

# C++ Builder 6

## 彻底研究

陈灿煌 著

内附范例光盘

```
STDMETHODIMP Tmedia_serverImpl::Get_FileStr(VARIANT* Value)
{
    try
    {
        *Value=Variant(fileNameString);
    }
    catch(Exception &e)
    {
        return Error(e.Message.c_str(), IID Tmedia_server);
    }
    return S_OK;
};
STDMETHODIMP Tmedia_serverImpl::Set_FileStr(VARIANT Value)
{
    try
    {
        FileNameString=Value;
    }
    catch(Exception &e)
    {
    }
}
```

<vcl.h6>

Form6

Variant

■ C++ Builder 6是Borland公司的最新版本应用开发工具

本书从基础开发进行介绍,

由浅入深地向读者展示C++ Builder 6的强大功能

■ 本书详细地介绍了C++ Builder 6在数据库开发中的方法和技巧

■ 结合实际案例

讲解如何使用C++ Builder 6构建传统的客户/服务器应用程序以及如何快速开发Web应用程序

■ 本书内容丰富、资料全面,适合中高级读者或开发人员使用

中国铁道出版社  
CHINA RAILWAY PUBLISHING HOUSE

TP312C

31

# C++ Builder 6 彻底研究

陈灿煌 著

中国铁道出版社

2003·北京

## (京)新登字 063 号

北京市版权局著作合同登记号: 01-2002-4463 号

### 版 权 声 明

本书中文繁体字版由台湾博硕文化股份有限公司出版, 2002。本书中文简体字版经台湾博硕文化股份有限公司授权由中国铁道出版社出版, 2002。任何单位或个人未经出版者书面允许不得以任何手段复制或抄袭本书内容。

本书贴有博硕文化激光防伪标签, 无标签者不得销售。版权所有, 侵权必究。

### 图书在版编目(CIP)数据

C++ Builder 6 彻底研究/陈灿煌著. —北京: 中国铁道出版社, 2002. 11

ISBN 7-113-05019-0

I. C… II. 陈… III. C 语言-程序设计 IV. TP312

中国版本图书馆 CIP 数据核字(2002)第 092047 号

书 名: C++ Builder 6 彻底研究

作 者: 陈灿煌

出版发行: 中国铁道出版社(100054, 北京市宣武区右安门西街 8 号)

策划编辑: 严晓舟 郭毅鹏

责任编辑: 苏 茜 王占清

封面设计: 孙天昭

印 刷: 河北省遵化市胶印厂

开 本: 787×1092 1/16 印张: 51.75 字数: 1267 千

版 本: 2003 年 2 月第 1 版 2003 年 2 月第 1 次印刷

印 数: 1~4000 册

书 号: ISBN 7-113-05019-0/TP·821

定 价: 84.00 元

版权所有 侵权必究

凡购买铁道版的图书, 如有缺页、倒页、脱页者, 请与本社计算机图书批销部调换。

# 出版说明

C++ Builder 6 作为一个功能强大且易于上手的程序开发工具，得到越来越多的编程人员的青睐。用 C++ Builder 6 可以轻易地通过拖放组件来组建自己的应用程序，或一些编程人员调用 Windows API 编写 DLL 程序等开发出专业级别的高级软件。本书覆盖面甚广，共分为四大篇：Windows 程序设计、数据库程序设计、互联网程序设计及高级程序设计，内容独到经典，是一本学习和处理 C++ Builder 6 问题的参考好书。

本书由台湾博硕文化股份有限公司提供版权，经中国铁道出版社计算机图书中心审选，由彭木根、贾英茂、游广志、刘宇、朱远波、张新东、张琦、王东、刘小娜、陈贤淑及廖康良等同志参与了本书的整稿及编排工作。

2003 年 1 月

10570/06

# 目 录

## 第一部分 Windows 程序设计

<b>第 1 章 认识 C++ 语言</b> .....	3
1-1 你的第一个 C++程序 .....	4
1-2 基本数据类型(Basic Data Type) .....	7
1-3 运算符(Operators) .....	8
1-4 数组(Array) .....	9
1-5 指针(Pointer) .....	11
1-6 条件语句(Conditional Statement) .....	12
1-7 循环(Loop) .....	15
1-8 函数(Function) .....	16
<b>第 2 章 可视化设计的集成开发环境(IDE)</b> .....	23
2-1 代码编辑器(Code Editor) .....	24
2-2 代码浏览器(Code Explorer).....	29
2-3 窗体(Form).....	29
2-4 组件面板(Component Palette).....	31
2-5 对象检查器(Object Inspector).....	33
2-6 对象树状窗口(Object TreeView).....	34
2-7 加速栏(Speed Bar).....	35
2-8 弹出式菜单(Popup Menus) .....	35
2-9 调试器(Debugger) .....	36
2-10 所有工具都可 Dockable .....	48
2-11 联机帮助(On Line Help) .....	51
2-12 项目程序结构 .....	53
2-13 对象库的应用 .....	71
<b>第 3 章 C++ Builder 的程序设计基本原理</b> .....	75
3-1 对象的基本概念 .....	76
3-2 C++ Builder 提供的对象 .....	77
3-3 组件的继承 .....	81
3-4 组件的有效范围 .....	84



3-5	创建非可视化组件 .....	87
3-6	文字输入控制组件 .....	88
3-7	选项功能控制组件 .....	95
3-8	信息驱动操作方式 .....	101
3-9	读取鼠标信息 .....	104
3-10	窗口鼠标拖—放程序的编写 .....	107
3-11	读取键盘信息 .....	109
3-12	读取对象焦点信息 .....	111
<b>第 4 章</b>	<b>窗口程序的菜单设计 .....</b>	<b>117</b>
4-1	窗口程序的菜单设计种类 .....	118
4-2	C++ Builder 提供的菜单组件 .....	118
4-3	下拉式菜单及右键菜单的设计 .....	119
4-4	Button 和 BitBtn 菜单的设计 .....	125
4-5	多选项卡窗口的设计 .....	127
4-6	TToolBar 及 TImageList 组件的应用 .....	130
4-7	可视化的 VCL 组件都支持 dock 功能 .....	132
4-8	Action List 组件 .....	135
<b>第 5 章</b>	<b>窗口与窗口之间的关系与窗口的类别 .....</b>	<b>143</b>
5-1	什么是窗口 .....	144
5-2	VCL 提供的窗口类别 .....	144
5-3	窗口与窗口之间的关系 .....	146
5-4	动态产生窗口对象 .....	152
5-5	MDI 应用程序的设计 .....	155
5-6	程序 LOGO 窗口的设计 .....	164
5-7	信息提示窗口的应用 .....	167
5-8	标准对话框窗口的应用 .....	173
5-9	可视化的窗口继承 .....	182
<b>第 6 章</b>	<b>Windows 标准界面组件应用及 C++ Builder 的特殊语法 .....</b>	<b>185</b>
6-1	什么是 Windows 标准界面组件 .....	186
6-2	图形列表组件 (ImageList) .....	187
6-3	树状视图组件 (TreeView) .....	190
6-4	窗体视图组件 (ListView) .....	194
6-5	其他 Win32 控制组件 .....	198
6-6	C++ Builder 的特殊语法 .....	205
6-7	剪贴板的高级应用 .....	220
6-8	TStream 的应用 .....	223

第 7 章	Windows 程序设计高级技巧 .....	229
7-1	在 C++ Builder 的应用程序中调用 Win32 API .....	230
7-2	编写及调用 DLL 程序 .....	239
7-3	多国语言的程序开发功能 .....	251
7-4	编写应用程序的 On Line Help .....	257
7-5	开发 Windows NT Service 程序 .....	267
7-6	图形处理 .....	271
7-7	多媒体系统的开发 .....	288
7-8	文件处理 .....	289
7-9	如何在 C++ Builder 中控制 Microsoft Office .....	293
7-10	利用 C++ Builder 编写 OLE Server .....	297
7-11	打包 C++ Builder 开发的应用程序 .....	302

## 第二部分 数据库程序设计

第 8 章	数据库程序设计概念 .....	315
8-1	主从结构的实际内涵 .....	316
8-2	C++ Builder 的 Two-Tier 主从结构 .....	318
8-3	C++ Builder 的 Multi-Tier 结构 .....	321
8-4	C++ Builder 的多人组合开发环境 .....	325
8-5	对象库 (Object Repository) .....	326
8-6	数据字典 (Data Dictionary) .....	328
8-7	数据模块 (Data Module) .....	334
8-8	数据库维护辅助工具 .....	338
第 9 章	数据库应用程序设计基础 .....	349
9-1	创建第一个数据库应用程序 .....	350
9-2	功能强大的字段编辑器及 TField 组件 .....	351
9-3	什么是 TDataSet .....	358
9-4	设计数据库维护程序 .....	359
9-5	设计数据库搜索功能 .....	366
9-6	什么是 SQL .....	371
9-7	使用 SQL 语法的数据库程序设计方式 .....	373
9-8	使用保存在服务器端数据库上的 SQL 程序 (Stored Procedure) .....	385
9-9	数据库控制组件的应用 .....	391
第 10 章	数据库应用程序设计高级技巧 .....	405
10-1	分析使用 BDE 组件的效率及差异 .....	406
10-2	数据集 (DataSet) 的应用 .....	411

10-3	Database 的 Isolation Levels .....	414
10-4	文本文件与 SQL 数据库之间的转换.....	416
10-5	更新多个数据表产生的查询结果.....	419
10-6	BDE 数据库程序的错误信息管理.....	423
10-7	一对多数据表的设置 .....	429
10-8	连接权限及事务数据的控制.....	431
<b>第 11 章</b>	<b>Multi-Tier 数据库应用程序设计基础 .....</b>	<b>437</b>
11-1	Multi-Tier 数据库程序设计原理.....	438
11-2	在开始编写 Multi-Tier 应用程序之前 .....	440
11-3	编写 Multi-Tier 的基本数据维护程序 .....	446
11-4	如何在 Multi-Tier 的程序中进行 Transaction.....	449
11-5	如何把 SQL 命令从客户端程序传给应用程序服务器运行.....	459
11-6	如何把查询参数传给应用程序服务器上的 TQuery .....	470
<b>第 12 章</b>	<b>Multi-Tier 应用程序设计的高级技巧 .....</b>	<b>475</b>
12-1	Single Instance 及 Multiple Instance 的差异.....	476
12-2	Multi-Tier 的错误处理机制.....	477
12-3	把应用程序服务器编写成 NT Service 类型.....	482
12-4	客户端程序与应用程序服务器之间的数据传递.....	484
12-5	可以让你在客户端设置 master/detail 的关系 .....	485
12-6	TClientDataSet 组件的应用.....	488
12-7	控制客户端用户权限 .....	498
12-8	TDataSetProvider 组件的应用.....	506
12-9	支持 MTS (Microsoft Transaction Server) .....	510

### 第三部分 互联网应用程序设计

<b>第 13 章</b>	<b>Web 应用程序设计概念 .....</b>	<b>529</b>
13-1	目前 Web 应用程序开发方式有哪些.....	530
13-2	Web 结构的实际含义 .....	537
13-3	C++ Builder 在互联网上的强大支持 .....	540
13-4	开发环境及系统运行环境需求.....	544
13-5	服务器操作系统安装及设置.....	547
13-6	系统的整体安全规划及设置实现.....	550
<b>第 14 章</b>	<b>利用 C++ Builder 开发一般的 CGI 及 ISAPI 应用程序.....</b>	<b>557</b>
14-1	什么是 CGI (Common Gateway Interface) .....	558
14-2	CGI 的实际运行原理.....	559



14-3	CGI 程序与数据库之间的关系.....	571
14-4	什么是 ISAPI.....	577
14-5	ISAPI 的实际运行原理.....	578
<b>第 15 章</b>	<b>利用 C++ Builder 的 Web Broker 机制开发互联网应用程序.....</b>	<b>581</b>
15-1	C++ Builder 所提供的 Web Broker 组件.....	582
15-2	编写第一个 Web Broker 应用程序.....	582
15-3	数据输入窗体的 Web Broker 应用程序.....	588
15-4	查询数据库的 Web Broker 应用程序.....	593
15-5	开发 Microsoft IIS 专用的应用程序(ISAPI).....	603
15-6	开发 Apache Server 专用的应用程序.....	609
<b>第 16 章</b>	<b>如何应用 C++ Builder 的 Internet Express 技术.....</b>	<b>615</b>
16-1	什么是 Internet Express.....	616
16-2	编写单文件维护程序.....	621
16-3	编写一对多维护程序.....	631
16-4	编写查询程序.....	634
16-5	利用 QueryForm 来编写窗体输入程序.....	640
<b>第 17 章</b>	<b>利用 C++ Builder 开发 ActiveX 的应用程序.....</b>	<b>647</b>
17-1	如何利用 C++ Builder 开发 Active X 应用程序.....	648
17-2	在 ActiveX 程序中访问远程数据库.....	655
17-3	如何 Deploy 开发完成的 ActiveX 应用程序.....	656
17-4	如何利用 Package 功能来达到缩小 ActiveX 的目的.....	659
17-5	如何在互联网上实现 ActiveX.....	664
17-6	改善 Socket 的传输效率及安全性.....	665
17-7	Load Balancing 及 Fail Over 的设置.....	668
<b>第 18 章</b>	<b>利用 C++ Builder 的 Web Snap 机制开发互联网应用程序.....</b>	<b>671</b>
18-1	利用 Web Snap 功能编写 Web 应用程序.....	672
18-2	利用 Web Snap 编写单文件维护程序.....	679
18-3	利用 Web Snap 的 Login 机制.....	682
<b>第 19 章</b>	<b>C++ Builder 提供的 XML 机制.....</b>	<b>691</b>
19-1	什么是 XML.....	692
19-2	如何利用 C++ Builder 处理 XML 文档.....	698
19-3	利用 Data Binding 向导来处理 XML 文档.....	702
19-4	如何利用 XML Mapping 工具程序.....	711
19-5	C++ Builder 提供的 XML 组件.....	716
19-6	实际应用范例说明.....	716

<b>第 20 章 C++ Builder 提供的 Web Service 机制</b> .....	719
20-1 什么是 Web Service、SOAP、WSDL .....	720
20-2 C++ Builder 如何支持 Web Service .....	726
20-3 编写 Web Service 的服务器端程序 .....	727
20-4 编写 Web Service 的客户端程序 .....	732
20-5 从你的 Web 程序去调用别人的 Web Service .....	744
20-6 包含 DataModule 的 Web Service .....	750

## **第四部分 高级程序设计**

<b>第 21 章 报表、统计图表、商业决策分析应用</b> .....	755
21-1 QuickReport 组件的应用.....	756
21-2 TPrint 组件的应用.....	765
21-3 C++ Builder 提供的商业统计图表组件.....	773
21-4 使用 Chart 组件.....	774
21-5 使用 DBChart 组件.....	782
21-6 利用 C++ Builder 来开发商业决策分析应用程序.....	791
21-7 以统计图表来显示决策分析结果.....	798
<b>第 22 章 利用 C++ Builder 开发 COM 的应用程序</b> .....	801
22-1 COM 的运行原理.....	802
22-2 C++ Builder 如何支持 COM 机制.....	804
22-3 编写 COM 的服务器端程序及客户端程序.....	809

第 1 章 认识 C++ 语言

第 2 章 可视化设计的集成开发环境(IDE)

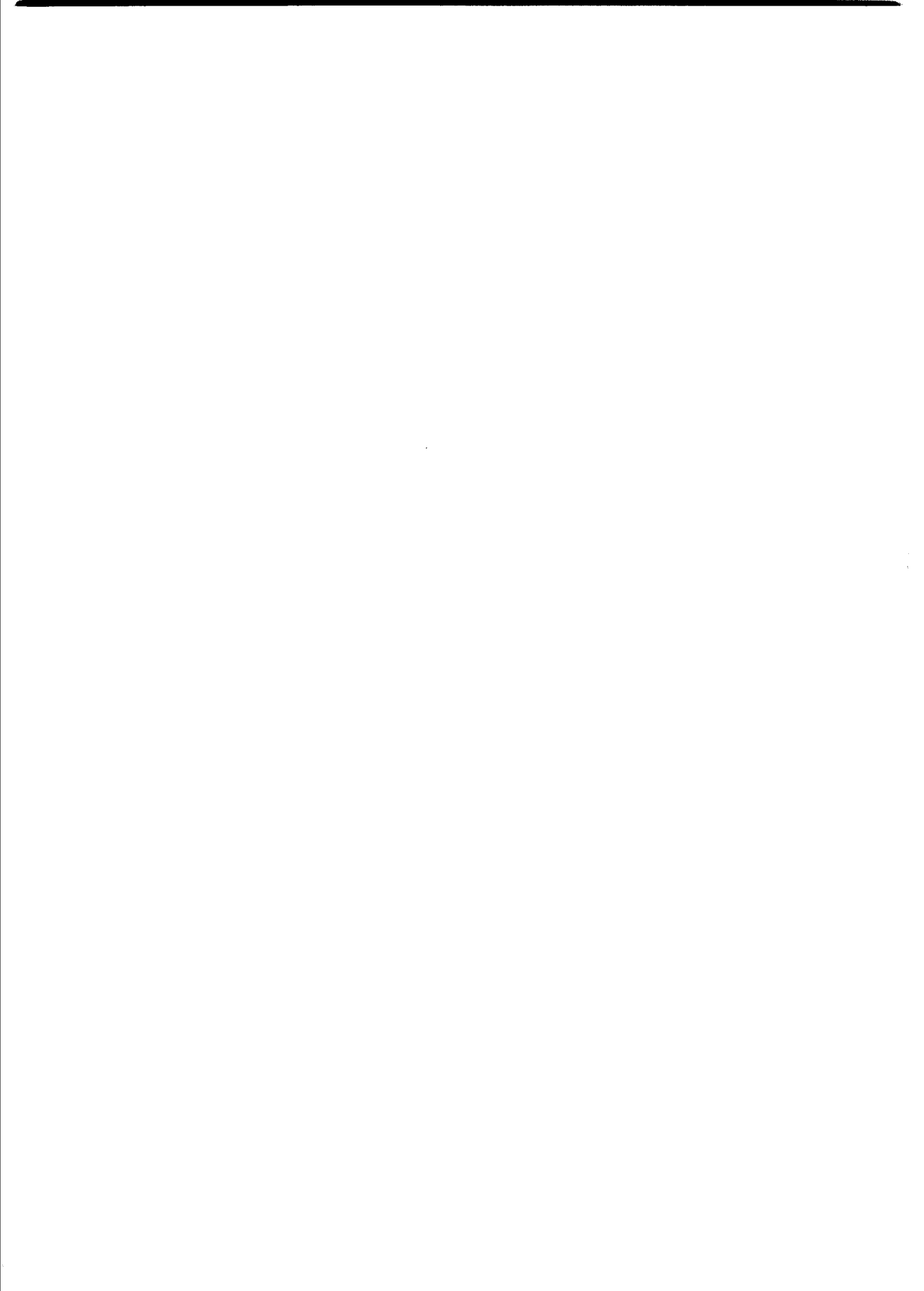
第 3 章 C++ Builder 的程序设计基本原理

第 4 章 窗口程序的菜单设计

第 5 章 窗口与窗口之间的关系与窗口的类别

第 6 章 Windows 标准界面组件应用及 C++ Builder 的特殊语法

第 7 章 Windows 程序设计高级技巧



# 第 1 章

## 认识 C++ 语言

- ◆ 你的第一个 C++ 程序
- ◆ 基本数据类型 (Basic Data Type)
- ◆ 运算符 (Operators)
- ◆ 数组 (Array)
- ◆ 指针 (Pointer)
- ◆ 条件语句 (Conditional Statement)
- ◆ 循环 (Loop)
- ◆ 函数 (Function)

### 1-1 你的第一个 C++ 程序

本书的书名为《C++ Builder 6 彻底研究》，所以本书的重点还是在 C++ Builder 这个开发工具，本书会深入探讨如何彻底利用 C++ Builder 这套工具来开发各种应用程序。不过，在开始深入了解 C++ Builder 这个好用的 C++ 开发工具之前，不可避免的，我们还是先来回顾一下 C++ 本身语法的一些基本概念及应用。

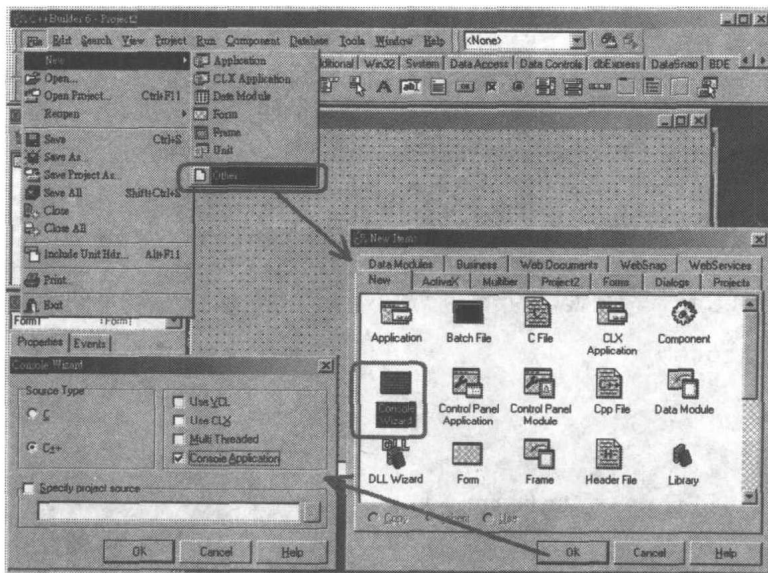


图 1-1 C++ Builder 提供的 Console Wizard

由于 C++ Builder 本身是一套 Windows 程序的开发工具，所以它所提供的 Framework 主要还是偏重在图形界面的 Windows 程序，不过它仍旧提供文字的 console 模式，由于本章的目的主要在于让读者了解 C++ 语法的应用，所以不必大费周章地采用图形界面，因此本章的范例程序都是 console 模式，在 C++ Builder 中要产生 console 模式的程序结构很简单，只要点击选中 [File] → [New] → [Other] 打开 Object Repository，选择 [New] → [Console Wizard] 即可启动向导程序，由图 1-1 可以看出此向导有许多选项，请你点击选中 C++ 及 Console Application 两项即可，最后点击选中 [OK]，C++ Builder 将会自动帮你产生 Console 程序，如下所示：

```
//-----  
#pragma hdrstop  
//-----  
#pragma argsused  
int main(int argc, char* argv[])  
{  
    return 0;  
}  
//-----
```

基本上，这个程序只是个空框架，没做任何事，此时你就可以在这个空框架上开始做文

章，假设我们希望能显示一段文字，那么你就可以把程序改写成：

```
//-----  
// C++ Builder 彻底研究范例  
//-----  
#include <iostream.h>  
#include <conio.h>  
  
#pragma hdrstop  
//-----  
#pragma argsused  
int main(int argc, char* argv[])  
{  
    /* 第一个 C++ 程序 */  
    cout<<"Hello, C++ World";  
    getch();  
    return 0;  
}  
//-----
```

如果你运行上面这个程序后，将会得到如图 1-2 所显示的结果。Windows 会打开一个 DOS 窗口，此程序将“Hello, C++ World”的信息显示在 DOS 窗口里，接下来，你只要按一下[Enter]键，程序就会结束运行。接着，让我们仔细的来看看此范例程序中的每一行代码所代表的意义。

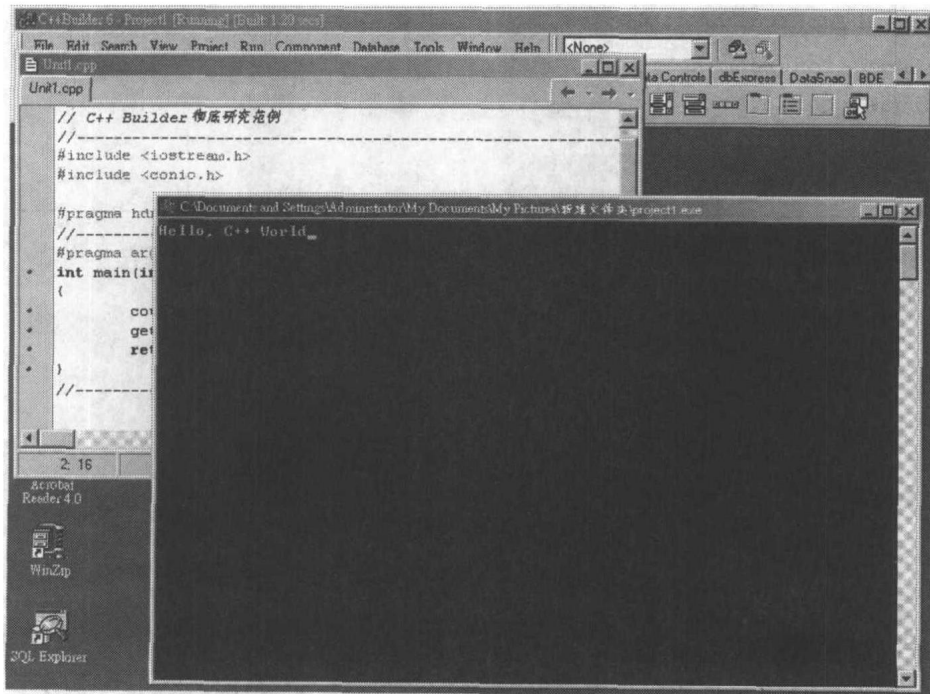


图 1-2 第一个 C++程序的运行结果 (console)



首先看到的是含有#include 指令的#include <iostream.h>及#include <conio.h>,它们的作用在于引入头文件(header file)。当运行编译时,include 指令会指示 C++ Builder 编译器把括号内的头文件include到我们的代码中。一般的头文件均是以.h或.hpp作为文件的扩展文件名。多数的头文件是编译器的厂商所提供,有的则是程序设计师预先设计好供人们(或自己)使用。依照多数程序设计习惯,我们通常会使用#include <文件名>来引入编译器厂商内置的头文件(系统默认路径下),至于自己所设计的头文件,则使用#include "文件名"(用户程序路径下)。

至于程序中使用双斜线//及/\* \*/符号来标识的部分,代表注释的意思,其后同一行的文字仅作为注释用,与程序内容无关;另外一个用来注释的符号是/\*,它的用法与//略有不同,/\*必须与\*/成对出现,注释文字则放在 /\* 和 \*/之间,它的范围不限于同一行。

C++程序由 main()开始运行,这是任何一个 C++程序的开始运行的地方或称为入口点(Entry point)。在这程序中,跟随着关键字 main 的小括号“( ”及“ )”之间并没有任何的参数(argument),这表示程序运行时我们不需要由程序外部输入任何参数值。而大括号“{”及“}”则代表 main()程序的开始与结束。

cout 指令则是负责把字符串输出到屏幕上,此指令的定义放在 iostream.h 头文件内,这也是我们为什么要把 iostream.h 文件 include 进来的原因。而由于运行环境是在 Windows 下,所以如果 console 程序只显示字符串,那么运行结果将一闪而过,你必须加上 getchar()函数,让程序运行完显示字符串后停下来等待输入,才能清楚看到程序的运行结果。而 getchar()函数则是被定义在 conio.h 头文件内。

由于本章的范例程序只会用到 console 的输入及输出,所以下面再介绍一个包含 console 输入及输出指令的范例程序:

```
//-----  
#include <iostream.h>  
#include <conio.h>  
  
#pragma hdrstop  
//-----  
#pragma argsused  
int main(int argc, char* argv[])  
{  
    int n;  
    cout<<"输入一个整数 :";  
    cin>>n;  
    cout<<"\n 你输入的整数是 : "<<n<<endl;  
    cin>>n;  
}  
//-----
```

这个程序中,首先定义一个整数名称为 n,接着使用 cout 指令输出字符串提示输入一个整数,而 cin 指令则会让程序接收用户由键盘输入的整数,并且把值放入 n 变量中。运行结果如图 1-3 所示。

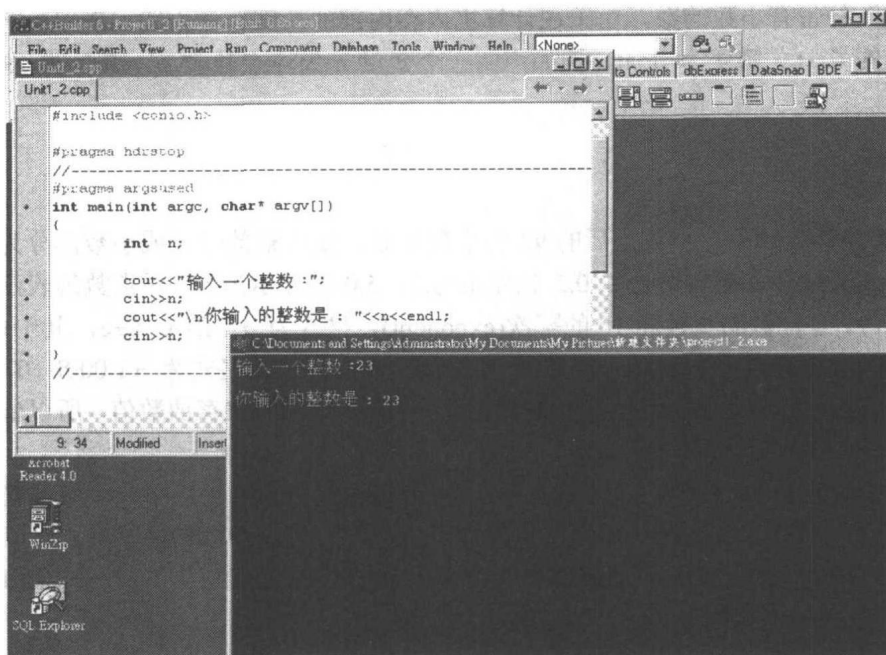


图 1-3 console 输入及输出指令的程序运行结果

到此你应该了解如何打开及运行一个 C++ Console 程序，也大略了解 C++ 最基本的结构了吧，接下来我们再来看看 C++ 的基本语法。

## 1-2 基本数据类型(Basic Data Type)

变量是程序中用来保存数据的内存块的标识名称(Identifier)，指定一个标识名称的 C++ 命令称为变量声明(Variable Declaration)，每一个变量声明的指令至少要包含变量的数据类型(data type)及变量名称(name)，变量的数据类型用来决定将存放在内存的变量值的类型(如 int、char 等)，变量名称则是作为内存位置的标识来用。

你可以使用任意的字符和数字及下划线( \_ )的组合作为变量名称，但是有两个限制，一个是不得使用 C++ 的保留字作为变量名称，另一个则是不得在名称中使用特殊字符(如 @、\$ 和空格键)或算术运算符符号(+、-、\*、/、%)。在 C++ 中，变量名称的大小写是有区别的，例如 name、Name、NAME 代表三个不同的变量。

当我们要将一个数据放入变量所指的内存中，必须使用 Assignment 的指令，上述的 Assignment 指令就是 "=" 符号，要特别注意的是 "=" 符号在 C++ 里是被用作 Assignment，而不是相等(Equality)的意思。这是许多初学者常混淆的地方。

C++ 提供有不少的基本数据类型，其中最常使用到的四个基本的数据类型分别是整数(Integer)、浮点数(Floating Point)、双精度浮点数(Double)和字符(Character)。

整数类型包括所有自然数(int、正数、负数及零)、没有符号的整数(unsigned int)、短整数(short)及长整数(long)，用来定义整数类型的关键字为 int。在一个 32 位的操作系统(如 Unix、Linux 或 Windows 2000)中，一个整数的变量使用 32 位(4 bytes)的存储空间，其值的范围是从 2,147,483,647 到 -2,147,483,648。