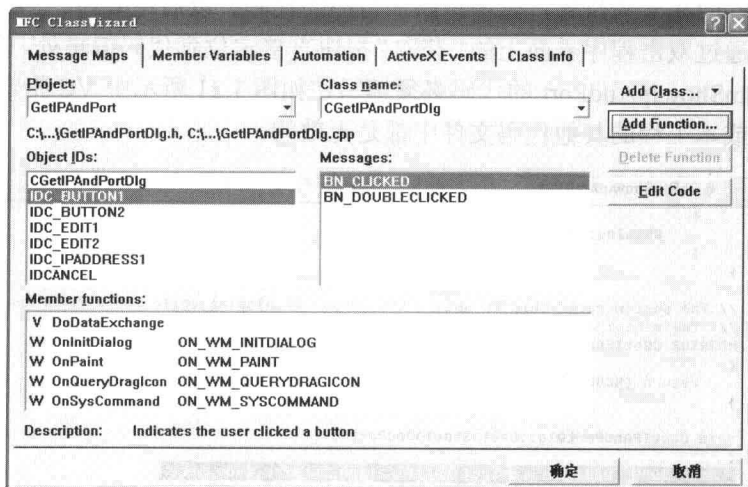


图 1.18 为“获取”按钮添加事件处理函数

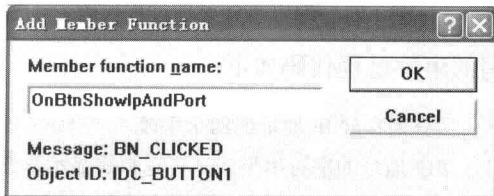


图 1.19 为事件处理程序命名

在如图 1.19 所示的对话框中命名这个处理程序为 OnBtnShowIpAndPort。

单击“OK”按钮,在下方的“Member functions”列表中可以看到新添加的处理函数,如图 1.20 所示。

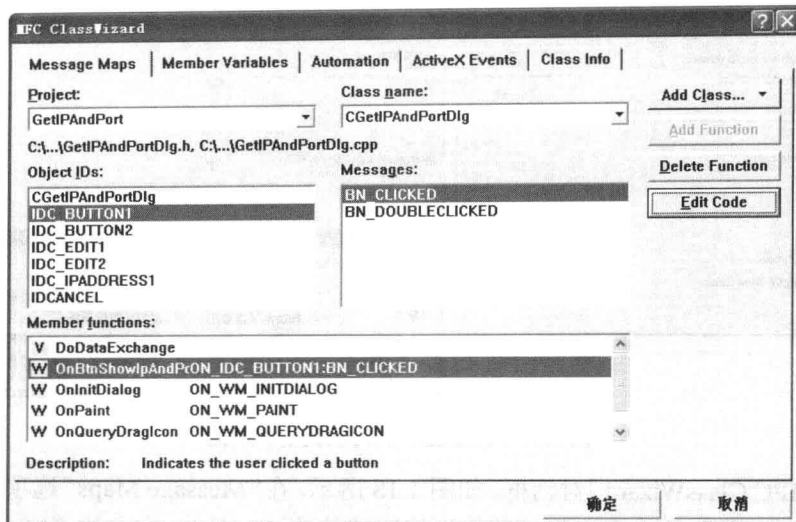


图 1.20 成功添加事件处理函数

单击“Edit Code”按钮，VC 自动打开需要编辑的程序代码文件 GetIPAndPortDlg.cpp 并定位到源文件中相应的位置，供用户添加自己的代码。以后在编译、调试程序的时候，读者也可随时通过双击程序界面上的“获取”按钮直接定位到代码编辑处。需要指出的是，事件过程 OnBtnShowIpAndPort 的代码**必须**填写在如图 1.21 所示中 VC 指定的地方，写在其他任何地方或本工程的其他代码文件中都是无效的。

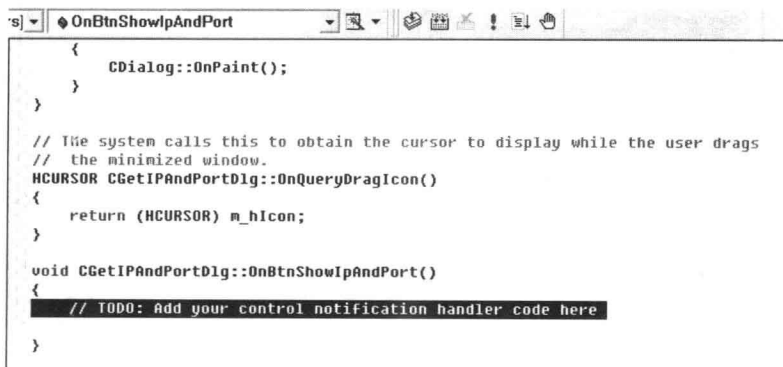


图 1.21 编辑事件处理过程代码

为“获取”按钮编写的事件过程代码如下：

```

BYTE nField[4];           //分别存放 IP 地址的四个字段
CString sip;              //IP 地址的字符串形式（可以直接显示在界面上）
UpdateData();             //刷新对话框界面，获取用户输入
//验证输入是否合法
if(m_ip.IsBlank())        //若用户没有填写 IP，则提示填写
{
    AfxMessageBox("请填写 IP 地址!", "提示", MB_OK|MB_ICONWARNING);
    return;
}

```