21世纪重点大学规划教材

任 哲 等编著

Windows程序设计

技术基础——MFC与.NET

机械工业出版社
CHINA MACHINE PRESS

配电子教案

21 世纪重点大学规划教材

Windows 程序设计技术基础

——MFC 与.NET

任 哲 等编著

机械工业出版社

本书是一部综合介绍 Windows 程序设计的高等院校教材。本书在读者学习了 C/C++、操作系统、数据结构等相关课程的基础之上，从 Windows Win32 程序框架及其设计出发，以 Windows 的 MFC 和.NET 为主线，重点介绍了蕴含在这些框架中的软件工程思想和方法，同时也以精要的方式介绍了 C#语言与 C/C++的重要区别。

通过学习本书，读者能够基本掌握现代软件工程的一些核心思想及方法。

本书适合作为普通高等院校计算机或相关专业的教材，也可作为相关领域培训机构的教学及参考用书。

图书在版编目（CIP）数据

Windows 程序设计技术基础——MFC 与.NET / 任哲等编著. —北京：机械工业出版社，2009.6

（21 世纪重点大学规划教材）

ISBN 978-7-111-26921-2

Ⅰ. W… Ⅱ. 任… Ⅲ. 窗口软件，Windows-程序设计-高等学校-教材

Ⅳ. TP316.7

中国版本图书馆 CIP 数据核字（2009）第 065045 号

机械工业出版社（北京市百万庄大街 22 号 邮政编码 100037）

策划编辑：唐德凯

责任编辑：唐德凯

责任印制：乔 宇

北京诚信伟业印刷有限公司印刷

2009 年 6 月 • 第 1 版第 1 次印刷

184mm×260mm • 25.5 印张 • 633 千字

0001— 3000 册

标准书号：ISBN 978-7-111-26921-2

定价： 42.00元

凡购本书，如有缺页，倒页，脱页，由本社发行部调换

销售服务热线电话：（010）68326294 68993821

购书热线电话：（010）88379639 88379641 88379643

编辑热线电话：（010）88379753 88379739

出 版 说 明

“211 工程”是“重点大学和重点学科建设项目”的简称，是国家“九五”期间唯一的教育重点项目。

进入“211 工程”的 100 所学校拥有全国 32%的在校本科生、69%的硕士、84%的博士生，以及 87%的有博士学位的教师；覆盖了全国 96%的国家重点实验室和 85%的国家重点学科。相对而言，这批学校中的教授、教师有着深厚的专业知识和丰富的教学经验，其中不少教师对我国高等院校的教材建设做过很多重要的工作。为了有效地利用“211 工程”这一丰富资源，实现以重点建设推动整体发展的战略构想，机械工业出版社推出了“21 世纪重点大学规划教材”。

本套教材以重点大学、重点学科的精品教材建设为主要任务，组织知名教授、教师进行编写，教材适用于高等院校计算机及其相关专业，选题涉及公共基础课、硬件、软件、网络技术等，内容紧密贴合高等院校相关学科的课程设置和培养目标，注重教材的科学性、实用性、通用性，在同类教材中具有一定的先进性和权威性。

为了体现建设“立体化”精品教材的宗旨，本套教材为主干课程配置了电子教案、学习指导、习题解答、课程设计、毕业设计指导等内容。

机械工业出版社

前　言

近年来，随着我国计算机技术及应用水平的飞速发展和提高，对于人才的需求不仅越来越旺盛，而且要求也越来越高。特别是对那些掌握了现代软件工程思想和技术，具有一定系统设计能力的人才的需求显得更为迫切。显然，这也就是在向高等院校提出要求：计算机专业学生必须在学习期间得到相应的系统设计能力的培养和训练。为此，各高等院校在计算机课程体系及课程内容的重构和改革中，对于集中地反映了现代软件工程思想和技术的两门课程——Java 和 Windows，都投入了极大的精力。

相对于侧重网络应用的 Java，Windows 课程内容的改革显得更为重要，也更困难。说它重要，是因为作为微型计算机上的第一个图形界面和事件驱动系统，Windows 本身就是软件工程发展的重要成果，它几乎涵盖了微型计算机应用技术的所有方面，所以它是学习软件工程思想和技术的极好素材，是高等院校计算机专业的必修课程；说它困难，是说这门课的改革困难，因为 Windows 经历了太多的历史发展阶段，它所涉及以及由它所产生的技术实在太多、也太繁杂，从而与有限的课时形成了尖锐冲突。于是，如何按照培养目标，准确、合理地对 Windows 技术进行精选，提炼其核心思想，结合应用需要，使学生能在有限的学习时间内理解并掌握现代软件工程思想基础，从而形成较强的系统设计和应用能力，就成了该课程改革的难点和重点。

为解决上述问题，作者与同行及业界朋友进行了广泛、深入的探讨，大体上形成了以下两点基本认识：

1．以 Windows 应用技术为载体，重点介绍现代软件工程设计思想

由于 Windows 本身涉及了软件工程的各个方面，特别是它为用户所提供的 MFC 和.NET Framework，既是应用所需要的程序框架，又集中体现了现代软件工程思想的发展和应用，所以它们是培养学生系统架构能力的极好素材。因此，Windows 教学应该坚持以 Windows 应用技术（即 MFC 和.NET Framework 中的技术）为载体，以介绍现代软件工程设计思想和技术为目的的原则。

具体地说，就是面对 Windows 如此之多的技术和极其有限的教学时数，整个的教学内容应该重在说理，而不能就技术谈技术，更不能成为某种开发工具的使用讲座。目的是使学生能够尽快地把握现代程序设计思想及其发展方向，为学生以后的发展奠定坚实的基础，从而不会使学生永远处在软件产业链的最底层，成为“代码工厂”的“打工者”。

2．注重应用，并在应用中培养学生的自学、质疑和创新能力

大家都知道，高等教育的责任是启发学生心智，培养学生质疑和创新的能力。但在毕业生就业压力极大的今天，如何使毕业生具有某种“实用”性也是当今中国高等教育所面临的一个严峻课题。也正因为这个原因，目前有相当一部分高等院校开设了一些所谓的“实用”课程。但什么叫“实用”，怎么才是“实用”，却是一个需要极其谨慎对待的问题。作者认为，凡是真正“实用”的人才，他的“应用”能力一定相当强。换句话说，“实用”仅是人才的外在表现，而“应用”能力才是人才的素质，而应用能力是需要理论基础的。所以，作者认为，

符合应用需要的理论必须是普通高等教育的重点，在理论教学活动中积极鼓励学生质疑现存理论，让学生在质疑和验证的过程中，通过大量的自学、讨论等方式来提高理论应用能力，从而最终达到企业所要求的“实用”性。

综合上述考虑，作为教材或教学参考书，本书按照历史发展顺序，分为了基于 C 的 Win32、基于 C++的 MFC 和基于 C#的.NET 三个部分。

本书使用了两章的篇幅介绍了 Win32。重点介绍了 Windows 的起源及其应用程序结构，图形界面和事件驱动，微软早期的“组件”和可变代码分离的软件工程思想等基本概念，最后以代码重构的方式引出了消息映射概念。这部分内容之所以篇幅比较小，一是学生这时应该具有了较好的 C 语言基础，只要了解了 Windows 程序结构，那么那些函数形式 Win32 API 完全可以由学生自学；二是现在也没有多少人直接使用 Win32。

MFC 部分所占篇幅较大。因为作为先修课的 C++，一直存在着难于在课程内引入大型实例的问题，从而影响了学生对面向对象程序设计思想的理解，而 Windows 的 MFC 恰恰可以弥补这个问题，从而使学生既学习了 MFC 又进一步理解了 C++，可以说是一举两得。再者，作为微软的第一个面向对象的应用程序框架，MFC 是微软所有后续技术（甚至 Sun 的 Java）的基础，它们的核心思想和技术几乎都出自于 MFC，只不过借助于完全面向对象技术对 MFC 进行了更为合理的封装和重构。

在对 MFC 的介绍中，为使学生可以平稳地从 Win32 过渡到 MFC。首先用 C++对 C Windows 程序进行了模拟类封装，进而引出了 MFC 程序框架；其后，重点介绍了极富面向对象特色的文档/视图结构（观察者模式的雏形），对后续技术影响重大的类信息表（即后来成为 C#反射机制基础的元数据和清单）、对象动态创建（工厂模式的雏形）、对象序列化等内容；另外，为了和后续的 NET 衔接，还介绍了 Windows 的动态链接库、COM 组件。

对于 NET 部分，本书也给予了较大的篇幅。一方面是因它极可能是学生毕业后马上要使用的技术；另一方面它为网络应用发展了跨语言和跨平台、分布式组件（程序集）、反射机制等技术。特别是它还反映了当代程序设计语言与编译器和虚拟平台的强烈相关性。也就是说，如果不对编译器和运行平台具有一定程度的了解，就没办法学习和正确使用该语言。

为了培养学生的自学能力和减少课时，对于微软新发展的 C#语言，本书采用了讲授和自学相结合的方式。其中的语言基础部分，只以精要的方式介绍了它相对于 C++的重要发展（例如，接口、抽象类、委托、属性、事件等），把其他内容留给了学生自学，重点介绍.NET 所发展的程序集、反射、特性等一些新技术。

由于对于本书内容的组织存在的难度较大，在编写过程中也存在着一些争论，作为参考，作者也借本书出版的机会谈一谈这些争论和自己的观点。

1．为什么要学那个连微软自己都声称“过时”了的 MFC

这是自作者写了《MFC Windows 应用程序设计》一书之后，一直有人在提出的问题。

确实，在推出了.NET 之后，微软似乎不再鼓励用户使用 MFC，用户在.NET 的 Visual Studio 中打开由 VC++制作的 MFC 工程时，它会通知用户“该工程已过时”。但这是通知你在 VC++上制作 MFC 工程已过时，而不是你在 Visual Studio 制作的 MFC 工程已过时。再者说，为什么要在适合网络应用开发的.NET 上来开发使用于桌面系统的 MFC 工程呢？

诚然，鉴于当初面向对象思想的不成熟及 C++语言的限制，MFC 作为一个面向对象的应用程序框架，它在某些方面确实有一些瑕疵。但也正因为此，MFC 才暴露了微软的一些核心

思想和技术，而这些东西并没有过时。想想看，微软会置自己多年的技术积淀而不顾，去另搞一个什么全新的东西吗？不会，决不会！阅读了本书之后，读者就会知道，从某种意义上来说，.NET 就是 MFC 的重构，只不过它把一些核心技术封装到了运行平台和编译器，在外面再也看不到了（特别是 MFC 的那三组宏）。所以，作者一直认为，从软件工程思想、方法教学和学习的角度来看 MFC 更具有价值。

另外，要知道MFC是一个技术发展过渡阶段的产物，它一头连接着面向过程技术的Win32（尽管它也含有初级的面向对象思想），另一头连接着完全面向对象技术的.NET。因而，从教学和学习的策略上讲，只有学习 MFC 才可以达到突破一点、打通全局的功效。至于在实际开发工作中是使用 MFC，还是使用.NET 或其他什么工具，则是另一码事。

2．在教材的内容上，要“十指”兼顾，还是只要“一指”

显然这是针对本书的内容而提出的问题，因为本书几乎涵盖了微软全部重要技术，从而担心会使学生“样样通，样样松”。

这个问题太辩证，本人很难说得明白，但作者认为，高等教育培养目标中的：“厚基础，宽口径，适应性强”是完全正确的，所以这个问题也就无需再作什么太多的讨论。

当然，作者也明白提问者的苦衷，说到底，就是担心是否能就业的问题，于是也就有了“针对性强、上手快”、“宁要一指强、不要十指弱”之类的说法和做法。但作者认为，这不符合高等教育培养目标，本人也从来不信那些口径真正宽、基础真正厚的毕业生会找不到工作。造成现在就业难的现象，除了其他原因之外，恰恰证明了我们的毕业生口径还不够宽、基础还不够厚。但要清楚，口径宽和知识杂（课程门数多）不是同一个概念，同样，基础厚和基础课时数长也不是同一个概念。也就是说，鲁迅笔下的那个知道“回”字的四种写法，却连要饭都不会的孔乙己，决不是“厚基础，宽口径”。诚然，处理好“十指”和“一指”的关系，确实是个说说容易做起来难的事，但坚持“厚基础，宽口径”总不会错，否则学生真的就会没饭吃。想想看，按照现在某些学校仅以某种开发工具的使用，或以语言课为主来展开教学的方法，其毕业生的发展前景会好吗？

再者，既然 Win32、MFC 和.NET 都是微软技术，那么它们的核心思想和技术一定是个不断生长的过程，是一脉相承、相互衔接的一个完整体系，如果不把它们组织为一门课程而把它们割裂开来，那是不太合适的。当然，关键还是在于是否能准确地从 Windows 中把反映了现代软件工程的核心理论和技术提炼出来，从而使学生能够举一反三，在理论指导下全面地提高应用能力，既“样样通”又不“样样松”。这也正是本书所力图解决的问题。

3．能否以面向对象思想和设计技术为基点，将微软的 MFC、C#和 Sun 的 Java 结合起来写成一部教材

显然，与前一个问题相比，这是另一个极端。但这绝对是一个极富创造性的大课题，也是高等教育应有之义，如果能做到这一点，无疑是对高等学校计算机教育的一大贡献。不讳言，这也是作者多年梦寐以求，正在努力的一个方向，尽管可能做不到。但作者相信，只要有这个想法，总会有人能做得到。所以在此也提出这个问题，希望有志于此的同仁能在这方面作一些努力，从而使人们能早日见到相关的教材。

综上所述可知，本书信息量巨大，需要大量的素材来支撑，所以在编写过程中作者参考了大量文献和资料，在此，作者谨对这些文献和资料的作者表示诚挚的谢意。

另外，将信息量如此巨大的 Windows 技术在一本仅为几十万字的教学用书中进行介绍，

无论在选材，还是内容组织方面，本书一定会存在一些不足之处。特别是因微软的某些技术并不公开，导致本书有些内容在编写时难度很大，所以恳请读者，如果发现了问题，欢迎与作者联系，以期有机会对本书作进一步改进。在此，先衷心表示感谢。

最后请注意，这是一本注重说理的教学用书，对于那些希望了解 Windows 程序设计技巧以及开发环境、控件使用方法等细节的读者，应选择学习更侧重技术性的书籍。为了便于教学，本书所涉及的程序请从机械工业出版社教材服务网（www.cmpedu.com）下载。

参加本书编写的作者有任哲、苗巍、曹宏宇。

作者的电子信箱为：renzhe71@sina.com，欢迎来信！

作　者

2008 年 12 月

目　录

第 1 章　Windows 程序基础

Windows 是一种应用于微型计算机的操作系统，它为应用程序提供了一个多任务环境，这个环境具有一致的图形化窗口和菜单。在 Windows 操作系统上运行的应用程序叫做 Windows 应用程序。

本章主要内容：

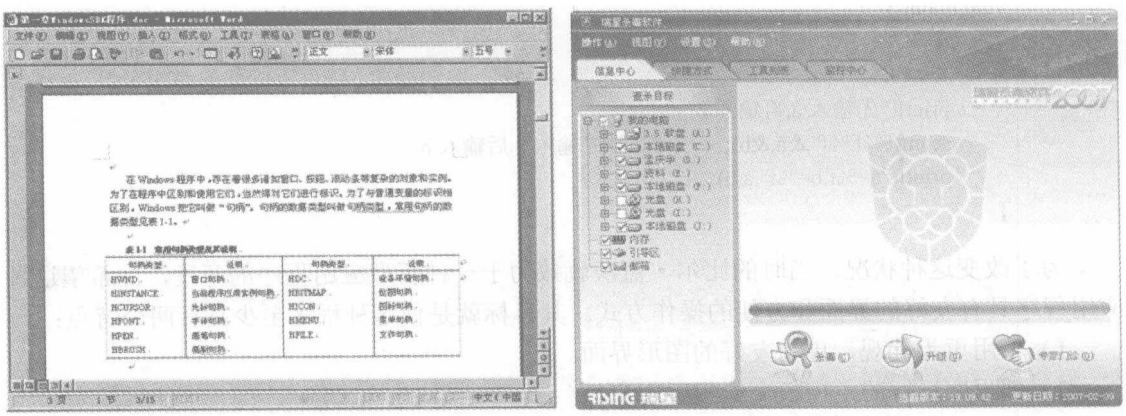
- Windows 的基本概念。
- Windows 应用程序中的数据类型。
- Windows 应用程序的消息处理机制。
- Windows 应用程序的代码重构。

1.1　Windows 应用程序的基本概念

凡是运行在 Windows 操作系统上的应用程序就叫做 Windows 应用程序。这种应用程序具有两大特点：一是具有图形界面；二是它是事件驱动的。

1.1.1　窗口界面与 Windows 史话

Windows 应用程序的一个突出特点是它有一个美观的图形用户界面（GUI），如图 1-1 所示。这种图形用户界面与键盘和鼠标的配合，大大方便了用户对应用程序的控制与操作。



a)　　b)

图 1-1　Windows 应用程序的图形窗口界面

a) 基于窗口的应用程序　b) 基于窗体（对话框）的应用程序

观察程序的图形界面，可以发现，它由许多不同的图形元素组成，其中某些图形元素在接受了用户的某个动作后，可以使程序执行某种相应的操作。例如，用鼠标单击图 1-1a 窗口界面的工具条最左边按钮，程序就会建立一个新的文件；而单击工具条上带有软磁盘图标

的按钮，程序就会把当前文件存盘。

显然，这个图形界面是 Windows 应用程序与用户交换信息的一个“窗口”。简单的 Windows 应用程序只有一个窗口，复杂的 Windows 应用程序可能有多个窗口，这也是这种操作系统被叫做 Windows 的原因。

其实，凡是最早使用过 Windows 的人都知道，早期的 Windows 还不是一个完整的操作系统，它借用了 DOS 的硬件抽象层，但它实现了虚拟存储管理和多任务的并发运行，如图 1-2a 所示。

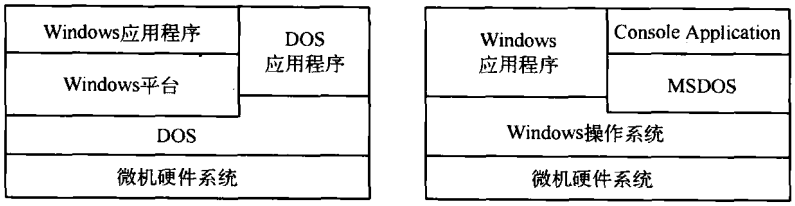


图 1-2　Windows 与 DOS 的关系

a) 早期的 Windows　b) 后来的 Windows

之所以出现这种状况，自然有它的历史原因。在早期，微型计算机使用的操作系统就是 DOS。人们天天面对 DOS 的那个黑黑的字符界面，工作得极其辛苦和枯燥。除了界面难看之外，DOS 程序还有一个令程序用户生厌的特点，即它总是按照程序的执行顺序强迫用户做一些工作，如果用户不按照程序所指示的顺序来输入数据，那么就会出错，例如，在如下程序：

```
#include<stdio.h>
void main( )
{
    float a,b;
    printf("先输入 a,再输入 b");
    scanf("%f,%f",&a,&b);        //必须先输入 a 后输入 b
    printf("a=%f,b=%f",a,b);
}
```

为了改变这种状况，当时的比尔·盖茨就致力于一种新型应用程序的开发，并希望这种应用程序具有友好的界面和方便的操作方式。其目标就是使这种程序至少具有两个特点：

1）采用更为美观，更为友好的图形界面。

2）把程序运行的主控权交给程序用户。

于是，为了支持图形界面和用户鼠标的操作，他们以 DOS 为运行平台，创建了一个大型函数库，从而实现了对图形程序界面的支持。随后，由于计算机硬件从 16 位发展到了 32 位，硬件的发展给虚拟内存管理和多任务管理提供了强大的支持，于是微软又在 DOS 的硬件抽象层之上开发了操作系统部分，即 Windows。在那时，人们开机之后首先进入的是 DOS，然后再键入命令>C:\Windows，系统才会启动 Windows。

从 DOS 的角度来看，当初的 Windows 是运行在 DOS 上以 main( )为起始函数的一个应