附录 B　安装 DirectX 和使用 Visual C/C++

B.1　安装 DirectX

附带光盘中最重要的、必须安装的部分是 DirectX SDK 及其运行阶段文件。安装程序位于 DIRECTX\目录中，该目录中还有一个 README.TXT 文件，阐明了最后的修改。

注意：必须安装了 DirectX 8.0 SDK 或更高版本才能使用本书的源代码。如果不能确定系统中是否已经安装了最新版本的 DirectX SDK，请运行安装程序进行确认。建议使用 DirectX 9.0，作者在编译本书的程序时，使用的都是该 SDK。

安装 DirectX 时，请注意安装程序将 SDK 文件拷贝到了哪个目录下，因为编译程序时，必须正确地指定.LIB 和.H 文件的搜索路径。

另外，当您安装 DirectX SDK 时，安装程序会询问是否安装 DirectX 运行阶段文件。像 SDK 一样，运行阶段文件也是运行程序所必不可少的。然而，运行库有两个版本：

- Retail——这是完整的零售用户版本，也是可以指望用户该拥有的版本。该版本的速度比调试版本快。如果需要，可以安装零售版本来覆盖调试版本。
- Debug——这个版本提供了用于调试的挂钩（hook），建议使用该版本进行开发，但 DirectX 程序的运行速度可能慢些。

注意：Borland 用户请注意，DirectX SDK 有 Borland 版本的 DirectX .LIB 导入库（Import Library），可以在 DirectX SDK 的安装目录下的 BORLAND\目录中找到，编译时必须使用这些文件。另外，请务必访问 Borland 公司网站，阅读 BORLAND\目录中的 README.TXT 文件，以获得关于使用 Borland 编译器编译 DirectX 程序的最新提示。

最后，微软公司在不断更新 DirectX，请务必不时地访问微软公司的 DirectX 网站，网址如下：

http://www.microsoft.com/directx/default.asp

B.2　使用 Visual C/C++编译器

最近几年，我收到的询问如何使用 C/C++编译器的电子邮件数不胜数。我不希望再收到关于编译器问题的电子邮件，除非血从屏幕中喷出来且计算机在用口语说话！这些问题都是编译器新手提出的。您不能指望不事先阅读手册，就能够使用像 C/C++编译器这般复杂的软件。在编译本书的程序前，认真阅读编译器用户手册吧。

首先概述一下本书使用编译器的情况：我使用 MS VC++ 6.0 编译本书的程序，所有程序都能够用该编译器编译。我估计 VC++ 5.0 也能够编译，但不是十分确定。如果您使用的是 Borland 编译器或 Watcom 编译器，也应该能够编译，但可能要做一些额外的工作来正确设置编译器。为避免令人头疼的编译器设置，

建议您购买一份 VC++，其售价为 99 美金。对于 Windows/DirectX 程序来说，微软公司的编译器是最好的，它令各方面配合得更好。在编译其他程序时，我曾使用过 Borland 编译器和 Watcom 编译器，但对于 Windows 应用程序，我认识的很多游戏编程专家没有不使用 MS VC++的。根据要干的工作选择合适的工具。

另外，也可以使用最新的 Microsoft C/C++ .NET 编译器。然而，请务必阅读编译器手册，因为与 VC++ 6.0 相比，设置和对话框可能稍有不同。

B.3 编 译 提 示

下面是一些设置 MS VC++编译器的提示，其他编译器也大致相同：

- 应用程序类型（Application Type）——DirectX 程序是 Windows 程序，准确地说，它们是 Win32.EXE 应用程序。因此，编译 DirectX 程序时，总是将编译器的编译类型设置为 Win32.EXE 应用程序。生成控制台（Console）程序时，应将编译类型设置为控制台应用程序。另外，建议您建立一个工作区（Workspace），在其中编译所有的程序。
- 搜索路径（Search Directories）——编译器需要两项信息才能正常编译 DirectX 程序：.LIB 文件和.H 文件。设置编译器（Complier）和链接器（Link）的搜索路径选项，使之在 DirectX SDK Library 目录和 Include 目录中搜索.LIB 和.H 文件，这样编译器在编译时将能够找到这些文件。然而，这还不够，还必须使 DirectX 路径位于搜索树的最前面。这是因为 VC++自带了一个旧的 DirectX 版本，如果不小心，链接的将会是 DirectX 3.0 文件。另外，务必在工程中手工包含 DirectX.LIB 文件，这意味着可以使用 DDRAW.LIB、DSOUND.LIB、DINPUT.LIB 和 DINPUT8.LIB 等。
- 错误级别设置（Error-Level Setting）——务必合理地设置编译器的错误级别，如设置为级别 1 或级别 2。不要将其关闭，但也不要设置成报告任何错误。本书的代码都是专家级 C/C++程序，但编译器可能认为有很多事情我该做而没有做，所以将告警级别调低些。
- 类型转换错误（Typecast Errors）——如果编译器指出某行代码有类型转换错误，做相应的类型转换就是了。我收到很多不知道类型转换为何物的读者来信，如果您也不知道，请参阅 C/C++书籍。必须承认，我的程序中或多或少会漏掉一些显式地类型转换，VC++6.0 错误信息有时候看起来来势汹汹，当您遇到此类错误时，请先查看编译器要求的是什么类型，然后将右值（rvalue）转换为这种类型，便可以修复错误了。
- 优化设置（Optimization Settings）——由于我们开发的不是发行版产品，因此不要将编译器设置成最佳优化级别，只需设置为标准级别即可，该级别更看重速度而不是目标文件的大小。在 VC++中，要设置优化级别，可选择菜单 Project/Settings，然后单击选项卡 C/C++，然后在类别 Optimizations 中选择。
- 线程模型（Threading Models）——本书 99％的范例都是单线程的，所以使用单线程库。如果您不知道线程的含义，请参考编译器方面的书籍。然而，如果需要使用多线程库，我将在书中指明。要编译使用多线程的程序，需要切换到多线程库。在 VC++中，要设置线程模型，可选择菜单 Project/Settings，然后单击选项卡 C/C++，然后在类别 Code Generation 中选择。
- 代码生成（Code Generation）——这个选项控制编译器生成的代码类型。设置该选项为 Pentium Pro，我好久没有见到过 486 的计算机了，因此没必要担心兼容性问题。在 VC++中，要设置目标处理器，可选择菜单 Project/Settings，然后单击选项卡 C/C++，然后在类别 Code Generation 中选择。
- 结构对齐（Struct Alignment）——这个选项控制如何对结构进行填充。PentiumX 处理器擅长处理 32 位整数倍的数据，所以将该选项设置为最大（VC++当前支持的最大值为 16 字节）。虽然生成的可执行代码可能稍微大些，但运行速度将快得多。在 VC++中，要设置目标处理器，可选择菜单 Project/Settings，然后单击选项卡 C/C++，然后在类别 Code Generation 中选择。
- 编译器更新——信不信由您，Microsoft 在不断更新一切产品，包括编译器。因此，请务必从 MSDN 下载并安装最新的 VC++（.NET）Service Pack，其网址为 http://msdn.microsoft.com/。
- 最后，编译程序时，务必包含主程序中引用的所有源文件。例如，如果主程序包含了头文件 T3DLIB1.H，必须在工程中包含源文件 T3DLIB1.CPP。