Microsoft Visual Studio.NET

精华版编程高手时尚丛书

Visual C++.NET

深入编程与实例剖析

陈坚 主编
陈鸿飞 孙志月 参编

西安电子科技大学出版社
http://www.xduph.com

精华版编程高手时尚丛书

Visual C++.NET 深入编程与实例剖析

陈坚　主编

陈鸿飞　孙志月　参编

西安电子科技大学出版社

2002

内 容 简 介

本书详尽介绍了使用 Visual C++.NET 设计 Windows 应用程序的各种技术，并通过丰富的范例帮助读者迅速掌握这一技术。

全书分为 10 章，主要介绍了 Windows 编程的一些较高级技术和应用，内容包括 Windows 通用控件，图形图像显示，内存分配，钩子、进程和线程，数据库技术，MODEM 通信技术，TCP / IP 和 Sockets 通信技术，以及 Internet 通信技术等。

本书主要面对中、高级读者，并结合了一些实际应用，目的是使读者对 Visual C++的了解更加深入、完整和系统化。熟悉 Microsoft 公司的 Windows SDK、Visual Basic 或 Borland 公司的 Delphi、C++ Builder 编程的程序员都能够从本书中找到感兴趣、有用的章节。

图书在版编目（CIP）数据

Visual C++.NET 深入编程与实例剖析 / 陈坚主编.

西安：西安电子科技大学出版社，2002.1

（精华版编程高手时尚丛书）

ISBN 7-5606-1102-8

Ⅰ.V… Ⅱ.陈… Ⅲ.C 语言—程序设计 Ⅳ.TP312

中国版本图书馆 CIP 数据核字（2001）第 097646 号

策划编辑 毛红兵 李惠萍

责任编辑 龙 晖

出版发行 西安电子科技大学出版社（西安市太白南路 2 号）

电 话 (029)8227828 邮 编 710071

http: // www.xduph.com E-mail: xdupfxb@pub.xaonline.com

经 销 新华书店

印 刷 西安兰翔印刷厂

版 次 2002 年 1 月第 1 版 2002 年 1 月第 1 次印刷

开 本 787 毫米×1092 毫米 1/16 印张 20.5

字 数 485 千字

印 数 1～4000 册

定 价 26.00 元

ISBN 7-5606-1102-8/TP・0553

XDUP 1373001-1

*** 如有印装问题可调换 ***

前　言

Visual Studio.NET（又称为 Visual Studio 7.0）是微软继 Visual Studio 6.0 后推出的又一代可视化开发工具。所谓 .NET，就是指 .NET 框架（.NET Framework）。它是一种用于构建、配置和运行 Web 服务和应用程序的多语言环境。

Visual C++.NET 是 Visual Studio.NET 的重要组成部分之一，是 Microsoft 公司推出的目前使用极为广泛的基于 Windows 平台的可视化编程环境。相对于 Visual C++ 6.0，Visual C++.NET 在 ATL、DCOM、MFC、数据库等方面功能有很大增强，尤其在开发环境界面上变化很大，采用了全新的平面化操作界面，具有很强的亲和 性。

“Visual”的意思是“可视的”，Visual C++就是可视化编程语言。利用 Visual C++可以简化程序员的编程，提高工作效率。

Visual C++贯穿了面向对象的程序设计思想，其核心是 Microsoft 基本类库（Microsoft Foundation Classes，缩写为MFC），称之为“应用程序框架”。它一方面用类封装了 Windows API，另一方面使用称为“消息映射”的机制把 Windows 消息和命令传递到窗口、文档、视以及 MFC 应用程序中的其它对象。因而 MFC 成功地将面向对象和事件驱动编程概念联系起来，并得到很好的配合。使用 MFC 编写 Windows 应用程序简单方便，代码量小。

本书分为 10 章。在此将全书内容简要介绍如下：

第一章简单介绍了 MFC 应用程序结构、编程工具以及 MFC 类的层次结构。

第二章介绍了 Windows 通用控件特性与编程。

第三章详细介绍了作为多媒体技术核心的图形图像显示处理技术。

第四章首先介绍了 Windows 线性内存体系结构，然后具体介绍了各种内存分配技术，最后讲解了 MFC 内存诊断技术。

第五章首先介绍了用于监视系统中各种消息的钩子和钩子过程，然后介绍了 Windows 实现抢先多任务机制的关键技术——线程，最后介绍了多线程的同步技术。

第六章首先介绍了文档模板、切分窗口和“画中画”等另外一些深层次的内容，然后通过一个完整的综合实例来帮助读者更好地掌握前面各章介绍的编程技术。

第七章详细介绍了 ODBC 等数据库编程技术。

第八章介绍了使用越来越广泛的 MODEM 通信技术。

第九章介绍了 Intranet 和 Internet 网络的核心技术——TCP / IP，以及 Sockets 通信技术。

第十章详细介绍了 Internet 通信技术，包括对 HTTP、FTP 等服务器的交互技术。

为了帮助读者迅速掌握所介绍的内容，本书给出了丰富的范例程序。所有程序全部经严格测试，无需调试。读者完全可以即拿即用，从而节省了大量的时间，提高了工作效率。另外，有关 Visual C++编程技术以及 Internet 编程技术介绍还可参见由西安电子科技大学出版社出版的《Windows 95 多媒体应用程序设计技术》和《Visual InterDev 6.0 技术内幕》两书。

2002/04

从本书的酝酿到编写，自始至终都得到西安电子科技大学出版社李惠萍老师的关心和指导，在此表示衷心的感谢。

由于作者学识水平有限和时间仓促，书中难免有不足和错误之处，恳请各位同仁及广大读者给予批评指正。欢迎将对本书的任何指教或宝贵意见通过 E-mail 发送给作者，以便作者在下版中改进。

作者的 E-mail 地址是 chinj@163.net，网址是 http://chinj.yeah.net。

作　者

2001 年 11 月

目　录

第一章

Visual C++和基本类库

Microsoft推出的Visual C++和Microsoft基本类库(Microsoft Foundation Classes，缩写为MFC)成功地将面向对象和事件驱动编程概念联系起来，并得到很好的配合，使得编写Windows应用程序的过程变得简单、方便，且代码量小。本章首先概述MFC应用程序结构等一些基础知识。

本章主要内容：

- MFC框架、文档和视图结构
- AppWizard、AppStudio和ClassWizard
- 调试技术
- MFC类的层次结构

1.1 MFC 框架、文档和视图结构

MFC 实现了两项功能：一是定义了一个应用程序的初始行为，二是对 Windows API 进行类封装。MFC 类不仅封装处理 Windows API 的所有细节，还简化了 OLE(对象链接与嵌入)和 ODBC(开放式数据库互连)等处理。

MFC 既支持单文档接口(SDI)应用程序，还支持多文档接口(MDI)应用程序。SDI 应用程序(如 NOTEPAD 记事本)一次只允许打开一个文档框架窗口，而 MDI 应用程序允许在同一个应用实例中打开一个或多个文档框架窗口。MFC 支持 MDI 应用程序的“Last File Used(最近的文件)”列表。另外 MFC 对工具栏的泊位或漂浮、联机帮助也提供自动支持。

一个 SDI 或 MDI 程序主要由框架(Frame)、文档(Document)和视图(View)构成。单文档窗口和 MDI 各子窗口都是框架窗口，它的核心是“文档—视图”结构。用户处理的数据单位就是一个文档对象。文档对象用于维护、加载及保存数据。它是由“文件”菜单中的“新建”或“打开...”命令创建的，一般都保存在一个文件中。用户通过文档之上的视图与文档进行交互。视图是镶嵌在框架窗口客户区域内的一个窗口，它显示其关联的文档的数据，接收鼠标与键盘输入并把它翻译为选择与编辑等操作。另外通过用户接口对象，如菜单、按钮，将命令发送给文档、视图及应用程序中的其它对象，然后由这些对象完成命令的执行。

具体而言，我们可以认为应用程序主要由以下几个对象构成。

1) 应用程序对象

应用程序类(从 CWinApp 派生而来)完成应用程序的初始化和清除工作。有且只有一个应用程序对象，它创建和管理所有文档类型的文档模板。

2) 线程对象

应用程序类是应用程序的主执行线程(称为主进程)。其它独立的执行线程(称为辅进程)需从 CWinThread 派生。CWinApp 的基类也是 CWinThread 类。

3) 文档模板

文档模板定义了文档类、视图类和框架窗口类三者之间的关系。CSingleDocTemplate 类实现 SDI 文档模板，CMultiDocTemplate 类实现 MDI 文档模板。

4) 文档

文档类(从 CDocument 派生而来)指定了应用程序的数据。若应用程序支持 OLE 功能，则文档类可从 COleDocument 或其派生类中派生。

5) 视图

视图类(从 CView 派生而来)是用户的“数据窗口”。视图类指定了用户查看文档数据的方式和交互方式，如见到的文档可能是文本，也可能是图表或图像等。通常情况下，一个文档只有一个视图，但在有些情况下一个文档可能要有多个视图。

MFC 已定义了许多视图类，应用程序可以根据不同需要从不同的视图类派生。如果需要滚动，则视图类可从 CScrollView 派生。如果视图中有一个基于对话框资源的用户接口，则视图类可从 CFormView 派生。对于简单的文本数据，可从 CEditView 派生。对于基于格式的数据访问应用程序，可从 CRecordView 派生。

6) 框架窗口

视图是显示在“文档框架窗口”中的。在 SDI 应用程序中，文档框架窗口也是该应用程序的“主框架窗口”。在 MDI 应用程序中，文档窗口是显示在主框架窗口中的子窗口，所派生的主框架窗口类指定了包含视图的框架窗口的格式及其它特性。

SDI 应用程序的文档框架窗口类的基类为 CFrameWnd 类。MDI 应用程序的主框架窗口类的基类为 CMDIFrameWnd 类，文档框架窗口类的基类为 CMDIChildWnd 类。

总之，应用程序的这些对象通过命令和消息相互联系，共同对用户动作进行响应。一个应用程序对象管理一个或多个文档模板；每个文档模板创建并管理一个或多个文档；用户通过包含在框架窗口中的一个视图观察和处理文档。

1.2 MFC 编程工具及其相互关系

Visual C++的 MFC 应用框架将编辑器、编译器、连接器、调试器、AppWizard、AppStudio、ClassWizard 等工具集成在同一环境中，极大地方便了程序员开发 Windows 应用程序。其中 AppWizard、AppStudio、ClassWizard 三个工具尤为重要。

1.2.1 AppWizard

AppWizard 又称为 MFC 应用程序向导，是一个代码发生器，用于创建 MFC 应用程序。它按照用户通过对话框指定的特性、类名及源代码文件名来创建一个可启动的 Windows 应用程序框架。在此基础上程序员可完成自己应用程序的特殊功能。生成的应用程序项目文件包含应用程序、文档、视图和框架窗口等类实现文件，包含标准菜单和可选择工具条的资源文件，其它必需的 Windows 文件，以及可选的含有 Windows Help 主题的 .rtf 文件或 .htm 文件等。AppWizard 还可以包含对 OLE 和数据库的支持。

在开发应用程序的进程中，必须通过修改菜单资源，加入处理函数等，使程序完全有血有肉。“// TODO”注释表示在此处需要程序员加入实际操作代码。

1.2.2 AppStudio

AppStudio 是一个资源编辑器，用于创建用户界面和编辑程序资源，可用来创建包含 #define 常量在内的头文件。具体地讲，AppStudio 可以创建并编辑菜单、对话框、定制的控件框、加速键、位图、图标、光标、字符串和版本等资源。它对应于 Studio Workspace 的“资源视图”，如图 1.1 所示。

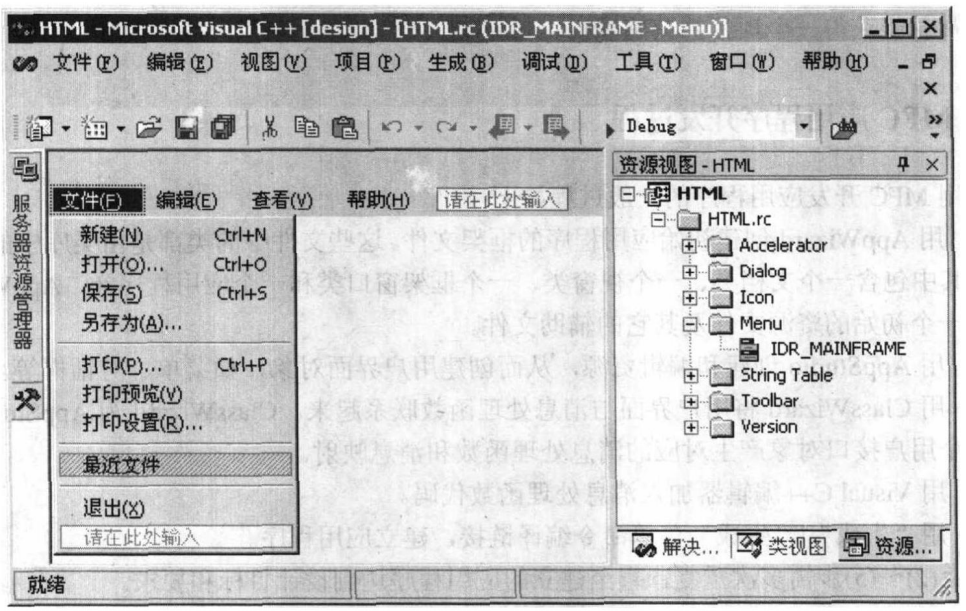

图 1.1 Visual C++ 工作室

通常该资源编辑器与 ClassWizard 一同使用，例如，创建了对话模板资源后，就可以利用 ClassWizard 创建新的对话框类，然后可以映射对话框中各命令按钮的消息处理函数以及增加成员变量。

AppWizard 建立含有基本资源(如标准菜单、About ...对话框、含有预定义字符串的字符串表等)的 .rc 文件。其它文件包含缺省图标，版本资源。

在编程时常常需要建立新的资源和修改原有资源。对于菜单、工具条按钮和对话框等资源，在创建之后可以使用 ClassWizard 将它们链接到代码中。

1.2.3 ClassWizard

在编写程序过程中，最频繁使用的 MFC 工具就是 ClassWizard(MFC 类向导)。程序员利用 ClassWizard 填写实施细节，选择把哪些消息映射给哪些对象，然后实现这些映射。即利用它管理类和 Windows 消息，将 Windows 消息及用户界面对象(例如菜单)与程序代码联系起来。

利用它可以向 AppWizard 所产生的工作框架中添加新的类或函数，也可以对现存的类进行修改，这就避免了一般代码发生器经常遇到的一致性问题。

另外，还可以创建类成员变量、OLE Automation 的方法和属性、数据库类和覆盖 MFC 类中的虚函数等，以实现应用程序工作的细节。

Visual Workspace 提供的类视图将源代码组织为类，使得代码的查找、编辑更为容易。通过利用 ClassView，程序员可以将精力集中于类上，通过类名和成员函数进行工作，而不需要操作源文件和头文件列表。类视图是可缩放项目窗口的一个弹出对话框，用来显示项目的类和组成部件，使得程序员不必再手工打开它们。利用一个类似 Explorer 的界面，可以跳到、增加或移去类、成员函数和成员变量。由于类视图不必通过建立而自动修改，因而程序员总能获得一个项目的精确视图。

1.2.4 MFC 应用程序开发过程

使用 MFC 开发应用程序的一般过程如下：

(1) 用 AppWizard 创建初始应用程序的框架文件。这些文件中的类都是由类库中的类派生的，其中包含一个文档类、一个视窗类、一个框架窗口类和一个应用程序类。AppWizard 还创建一个初始的资源文件及其它的辅助文件。

(2) 用 AppStudio 创建和编辑资源，从而创建用户界面对象，如菜单、对话框等。

(3) 用 ClassWizard 将用户界面与消息处理函数联系起来。ClassWizard 为 AppStudio 产生的每个用户接口对象产生对应的消息处理函数和消息映射。

(4) 用 Visual C++编辑器加入消息处理函数代码。

(5) 用“生成”|“生成”菜单命令编译链接，建立应用程序。

以上(2)～(5)步需多次重复，直至建立的应用程序达到设计目标和要求。

1.3 MFC 应用程序调试技术

除非特别小的应用程序，否则程序不可能一次就建立成功且符合要求。在设计过程中，程序发生错误是不可避免的，且程序越大越复杂，错误发生的可能性也就越大。通常可分

为两种错误：语法错误和执行错误。程序员可以很容易发现并改正程序语法错误，但不易发现和改正执行错误。因此本节主要讨论如何调试应用程序执行中的错误。Visual C++和MFC 提供了多种方法供程序员调试应用程序。

1.3.1 Visual C++内置的调试器

与DOS 程序调试器不同，不能使用调试器来从头到尾跟踪Windows 程序。因为Windows 程序是消息驱动的，而非顺序驱动。我们只能在需要跟踪的消息处理函数的入口设置断点。

Visual C++内置调试器与 Visual C++工作平台紧密配合，可以设置断点，控制单步执行；查看变量，修改变量，查看寄存器；通过 Quick Watch 快速查看变量值。菜单命令“调试”|“逐语句”可跟踪进入类库，“调试”|“逐过程”可以避免进入类库中的代码。

只有将程序设置为 DEBUG 模式(含有调试信息)，才能进行调试。

1.3.2 TRACE 宏

TRACE 宏为程序员提供与 DOS 环境下 printf 函数类似的功能，它将格式化字符串输出到工作台的调试窗口(如图 1.2 所示)。TRACE 宏是在程序执行时跟踪变量的值的简便的方法。

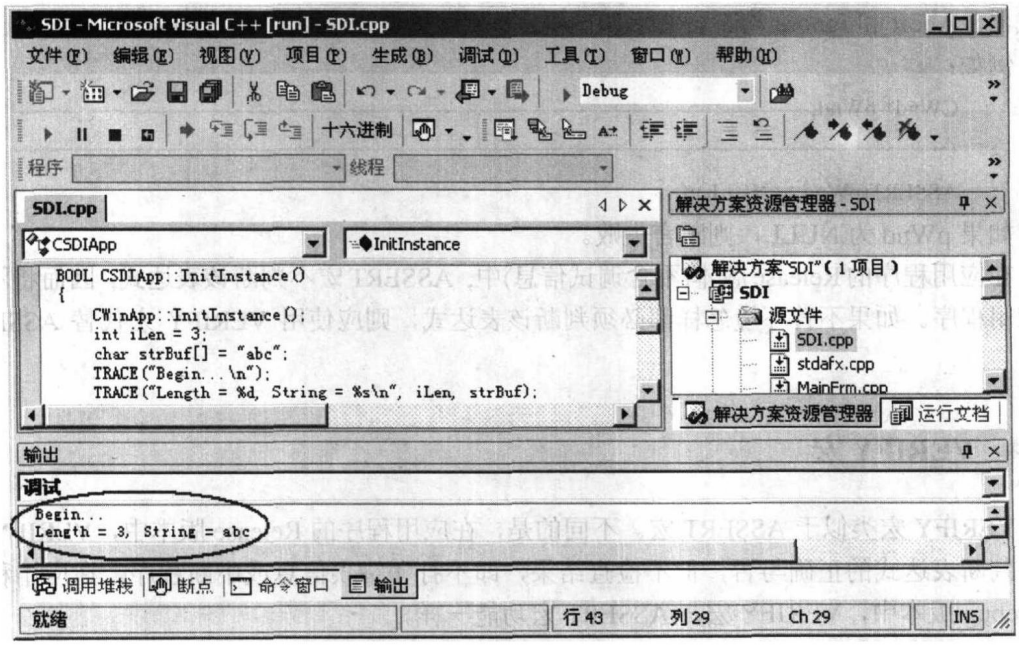

图 1.2 Visual C++调试窗口和 TRACE 输出

TRACE 宏用法如下：

TRACE(*exp*)

其中 *exp* 指定可变数目的参数。这些参数与 printf 函数中使用的可变数目的参数的用法相同。例如：

```
int iLen = 3;
char strBuf[] = "abc";
TRACE("Length = %d, String = %s\n", iLen, strBuf);
```

执行后，在“输出”窗口中输出：

```
Length = 3, String = abc
```

另外，类似功能的宏还有 TRACE0、TRACE1、TRACE2 和 TRACE3 等。

1.3.3 ASSERT 宏

ASSERT 宏为断言测试，用于检查函数做的假设。它在遇到错误条件时会引起程序终止。

通过设置适当的 ASSERT 参数，使它只在程序按预想方式工作时才为正确值。该宏指令判断 ASSERT 参数，如果参数表达式为假(0)，则通知用户并终止程序执行，反之若参数为真(非 0)，则不采取行动。

若 ASSERT 失败，则显示一对话框，消息框中显示如下信息：

```
assertion failed in file <name> in line <num>
Abort Retry Ignore
```

其中，<name>为源文件名，<num>为出错断言的行号。如果选择 Abort，则程序终止。如果选择 Ignore，则程序忽略该信息，继续执行。如果选择 Retry，则有可能在 ASSERT 后进入调试器。Abort 和 Ignore 均不会激活调试器，因此它们无法提供查看栈的方法。

例如：

```
CWnd* pWnd;
...
ASSERT(pWnd != NULL);
```

如果 pWnd 为 NULL，则断言失败。

在应用程序的 Release 版本(不含调试信息)中，ASSERT 宏不判断该表达式，因而将不中断应用程序。如果不管环境怎样都必须判断该表达式，则应使用 VERIFY 宏代替 ASSERT 宏。

1.3.4 VERIFY 宏

VERIFY 宏类似于 ASSERT 宏。不同的是，在应用程序的 Release 版本中，VERIFY 宏仍要判断表达式的正确与否，但不检查结果，即不打印错误信息或中断程序。在应用程序的 Debug 版本中，VERIFY 宏与 ASSERT 宏功能一样。

VERIFY 宏用法如下：

```
VERIFY(booleanException)
```

其中参数 *booleanException* 指定其值将被判断为非 0 或 0 的一个表达式(包括指针值)。

在 Debug 版本中，VERIFY 宏判断其参数的值。如果为 0，则 VERIFY 宏显示一消息框，显示诊断消息 assertion failed in file <name> in line <num>，并终止程序。若条件为非 0，则该宏不采取任何动作。其中<name>为源文件名，<num>为源文件中断言出错的行号。

1.3.5 消息框

无论 Debug 版本，还是 Release 版本，都可以利用 AfxMessageBox 或 MessageBox 在需要输出信息的地方显示消息框，与用户交互。在消息框中可以显示各种值，适用于简单程序调试。

1.4 MFC 类库层次结构

为了使读者对于 Visual C++类库有一个整体了解，本节简要介绍 Microsoft 基本类库的层次体系(如图 1.3 所示)。具体而言，MFC 可以分为 10 大类，分别管理 Windows 应用程序的不同方面:

(1) 基类：Cobject。
(2) 应用体系结构类。
(3) 窗口、对话框和控件类。
(4) 绘图打印类。
(5) 简单数据类型类。
(6) 集合类。
(7) 文件和数据库类。
(8) Internet 和网络类。
(9) OLE 类。
(10) 调试和异常类。

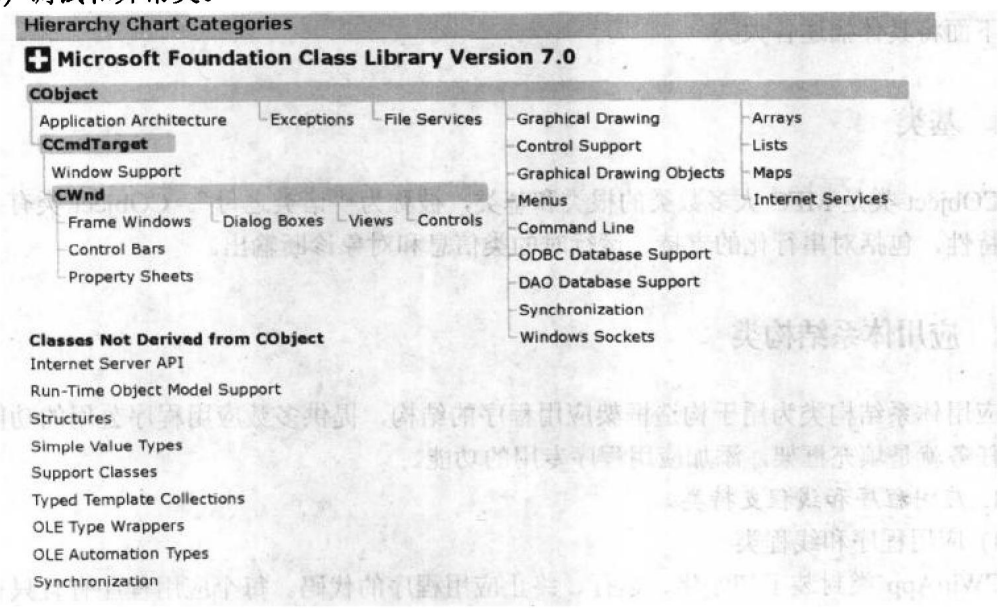

图 1.3 MFC 的分类

图 1.4 给出了 MFC 的部分类的继承关系，全图如同一根茂密的大树，读者具体可参见相应的 MSDN Library。

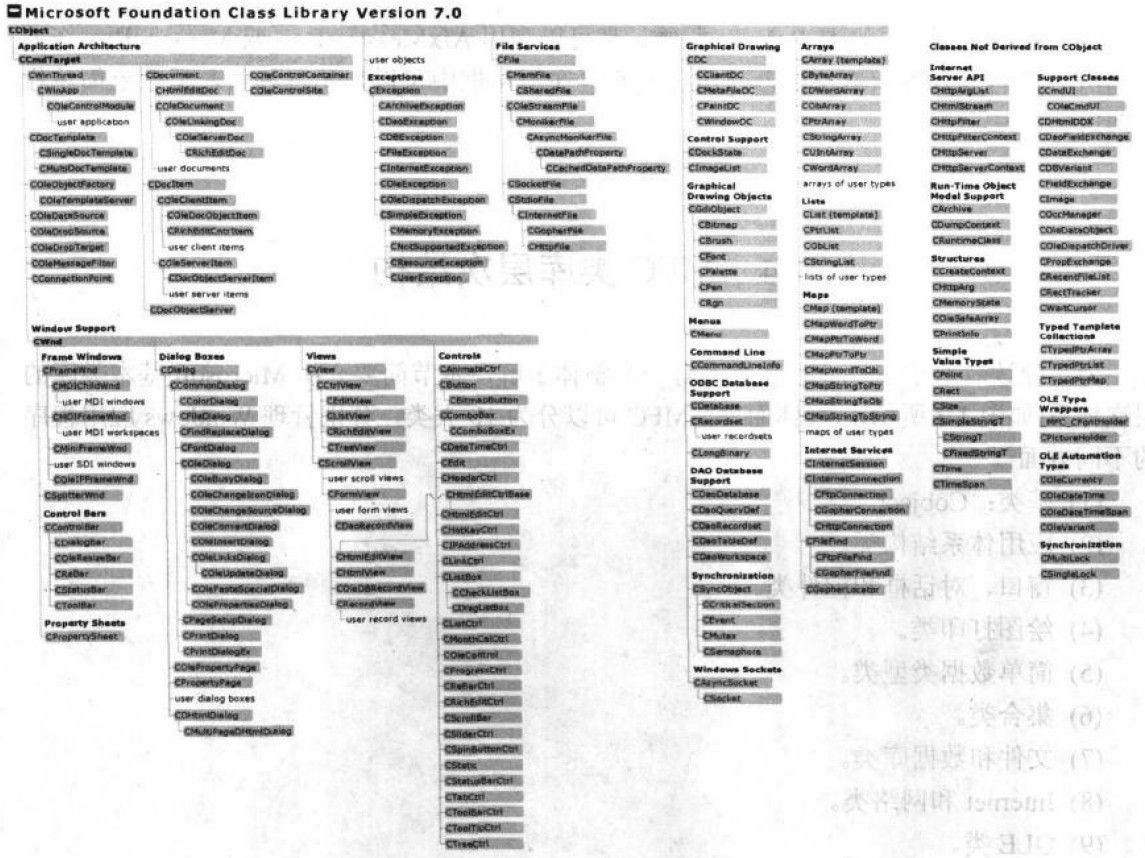

图 1.4　MFC 继承层次结构

下面将具体描述各大类。

1.4.1　基类

CObject 类是 MFC 大多数类的根类和基类，被称为“诸类之母”。CObject 类有很多有用的特性，包括对串行化的支持、运行时的类信息和对象诊断输出。

1.4.2　应用体系结构类

应用体系结构类为用于构造框架应用程序的结构，提供多数应用程序公用的功能。编程的任务就是填充框架，添加应用程序专用的功能。

1. 应用程序和线程支持类

1) 应用程序和线程类

CWinApp 类封装了初始化、运行、终止应用程序的代码。每个应用程序有且只有一个应用程序对象。