Borland C++ Builder 3.0

自学培训教程

〔美〕K. Reisdorph 著
希望图书创作室 译
龙启铭 校

本书配套光盘包括两部分内容：
1. 本书配套电子书
2. 赠送“计算机基础知识全面速成”多媒体学习软件

北京希望电脑公司出品
1998

内 容 简 介

Borland C++ Builder 是以 C++ 为编写语言的快速应用程序开发(RAD)工具。Borland C++ Builder 3.0 是其最新版本。本书从 C++ 基础概念开始，详细介绍了 C++ Builder 的编程环境、VCL 构件编程、C++ Builder 的各种工具、DLL、数据库应用开发，以及 Internet 编程等内容。全书内容按三周课程安排，内容由浅入深，实例丰富，是从事 Borland C++ Builder 应用与开发的广大技术人员重要参考书，也是大专院校相关专业自学、教学用书和社会同类培训班教材。

欲购本书或需技术支持的用户，请直接与北京海淀 8721 信箱书刊部联系，电话：010-62562329、010-62541992；或传真至 010-62579874，邮编：100080

版 权 声 明

本书英文版名为《Sams' Teach Yourself Borland C++ Builder 3 IN 21 DAYS》，由西蒙与舒斯特国际出版公司出版，版权归西蒙与舒斯特国际出版公司所有。本书中文版由西蒙与舒斯特国际出版公司授权出版。未经出版者书面许可，本书的任何部分都不得以任何形式或手段复制与传播。

版权所有，翻印必究。

Borland C++ Builder 3.0 自学培训教程

〔美〕K. Reisdorph 著

希望图书创作室 译

龙启铭 校

责任编辑：陆卫民

北京希望电脑公司出品

北京海淀路 82 号(100080)

北京媛明印刷厂印刷

新华书店、新华书店音像发行所发行、各地专卖店经销

* * * * *

1998 年 11 月第 1 版　　1998 年 11 月第 1 次印刷

开本 787×1092　1/16　　印张：34.625

字数：785 千字　　印数：1-5000

新出音管[1998]164 号

ISBN7-980021-07-X/TP・00

定价：50.00 元(1CD，含配套书)

译 者 序

本书是学习 BORLAND C++ BUILDER 3.0 使用的综合性图书，我们很高兴把它翻译过来，献给读者。本书翻译过程中，下列同志参与了翻译、整理和录排工作，在此表示衷心感谢：王凌飞，钟埕光，张荣，李青，赖华龙，刘文红，刘云昌，陈凌峰，陈莼颍，周阳生，邹能东，李耀平，彭振庆，黄志坚，黄其荣，袁卫东，何平，袁文卫,付宇光，陈立平，付文辉，赖玉洪，李坤荣。

邱仲潘

1998 年 10 月

作者简介

Kent Reisdorph 是 TurboPower 软件公司的高级软件工程师。他还拥有自己的顾问公司。Kent 是 Cobb 集团的《C++ Builder 开发月刊》的编辑，是 TeamB 的成员之一，每周花许多时间在 Borland 新闻组中，回答 C++ Builder 和一般 Windows 编程方面的问题。Kent 与妻子 Jennifer 住在科罗拉多的科罗拉多泉附近，同住的还有六个小孩 James，Mason，Mallory，Jenna，Marshall 和 Joshua。

致　谢

这么一本书自然不是在真空中写成的，而是得到了许多人的参与，包括麦克米兰和 Borland 公司的人。感谢麦克米兰公司的 Steve Sayre、Angie Allen、Renee Wilmeth 和 Sean Dixon 的帮助，本书进行中曾发生一些挫折，但麦克米兰公司的朋友们一直干劲实足。

感谢 Borland 公司的 Richard Army、Celeste Crocker 和 Nan Borreson。Nan 曾被本书折腾得够呛。Terri Bartos 也提供了很大的帮助，回答了许多重要问题。Terri 一直忙于 C++ Builder 工作，但总能不厌其烦地回答我的问题，并进行了一些技术编辑。另一个要特别感谢的 Borland 公司的朋友是 Ellie Peters（以及 Jeff）。Ellie 对本书进行了大量技术编辑，更重要的是她是我在 Borland 公司的内部联系人。需要了解某个问题时，我只要给 Ellie 一个电话或 e-mail，准能得到帮助。Jeff 和 Ellie 是好朋友，很高兴他们给本书提供了帮助。

感谢 TurboPower 软件公司的人们在我编写本书期间给予的关照。感谢 BobDelRossi 和 Lee Inman 在道义上的支持，感谢 Ralph Trickey 和 Terry Hughes 对数据库章节所做的技术编辑。

最后，感谢我的妻子 Jennifer。当我着手本书工作时，她为我承担了所有杂务，直到本书搁笔。她把一切照顾得妥贴而毫无抱怨之词。没有她就不可能有本书。

前　言

当你手头捧着这本书时，也许是因为你有多年的C++编程经验，希望了解C++Builder的快速应用程序开发(RAD)，也许因为你用过Borland的Delphi，想把这个知识用到C++编程环境中，也许是老板要你阅读本书，也许你是完全初学者，想了解Windows编程的美妙世界。

不管原因如何，都欢迎你。我相信本书是个有趣的读物，你一定会有同感的。书中的确要涉及一些工作，但绝不会让你感到乏味。将思路转变成可行的Windows程序非常方便。

阅读本书时，建议读者多多尝试。放下书，玩一玩，这比最好的老师更有帮助。读者还应多多参考各种联机资源，特别是Borland新闻组forums.borland.com，其中有Borland TeamB(自愿组织)的成员和Borland用户对各种深浅问题的答案。还要看看Borland的Web站点www.borland.com，其中有C++ Builder的进一步细节和更新信息。到www.mcp.com/info可以下载本书的源码和两个附录中的源码。

无论你是哪种读者，都希望你学习愉快。轻轻松松，边玩边学，祝你成功。

技术支持

如果需要关于本书的帮助，请与我们的技术支持部联系，电话为317-581-3833，e-mail地址为support@mcp.com。

请提宝贵意见

为了进一步提高我们的图书质量，欢迎读者提出宝贵意见和建议。读者是最重要的书评和批评家。我们重视读者的意见，切盼了解我们自己的成败，以期更上一层楼。

请提宝贵意见，联系地址为

programming@mcp.com

或者

Macmillan Computer Publishing
Programming Group
201 W. 103rd Street
Indianapolis, IN 46290

试读结束，需要全本PDF请购买 www.ertongbook.com

第一周

第一周要介绍如何用C++编写 Windows 程序。C++语言比较难学，但它是世界上许多公司和政府的标准编程语言。学习C++不太容易，但无论在技能上还是在经济上，回报都是丰厚的。

本周前四课要学习C++语言的基础。学完前四课后，你就可以编写简单测试程序来巩固学到的C++语言知识。但是，这些程序并不是你要购买C++ Builder 来编写的程序。前四课的测试程序是控制台应用程序，就像DOS 程序一样，没有漂亮的用户界面，不能吸引人。但是，这些程序有助于掌握C++的基础。

第5课开始，你就要学习一些与C++ Builder 的图形化编程有关的知识。我会介绍框架以及框架对 Windows 编程人员的意义。第5课要用C++ Builder 图形化编程工具建立简单测试程序。然后要用两天时间介绍C++ Builder IDE 如何使编程工作得以简化。这时的学习就开始有趣起来了。本周最后可以编写一些可执行的 Windows 程序。知道这些计划后，开始学习吧。

第 1 课　C++ Builder 入门

祝贺你选择了当今最热门的编程工具。但在开始使用C++ Builder 之前，要先了解C++的基础。本章介绍的内容包括：

- C++ Builder 简介
- 编写 Win32 控制台方式应用程序的信息
- C++语言简介
- C++变量和数据类型
- C++函数，包括 main()函数
- 关于数组

1.1　何谓C++ Builder

C++ Builder 是 Borland 公司新的快速应用程序开发(RAD)产品，用于编写C++应用程序。利用C++ Builder 可以更加方便快捷地编写C++ Windows 程序。我们可以生成 Win32 控制台应用程序或 Win32 GUI(图形用户界面)程序。用C++ Builder 生成 Win32 GUI 应用程序时，C++的所有功能都包装到 RAD 环境中。这就是说，可以真正利用快速应用程序开发的拖放技术生成程序的用户界面(用户界面指菜单、对话框、主窗口，等等)。也可以将 OCX 控件放落到窗体上，生成 Web 浏览器之类的特殊化程序。C++ Builder 具有所有这些功能，但程序执行速度并不受影响，因为我们仍然有C++提供的功能。

也许你会说，这太妙了，不错，但先别激动，C++语言可不太容易学。你不可能买一个C++ Builder 之类的程序，然后一夜之间变成熟练的 Windows 编程人员。C++ Builder 的确漂亮地隐藏了 Windows 程序的一些低级细节，但C++ Builder 不可能编写程序。最终还得由你来编程，因而你必须学习编程。编程是个难学的工作，但C++ Builder 能使编程变得相当轻松有趣，让你寓工作于快乐之中。

卷起袖子，穿上滑冰鞋，开始玩玩C++ Builder 吧。

1.2　C++ Builder IDE 概览

本节概要介绍C++ Builder IDE(集成开发环境)，第 6 课“C++ Builder IDE 剖析”中将进一步介绍其细节。由于你要进行 Windows 编程，所以我们假设你已经知道如何启动C++ Builder。首次启动这个程序，会出现图 1.1 所示的空窗体和 IDE。

C++ Builder IDE 分为三个部分。顶部的窗口可以看成主窗口，包含左边的工具条和右边的构件板。工具条可以单击访问打开、保存和编译项目的工具。构件板中包含各种可以拖放到窗体上的构件(构件指文本标题、编辑控件、列表框、按钮，等等)。为了方便，构件分成组。注意构件板顶部的标签，单击标签可以得到不同的构件。为了将构件放到窗体上，只要单击构件

板中的构件按钮，然后单击窗体中要放构件的地方。别担心构件的用法，稍后会作介绍。探索完毕后，单击 Standard 标签，稍后要介绍其中的内容。

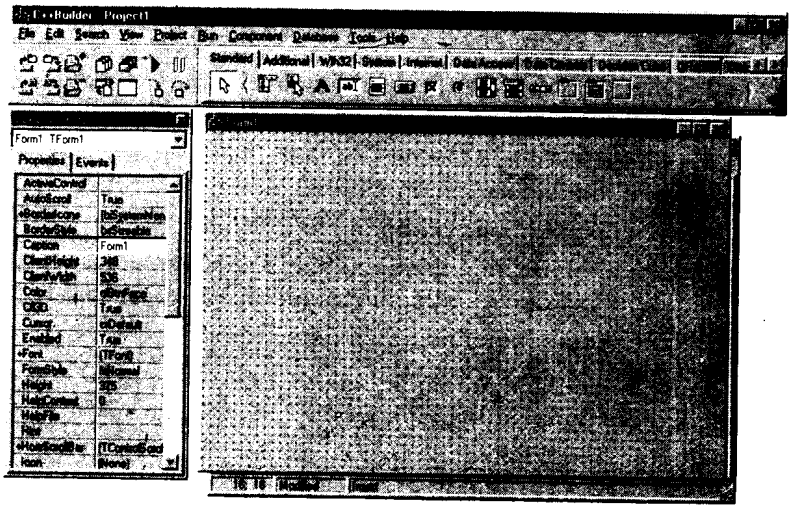

图 1.1 C++ Builder IDE 及其初始空窗体

新术语 构件(component)是自给自足的二进制软件块，可以进行特定的预定义功能，如文本标题、编辑控件、列表框。

工具条和构件板下面的屏幕左边是对象观察器(Object Inspector)。我们要通过对象观察器修改构件属性和事件。使用C++ Builder 要经常用到对象观察器。对象观察器根据所选构件不同，可能有一个或两个标签。对象观察器还有个 Properties 标签。构件属性控制构件如何操作。例如，改变构件的 Color 属性可以改变这个构件的背景颜色。不同构件的属性表名不相同，但构件通常有一些公有属性(如 width 和 Height 属性)。

新术语 属性(property)确定构件的操作方式。

通常，对象观察器除了 property 标签外还有个 Events 标签。用户与构件交互时会发生事件(event)。例如，单击构件时会产生单击事件，表示构件被单击。可以编写响应这些事件的代码，在发生事件时完成特定动作。和属性一样，不同构件的事件表各不相同。

新术语 事件(event)是构件与用户或 Windows 交互时发生的。

新术语 事件处理器(event handler)是响应事件时应用程序调用的方法。

对象观察器右边是C++ Builder 工作空间。工作空间最初显示窗体设计器(Form Designer)，顾名思义，窗体设计器是用于设计窗体的。在C++ Builder 中，窗体表示程序中的窗口。窗体设计器可以在窗体生成过程中放置、移动和缩放构件。

窗体设计器背后是代码编辑器(Code Editor)。代码编辑器中可以输入要编写的程序代码。对象观察器、窗体设计器、代码编辑器和构件板在应用程序建立过程中交互式地工作。

介绍C++ Builder IDE 构件后，下面要实际做一些工作。

1.2.1 Hello World

习惯上，每个编程书籍都先要生成一个程序，在屏幕上显示 Hello World 字样。习惯的势力是难于对抗的，Hello World 也不例外。下面几章要介绍一些细节工作，这里想在介绍C++的

基础之前，先让读者体会一下C++ Builder 的妙处。C++ Builder 能比 Windows 编程环境中现有的其它办法更方便地生成 Hello World 程序。

运行C++ Builder 后，注意空窗体。缺省情况下，窗体名为Form1(窗体名在C++ Builder 中很重要，将在稍后介绍)。窗体左边的对象观察器中显示了窗体的属性。单击对象观察器的标题条，Caption 属性加亮，光标放在其上，等待用户动作(如果 Caption 属性不在视图中，则要滚动对象观察器窗口找到 Caption 属性。属性按字母顺序列出)。输入"Hello World!"改变窗体标题。

说　明　修改属性时，C++ Builder 会飞快显示属性改变的结果。输入新标题时，注意窗体窗口标题相应改变。

然后单击工具条上的 Run 按钮(带蓝色箭头的按钮)，也可以按 F9 或选择主菜单中的 Run|Run)。C++ Builder 开始建立程序，出现图 1.2 所示的编译器状态对话框，注意 C++ Builder 跟踪建立程序所需的文件。过一会儿，编译器状态对话框消失，窗体出现，标题变为 Hello World!。这里运行的程序与空窗体几乎完全一样，也许你注意不到程序显示，因为它在窗体设计器的同一窗体位置上显示(但还是有差别，因为窗体设计器显示对齐栅格，而运行的程序则不显示)。祝贺你用C++ Builder 编写了第一个C++ Windows 程序，多么容易！

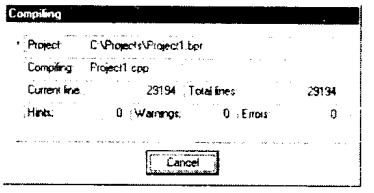

图 1.2　编译器状态对话框

这有什么？是没什么，但却是真正的 Windows 程序，可以拖动标题条进行移动，可以最小化，可以最大化，也可以单击 Close 按钮关闭。甚至可以在 Windows 资源管理器中找到这个程序(可能是\CBuilder\Projects\Project1.exe)并双击运行。

只是在标题中显示 Hello World！可能有点假。下面再深入一步。如果程序 Hello World！还在运行，单击窗口右上角的 Close 按钮将其关闭。这时窗体设计器再次显示，可以修改窗体(及程序)。

为了让程序更可用，可以在窗口中央加进些文本。为此，在窗体上加进文本标题。首先，单击构件板的 Standard 标签，构件板上第三个构件按钮标有 A 字。如将鼠标光标放在这个按钮上，则工具提示会显示 Label。单击标题按钮，然后单击窗体上任一部位，即在窗体上放了一个 Label 构件，缺省标题为 Label1。然后看看对象观察器，这时它显示 Label1 的属性(原先显示的 Form1 的属性)。Caption 属性加亮。单击对象观察器标题条或单击 Caption 属性并输入 Hello World!。这时窗体上的标题显示 Hello World!。还可以改变标题文本的大小。双击 Font 属性，这个属性展开显示其它字体属性。找到 Font 中的 Size 属性并将字体大小变为 24(当前设置为 8)。按 Enter 或单击窗体时，标题立即变为新尺寸。

由于标题可能没有居中显示，还要进行移动。移动构件时，只要单击构件并拖动到所要位置。得到所要的标题后，就可以重新编译和运行程序了。再次单击 Run 按钮。C++ Builder 会再次编译程序，一会儿，程序即运行。这时标题中和窗体中央都显示 Hello World！字样。图1.3

显示了 Hello World! 程序运行的样子。

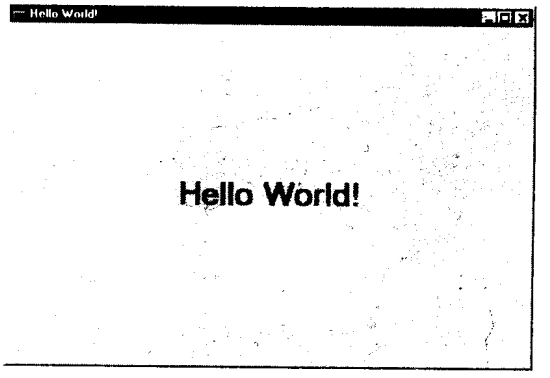

图 1.3 Hello World! 程序运行

从这个例子可以看出，用C++ Builder 编写C++ Windows 程序比过去有趣多了。为了准备后续工作，要先关闭C++ Builder IDE 中的当前项目。选择主菜单中的 File Close All，提示下单击 Yes 存为 Project1 或改存为 Hello World。

1.3 Hello World! 之二，Win32 控制台应用程序

下面两讲要介绍C++语言基础，其中要编写一些简单测试程序。这些测试程序适合作为控制台应用程序，运行时和 DOS 程序相似。Win32 控制台程序与 DOS 程序间有一些本质不同，但这里不必细究，下面是用C++ Builder 生成 Win32 控制台程序 Hello World。

新术语 Win32 控制台程序是在 Windows 95 或 Windows NT 的 DOS 窗口中运行的 32 位程序。

从主菜单选取 File|New，C++ Builder 显示对象仓库。奇怪的是，对象仓库的标题条显示 New Items(新项目)，但其中包含的是预定义项目、窗体、对话框和其它对象，可以加进应用程序中或作为新项目的雏形。第 9 课“在C++ Builder 中生成应用程序”中将详细介绍对象仓库。目前只要单击对象仓库中的 New 标签并双击 Console Wizard(控制台向导)图标。出现控制台应用程序向导对话框时，只要单击 Finish 按钮生成标准控制台程序。C++ Builder 会生成项目并显示代码编辑器，便于输入程序代码。图1.4显示了开始新的控制台应用程序时出现的代码编辑器。

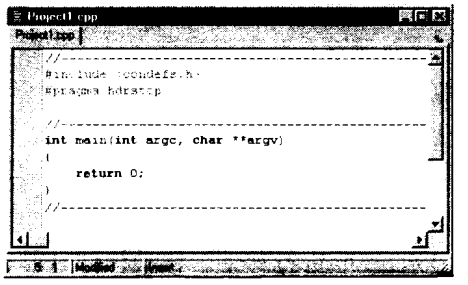

图 1.4 C++ Builder 代码编辑器窗口

试读结束，需要全本PDF请购买 www.ertongbook.com

这时的C++ Builder IDE 与生成GUI 应用程序时有两点不同。首先是没有窗体设计器，因为控制台应用程序是文本方式应用程序，不能显示窗体(这不完全准确，但本讲中可以这么说)。另外，对象观察器中是空的，构件只能放在窗体上，因此在控制台应用程序中对象观察器派不上用场。

提　示　编写控制台应用程序时，可以关闭对象观察器以腾出更多的代码编辑器空间。关闭对象观察器时，单击对象观察器标题条上的 Close 按钮。为了重新打开对象观察器，按 F11 或选择主菜单的 View|Object Inspector。

这时从代码编辑器窗口中可以看到下列文本：

```
//---------------------------------
#include <condefs.h>
#pragma hdrstop
//---------------------------------
int main(int argc,char * * argv)
{
  return 0;
}
//---------------------------------
```

说　明　如果C++ Builder 产生的控制台应用程序代码与上述清单和图 1.4 有出入不必担心。C++ Builder 之类的产品总是不断改进的，有时难免让读者费点心思。

分　析　这是最简单的C++程序，但却是个有效的C++程序。稍后要对这个程序稍作修改，以便让它做点实际工作。首先看看以//开头的语句，这是说明语句，只是在程序中用于分割代码，增加可读性(说明语句通常用于代码建档)。首次生成新的控制台应用程序时，C++ Builder 自动加进这些说明语句(在今后的代码清单中，为了节省篇幅，我们省略了说明语句)。另外，注意代码中以分号结尾的语句(尽管目前没有意义，但事实上本程序中只有一条实际执行语句)。C++程序的每条语句都以分号结尾。

在学习C 和C++语言的早期，编程人员必须先了解表达式与语句之间的差别。语句(statement)的正式定义是带分号的表达式。分号用于结束表达式和让其变成单行代码块。稍后会介绍代码块，这里要说明表达式(expression)是求值数值的代码单元。语句是闭合的表达式。例如下列语句：

```
c=a+b;
```

本例中，等号右边的部分 a+b 是个表达式，整个代码行是个语句。这初看起来不易理解，但读者会慢慢明白的。使用这两个术语时我会特别慎重。但目前只要记住，语句是闭合的带分号的表达式。

还要注意程序开头和结尾的大括号。C++代码块以({)开头，以(})结束。大括号用于表示循环、函数、if 语句等的代码块开头和结尾。本程序中只有一组大括号，因为它是个简单程序。

为了在屏幕上显示 Hello World!，需要利用C++类 iostream，所以要简单介绍一下这个

类(读者也许还不知道什么是类,但稍后会介绍)。iostream 类用流进行基本输入和输出,如在屏幕上打印文本或接受用户输入。cout 流用于向标准输出流发送数据。在控制台应用程序中,标准输出流就是控制台或屏幕。cin 流用于从控制台取得数据。如用户输入。iostream 实现两个特殊操作符,用于将信息放进流中或取得流中的信息。插入操作符(<<)用于将数据插入输出流中,而取得操作符(>>)取得流中的信息。为了将信息输出到控制台中,用下列语句:

```
cout<<"Do something!";
```

这就让程序将 Do something! 插入标准输出流中,执行这行语句时,文本显示的屏幕上。

说 明 cout 只用于控制台应用程序中。Windows GUI 应用程序没有标准输出流(GUI 程序中的一切都基于图形),所以 Windows GUI 应用程序中 cout 输出没有目标。标准 Windows 程序用 DrawText()或 TextOut()在屏幕上显示文本。C++ Builder GUI 程序也可以用 DrawText()或 TextOut(),但更简单的办法是象前面介绍的一样用 Label 构件显示文本。

使用 cout 之前,先要告诉编译器到哪里找 iostream 类的说明(称为声明)。iostream 类的声明放在文件 IOSTREAM.H 中。这个文件称为头文件。

新术语 头(header)文件中包含一个或几个类的类声明。

为了让编译器到文件 IOSTREAM.H 中去找 iostream 类的声明,用#indude 指令如下:

```
#include <iostream.h>
```

新术语 声明(declaration)就是描述类或函数的码段。定义(definition)是。类或函数的实际代码。在 C 和 C++中,声明和定义的差别很重要。一般来说,声明放在头文件中,而定义放在源码文件中。

这样编译器就能找到 iostream 类的声明和了解遇到 cout 语句时要做什么了。

提 示 如果没有包括程序引用的类和函数的头文件,则会产生编译错误。编译错误会显示 Undefined symbol 'cout' 之类的文字。看到这个错误消息时,应立即检查是否包括了所有程序引用的类和函数的头文件。为了找到类和函数声明所在的头文件,单击类或函数名并按 F1,打开 Windows Help,显示光标所在项目的帮助主题。在帮助主题顶上,可以看到类和函数声明所在的头文件。

编写控制台 Hello World 程序之前还有一点需要说明。iostream 类包含几个特殊的操作器(manipulator),可以控制流的处理方式。目前要介绍的是 endL(行末)操作器,用于在输出流中插入新行。我们用 endL 在文本输出到屏幕上后插入新行。注意 endL 中最后一个字符是 L 而不是 1。

对 iostream 类有一些基本了解后,就可以编写控制台 Hello World 程序了。将程序编辑到如下清单 1.1 所示。我在每行前面放上了标识序号。C++并不用行号,所以输入语句时要跳过这些行号。

清单 1.1 hello.cpp

```
1: #include <condefs.h>
2: #include <iostream.h>                    // add this line
3: #pragma hdrstop
4:
```

```
5: int main(int argc,char * * argv)
6: {
7:   cout << "Hello World!" << endl;      add this line
8:   return 0;
9: }
```

说　明　C++中不考虑空白。通常，放上空格或新行符都一样。显然，关键字和变量名中间不能插入空格，别的地方则可以随意插入。例如，下列代码是等价的：

```
int main(int argc,char * * argv)
{
cout << "Hello World!";
return 0;
}
```

等于

```
int main(int argc,char * * argv){cout<<"Hello World!",return 0;}
```

显然，前者更可读，更让人喜欢，虽然编程风格各不相同，但如果采用本书所用的编程规则，你会在实际编程工作中受益无穷。

分　析　单击工具条上 Run 按钮，程序编译和运行。程序运行时，可以看到弹出一个 DOS 窗口，出现 Hello World! 字样。但程序稍纵即逝，因为 main()函数末尾终止程序并立即关闭了控制台窗口。为了解决这个问题，需要加上两行代码防止控制台窗口立即关闭。标准 C 语言库中有个 getch()函数，用于取得键盘上的键击。利用这个函数可以防止控制台窗口立即关闭。在编辑器窗口中将程序编辑成如下清单 1.2 所示，说明语句可加可不加。记住要跳过行号。

清单 1.2　Hello.cpp(修订版)

```
1: #include <condefs.h>
2: #include <iostream.h>
3: #include <conio.h>                    // add this line
4: #pragma hdrstop
5:
6: int main(int argc,char * * argv)
7: {
8:     cout << "Hello World!" << endl;
9:     // add the following two lines
10:    cout << endl << "Press any key to continue...";
11:    getch();
12:    return 0;
13: }
```

分　析　这时运行应用程序，出现 Hello World! 字样并保持在屏幕上，为了终止程序和关闭控制台窗口，可以按键盘上任意键。

说　明　读者请注意保存清单 1.1 中的程序时 C++ Builder 如何保存控制台应用程序。保存项目时，C++ Builder 生成两个文件，项目文件扩展名为.BPR，包含 C++

Builder 建立该项目所需的信息：源文件扩展名为.CPP，包含实际程序代码。例如，如果将项目存为 Hello，则硬盘上有两个文件：Hello.bpr 和 Hello.cpp。为了打开项目，要打开 Hello.bpr，但代码编辑器窗口中显示的是 Hello.cpp。这初看起来有点乱，但不久读者就会明白其中的原因。第 6 课和第 10 课"再谈项目"中会详细介绍项目。

所列的程序也放在 http://www.mcp.com/sams/codecenter.html 中，这个例子要安装到硬盘上之后才能编译。尽管最初最好手工输入短程序，但今后较长的程序可以安装到硬盘上，避免输入错误和由此产生的编译错误。

Hello World 之二也没什么也不起，但下面几页介绍 C++ 语言时会用到控制台应用程序，所以要了解如何生成和运行控制台应用程序。下面转入 C++ 语言基础的介绍。

1.4 C++语言基础

C++是个强大的语言，可以用于做别的语言做不了的工作。但是，这种强大功能是有代价的。开始使用C++时，你可能会遇到内存溢出和访问失效等问题，使程序死机。这里用最简短的篇幅介绍C++语言基础。C++语言本身有专著介绍，这种书还特别厚，所以别指望我能用三言两语说清楚。读者学完本书并使用C++ Builder一般时间之后，最后对C++语言再作更深入的了解。

C++可以最充分地利用面向对象编程(OOP)的优势。OOP 不只是一个新名词，而有它的实际意义，可以生成可复用的对象。

新术语 对象(object)，和前面介绍的构件一样，是完成特定编程任务的软件块(构件是对象，但对象不全是构件，稍后会解释这点)。

对象只向用户(使用对象的编程人员)显示必须的部分，从而简化对象的使用。用户不必知道的所有内部机制都隐藏在幕后。这一切都包括在面向对象编程的概念中。OOP 可以用模块化方法进行编程，从而避免每次从头开始。C++ Builder 程序是面向 OOP 的，因为 C++ Builder 大量使用构件。生成构件后(你生成的或C++ Builder 内置的构件)，就可以在任何C++ Builder 程序中重复使用。构件还可以扩展，通过继承生成具有新功能的新构件。最妙的是，构件隐藏了所有内容细节，使编程人员能集中精力充分利用构件。第 4 课"C++类与面向对象编程"中会详细介绍对象和C++类。

1.4.1 入门简介

在C++之前先有C语言，C++是建立在C语言之上的，称为"带类的C语言"。这个C语言基础在当今的C++程序中仍然很重要。C++并不是取代C，而是补充和支持C。本章余下部分和下几章主要介绍C++中来源于C语言的部分。实际上，这里介绍的是C语言，第2课"C++基础"中才转入C++。读者不必关心哪个来自C，哪个来自C++，因为这些全在C++中。

C++语言很难按顺序介绍，因为我们要介绍的所有特性都是交叉的。我准备的一次介绍一块，然后拼凑起来。到第3课"高级C++"结束，你将对C++语言有个完整的了解。一下子没有掌握某个概念也没关系，有些概念必须经过实践才能完全了解。

1.4.2 变量

还是从变量讲起来吧。变量(variable)实际上是赋予内存地址的名称。声明变量后,就可以用它操作内存中的数据。下面举几个例子进行说明。下列码段用了两个变量,每条语句末尾用说明语句描述执行该语句时发生的情况:

```
int x;          // variable declared as an integer variable
x = 100;        // 'x' now contains the value 100
x +=50;         // 'x' now contains the value 150
int y = 150;    // 'y' declared and initialized to 150
x += y;         // 'x' now contains the value 300
x++;            // 'x' now contains the value 301
```

新术语 变量(variable)是留作存放某个数值的计算机内存地址。

注意 x 的值在变量操作时会改变,稍后会介绍操作变量的C++操作符。

警 告 声明而未初始化的变量包含随机值。由于变量所指向的内存还没有初始化,所以不知道该内存地址包含什么值。例如,下列代码

```
int k;
int y;
x=y+10; //oops!
```

本例中变量 y 没有事先初始化,所以 x 可能取得任何值。

例外的情况是全局变量和用 static 修饰声明的变量总是初始化为 0。而所有其它变量在初始化或赋值之前包含随机值。

变量名可以混合大写、小写字母和数字与下划线(_),但不能包含空格和其它特殊字符。变量名必须以字母或下划线开始。一般来说,变量名以下划线或双下划线开始不好。变量名允许的最大长度随编译器的不同而不同。如果变量名保持在 32 个字符以下,则绝对安全。实际中,任何超过 20 个字符的变量名都是不实用的。下例是有效变量名的例子:

```
int aVeryLongVariableName;    // a long variable name
int my_variable;              // a variable with an underscore
int _x;                       // OK,but not advised
int X;                        // uppercase variable name
int Label2;                   // a variable name containing a number
int GetItemsInContainer();    // thanks Pete!
```

说 明 C++中的变量名是考虑大小写的,下列变量是不同的:

```
int XPos;
int xpos;
```

如果你原先所用语言不考虑大小写(如 Pascal),则开始接触考虑大小写的语言可能不太适应。

1.4.3 C++数据类型

新术语 C++数据类型定义编译器在内存中存放信息的方式。

在有些编程语言中，可以向变量赋予任何数值类型。例如，下面是BASIC代码的例子：

```
x = -1;
x = 1000;
x = 3.14;
x = 457000;
```

在BASIC中，翻译器能考虑根据数字长度和类型分配空间。而在C++，则必须先声明变量类型再使用变量：

```
int x1 = -1;
int x = 1000;
float y = 3.14;
long z = 457000;
```

这样，编译器就可以进行类型检查，确保程序运行时一切顺利。数据类型使用不当会导致编译错误或警告，以便分析和纠正之后再运行。有些数据类型有带符号和无符号两种。带符号(signed)数据类型可以包含正数和负数，而无符号(unsigned)数据类型只能包含正数。表1.1列出了C++中的数据类型、所要内存量和可能的取值范围。

表1.1 C++数据类型(32位程序)

数据类型	字节数	取值范围
char	1	-128到126
unsigned char	1	0到255
short	2	-32,768到32,767
unsigned short	2	0到65,535
long	4	-2,147,483,648到2,147,483,648
unsigned long	4	0到4,294,967,295
int	4	同long
unsigned int	4	同unsigned long
float	4	1.2E-38同3.4E381
double	8	2.2E-308同1.8E3082
bool	1	true或false

从上表可以看出，int与long相同。那么，为什么C++还要区分这两种数据类型呢？实际上这是个遗留问题。在16位编程环境中，int要求2个字节而long要求4个字节。而在32位编程环境中，这两种数据都用4个字节存放。C++ Builder只生成32位程序，所以int与long相同。

说 明 在C++ Builder和BorLand C++ 5.0中，Bool是个真正的数据类型。有些C++编译器有Bool关键字，则Bool不是个真正的数据类型。有时Bool只是个typedef，使Bool等价于int。typedef实际上建立别名，使编译器在一个符号与另一符号间划上等号。typedef的语法如下：

```
typedef int Bool;
```

这就告诉编译器：Bool是int的别名。

说 明 只有double和float数据类型使用浮点数(带小数点的数)。其它数据类型只涉及

试读结束，需要全本PDF请购买 www.ertongbook.com