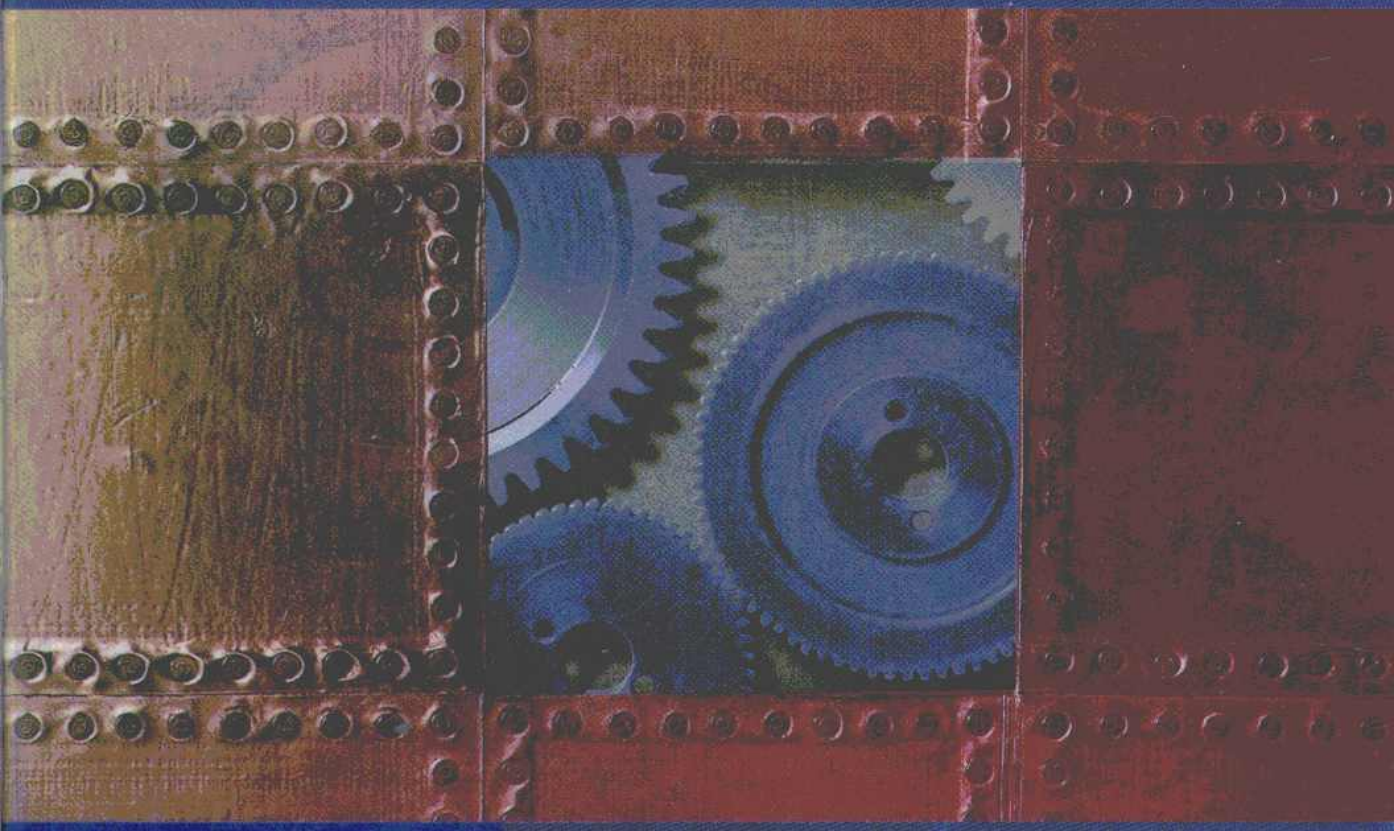
SAMS

计算机技术 译林 精选系列

附光盘

Windows 2000 API 超级宝典



[美] Richard J. Simon 著

萧湘工作室 译

人民邮电出版社

计算机技术译林精选系列

Windows 2000 API 超级宝典

[美] Richard J.Simon 著

潇湘工作室 译

人民邮电出版社

图书在版编目（CIP）数据

Windows 2000 API 超级宝典/（美）西蒙（Simon,R.J.）著；潇湘工作室译.
—北京：人民邮电出版社，2001.6

(计算机技术译林精选系列)
ISBN 7-115-09239-7

I. W… II. ①西… ②潇… III. 窗口软件，Windows 2000—软件接口—程序设计
IV. TP316.7

中国版本图书馆 CIP 数据核字（2001）第 24506 号

计算机技术译林精选系列
Windows 2000 API 超级宝典

◆ 著　　[美] Richard J.Simon
译　　潇湘工作室
责任编辑　李　际

◆ 人民邮电出版社出版发行　北京市崇文区夕照寺街 14 号
邮编　100061　　电子函件　315@ pptph.com.cn
网址　http://www.pptph.com.cn
读者热线　010-67129212　010-67129211(传真)
北京汉魂图文设计有限公司制作
北京顺义振华印刷厂印刷
新华书店总店北京发行所经销

◆ 开本:787×1092　1/16
印张:90.5
字数:2 240 千字　　2001 年 6 月第 1 版
印数:1－3 000 册　　2001 年 6 月北京第 1 次印刷

著作权合同登记　图字:01－2000－2483 号
ISBN 7-115-09239-7/TP·2173

定价:148.00 元(附光盘)

前　言

本书的目的是使程序员在开发Windows 2000应用程序时能够节省时间。Windows 2000推出的新的API和标准给开发人员带来了新的挑战。我将在本书中尽可能详细地介绍这些变化。我很想使本书的内容再丰富一些，但我也相信，本书已经包括了多数程序员使用的主要的API。

尽管我们在本书中说明了许多只是在Windows 2000中实现的新API，但本书的重点是为Microsoft Windows 2000和Windows 98支持的Win32 API提供完整的参考。我们提供了每个API的兼容性，以及在这两种操作系统上使你的应用程序兼容所需要的信息。本书着重于这两种操作系统支持的Win32 API，而不是整个Win32 API，因为多数应用程序不要求专用于Windows NT的API。

对于要编写窗口程序的Windows程序员，如果他面对一大堆组织混乱的文档资料，而且缺少示例，将会望而却步。确定函数“假设”的工作方式及其“实际”的作用，是相当耗费时间的。本书包括每个函数的完整说明和详细的参数说明，对于多数函数，我们提供了简单的示例，并说明其功能。

每章中包括的函数都具有相关性。例如，第3章介绍的函数是创建窗口所需要的。在各章的开始，我们首先简要概述本章所涉及的函数和每个函数共享的背景信息，随后是本章函数的提要并对函数进行详细说明，同时还附有示例。

本书的示例简要而全面，提供了函数所有的变量说明和用法，还有所要求的支持函数。示例的目的不是为了指导，而是为了在一个简单的工作程序中说明一个或几个函数。为了使示例简明扼要，我们采用了在正规应用程序中不适用的快捷方式。这种做法有助于消除在示例中必须使用的函数和普通Windows应用程序的支持编码之间发生混淆。

本书是用Windows 2000的发行版本完成的，我们尽了一切努力保证本参考信息完整而准确。本书配套光盘上包括用于示例的带有Service Pack 3的Microsoft Visual C++6编译程序。

希望您喜欢本书。编程快乐！

Richard J.Simon

第 1 章　Windows 2000 程序设计

Microsoft Windows 2000 是对 Windows NT 4 的重要升级，它引进了许多新的功能。这些功能有一些超出了标准的 Win32 API 的范围，因此在本书不对这些功能进行说明。然而，我们说明了几个新的 API 和对现有 API 所做的修改。Windows 2000 包括曾为 Windows 98 保留的几个功能，例如，IMC（Image Color Matching，图像颜色匹配）。还有几个是 Windows 2000 专用的新 API，Windows 98 中没有这些内容。Windows 2000 和 Windows 98 为开发人员提供了应用程序之间平滑、无缝的集成，例如，拖放功能和 OLE（Object Linking and Embedding，对象链接与嵌入）。为 Windows 2000 和 Windows 98 而设计的应用程序对数据提供了更多的图形化表示方法，给用户创建了更直观的界面，但是给负责开发的程序员带来了更多的困难。

1.1　用户界面功能

在基本的图形化用户界面演变为更加面向对象的界面的过程中，Windows 已经从以应用程序为中心的界面转化为以数据为中心的界面。这就意味着，因为应用程序已经成为处理界面中对象的工具，所以开发人员必须重新考虑应用程序的用户界面。

在 Windows 2000 和 Windows 98 中有许多功能，既有可视的功能，又有开发人员在设计应用程序时需要解决的幕后的功能。以下列出一些对最终用户最为直观的功能：

- 桌面是操作系统的一个功能部分。应用程序和文档可以放在桌面留待以后使用。
- 通过任务栏，用户可以通过指向和单击来访问应用程序，并且能在当前运行的应用程序之间进行切换。
- 对象可以拖放到桌面上，也可以在应用程序之间进行拖放。
- 在整个操作系统和应用程序内可以使用上下文相关的帮助。
- 用鼠标右键可以引出弹出菜单，选择上下文相关的对象。
- 所有应用程序的常见控件（如树形视图、列表视图、工具栏等）均提供相同的外观。
- 属性表为转换和查看对象属性提供了一个方便的用户界面。

1.2　最低要求

Windows 2000 应用程序有一系列最低要求，必须满足这些要求，才能认为是真正的 Windows 2000 应用程序。由微软公司制定的这些标准可用于验证各种应用程序，以便确定这些程序是否可

以称为 Windows 2000 应用程序。

Windows 95 的发行，使得首次出现了微软的标志程序，其要求很容易达到。由于 Windows 2000 应用程序复杂多变，如今的要求范围非常广泛，根据开发的应用程序类型，这些要求中有几项例外情况。在本书配套光盘的 WIN2KAPPSPEC.DOC 文件中，可以查到这些要求的当前版本。在微软的网站 http://msdn.microsoft.com/certification/appspec.asp 上，也可以查到最新版本。

1.3 样式指南

本节介绍开发人员在开发 Windows 2000 应用程序时应该使用的基本约定。其中一些标准具有灵活性，这意味着，当把一个应用程序看作 Windows 2000 应用程序时，不要求必须满足这些标准；但有一些标准是必须满足的，这些我们在前一节中已经讨论过。然而，开发人员应该尽可能严格地遵守所有标准，以便创建出能为最终用户提供共同的外观和感觉的 Windows 2000 应用程序。

1.3.1 窗口

用户通常使用两种不同类型的窗口同数据对象交互：一个主窗口和几个辅助窗口。主窗口主要用于查看和编辑，而辅助窗口允许用户指定选项或提供主窗口中对象的详细资料。辅助窗口通常是属性表或对话框。

1. 窗口组件

如图 1.1 所示，典型的主窗口由框架、客户区和标题栏组成。如果窗口内容超出可视区域，用户可以使用滚动条滚动窗口。窗口可以包括菜单栏和状态栏等其他组件。

每个窗口都有一个限定窗口形状的窗口框架。如图 1.1 所示，可调整大小的窗口有一个边框，其中提供了用以调整窗口大小的控制柄和关闭窗口按钮。如果窗口不能重新调整大小，则其边框没有可调整大小窗口那样宽。

标题栏

标题栏在上边框的下面，扩展了窗口的宽度。标题栏标识窗口显示的内容，并让用户执行许多窗口操作，它是移动窗口的控制点，上面有系统菜单及应用程序图标、最小化按钮、最大化按钮、还原按钮、关闭窗口按钮。在应用程序小图标上按鼠标左键、按 Alt+空格键、在标题栏上按鼠标右键等，均可以访问应用程序的系统菜单。

标题栏的最左边显示一个小的对象图标，代表当前窗口中正在查看的对象。如果应用程序是一个“工具”或“实用程序（指那些不是用于查看数据对象的应用程序）”，图标将是一个缩小为 16×16 像素的应用程序图标，应用程序名用作标题栏名称（如图 1.2 所示）。如果应用程序包括一个说明符，它应该包括一个破折号和说明文字。例如，Windows Explorer 会指出正在查看的是哪个项目。如果应用程序没有在系统中注册小图标，Windows 2000 将会使用缩小的 32×32 像素的标准图标，这可能会产生无法预料的结果。

如图 1.3 所示，像 Microsoft WordPad（写字板）这类使用数据对象的应用程序，在标题栏中使用代表数据对象的图标。在本章后面的“系统集成”一节中，将说明注册应用程序图标和数据文件类型的方法。标题栏文本由数据对象名和应用程序名组成。如果一个数据对象当前还没有名称，可以用简称自动生成一个，如 Document（文档）。如果不能有效地提供一个默认名称，可以在标题栏文本中使用占位符（Untitled，无标题）。

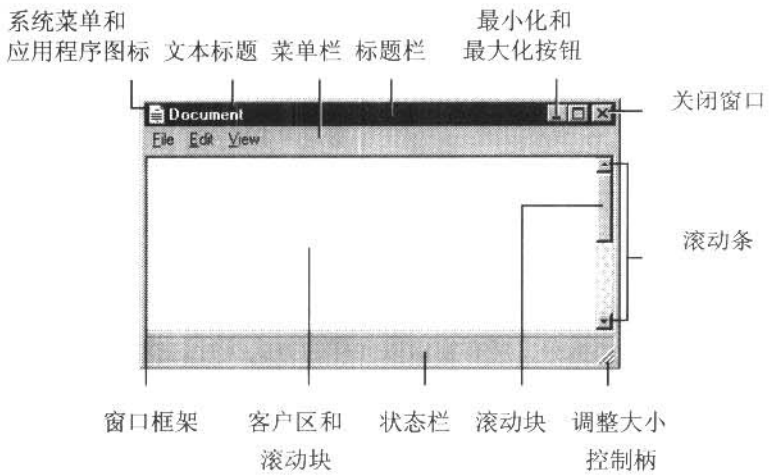


图 1.1　主窗口组件

图 1.2　“工具”或“实用程序”应用程序的标题栏

图 1.3　Document 标题栏

MDI（multiple-document interface，多文档界面）应用程序（如 Jasc Paint Shop Pro，见图 1.4）在父窗口标题栏中使用应用程序的图标，这个图标代表子窗口标题栏中的数据对象。应用程序名作为标题栏文本出现在父窗口上，数据对象名用作子窗口的标题栏文本。

当用户使 MDI 应用程序中的子窗口最大化时，隐藏子窗口的标题栏，并且组合标题（见图 1.5）。子窗口的标题栏图标显示在父窗口的菜单栏左上角。

同窗口相结合的命令按钮出现在标题栏右边。表 1.1 说明主按钮及其执行的动作。

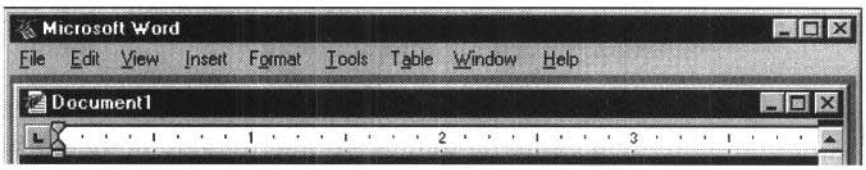


图 1.4　MDI 标准标题栏

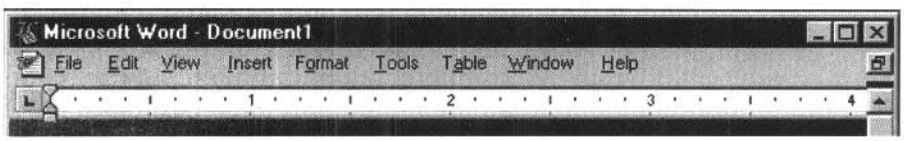


图 1.5　MDI 最大化的标题栏

表 1.1　　标题栏按钮

命令按钮	操作
	关闭窗口
	窗口最小化
	窗口最大化
	窗口还原

2. 窗口操作

用户在窗口上可以执行的基本操作包括激活、取消窗口活动状态，打开、关闭窗口，移动窗口，调整窗口大小，滚动和拆分。所有窗口以同样的方式执行这些操作，使用户在各个应用程序中可以以一致的方式操作窗口。

激活窗口

即便用户可以同时显示多重窗口，但多数用户一次只使用一个窗口，这就是活动窗口。活动窗口以其标题栏和窗口框架的颜色同其他窗口区分开。不活动窗口用不活动标题栏和框架颜色（这些颜色可以由用户设定，是动态的）。利用第 21 章叙述的 GetSyscolor 函数，应用程序可以检索这些颜色。当一个窗口不活动时，应用程序应该在窗口内隐藏选择的对象，以免在键盘输入时出现混乱。只有光标可以在不活动窗口上进行拖放操作。

打开和关闭窗口

当用户打开主窗口时，应该在任务栏创建一项。所有打开的主窗口都应该有一项，使用户能方便地在打开的应用程序之间进行切换。首次打开一个应用程序时，要使用合适的默认大小和位置。如果用户打开以前曾使用过的应用程序，程序将还原到以前的大小和位置。如果合适的话，还会还原用户界面的设置，例如选择状态、滚动位置以及其他相关的视图信息。如果用户要打开一个已经在运行的应用程序，请按表 1.2 的建议进行。

表 1.2　　已打开窗口的动作

文件类型	动作
文档或数据文件	启动对象现有的窗口，在窗口 Z 次序的顶部显示出来（就是各个窗口在顶部相互重叠的次序）
应用文件	显示一个消息框，指示应用程序已经有一个打开的窗口，用户可以选择切换到打开窗口，或者打开另一个窗口。两种选择都能激活已选的窗口，并将其送到窗口 Z 次序的顶部
已经打开的文档文件	激活文件现有的窗口。该窗口的 MDI 父窗口以 MDI 应用程序窗口的 Z 次序出现在窗口的顶部，在 MDI 父窗口内的 Z 次序顶部显示出文件
文档文件尚未打开，但与其关联的 MDI 应用程序已经在运行	为关联的文件创建一个新的 MDI 子窗口，并放到有 MDI 父窗口的 Z 次序顶部，把 MDI 父窗口放在 Z 次序的顶部

当用户关闭主窗口时，也关闭了所有独立的辅助窗口，并删除了任务栏项。对大多应用程序而言，当用户关闭主窗口时，应用程序被关闭。然而，一些应用程序（如打印机队列）在主窗口关闭时并不停止。在关闭主窗口之前，会显示一条消息，询问用户是保存还是取消所做的改变，

或者取消关闭操作。这就使得用户能够控制没有保存的未决事务。如果没有未保存的事务，则不提示用户就会关闭窗口。

移动窗口

用指向设备（如鼠标或光笔）把窗口的标题栏拖到新位置，或者使用弹出菜单的 Move（移动）命令，可以移动窗口。一般情况下，在移动窗口时，窗口的框架也会移动。在完成移动操作后，会更新窗口。在一些系统中，移动窗口时可以动态重绘窗口。当窗口移动时，应用程序不应该阻止重绘窗口。在用户移动窗口期间，画面还应该尽可能提供一个平滑的外观。

选择 Move（移动）命令后，用户可以用键盘移动窗口，使用箭头键并按回车键便可以移动和建立新的窗口位置。按 Esc 键中止移动操作，使窗口保持在前面的位置。应用程序将不允许用户把窗口移动到不能访问的位置。

重新调整窗口大小

除非信息以固定形式显示（如 Windows 计算器程序），主窗口由可调整大小的框架构成。用户有几种调整窗口大小的方法：调整框架大小、最大化和最小化操作、还原以及利用大小控制柄。在正常情况下，允许调整大小的应用程序窗口应该实现所有这些功能。窗口至少应该实现最小化和还原操作。

在窗口有调整大小的边框时，用户可以用指向设备拖动边框来调整窗口大小。另一个方法是从窗口的弹出菜单使用 Size（大小）命令。一般情况下，在用户调整窗口时，窗口的外形随着指针的移动而移动。在一些系统中调整窗口时，会动态重绘窗口。移动窗口时采用的规则同样适用于这里——不应该阻止重绘窗口，重绘应该尽可能地有效。通过选择 Size 命令并使用箭头键，用户可以使用键盘调整窗口。使用回车键来接受新的窗口大小，用 Esc 键中止调整大小的操作。

顾名思义，最大化操作是把窗口调整到最大。这个大小是排除任务栏所用空间的显示器的大小。对 MDI 子窗口而言，最大化的大小等于父窗口客户区的大小。应用程序不应该使用固定的显示器大小，而应该采用系统限定的形状和大小。使用标准系统函数（例如 SetWindowPlacement 函数）可以正确地自动放置窗口。从窗口的标题栏选择最大化命令按钮或从窗口的弹出菜单选择 Maximize（最大化）命令，用户可以最大化窗口。最大化窗口后，用还原按钮替代最大化按钮，并禁用 Maximize 命令，窗口弹出菜单的 Restore（还原）命令变为有效。

在最小化应用程序时，应用程序缩小在任务栏上。对 MDI 子窗口而言，要在父窗口内最小化窗口。要想最小化窗口，用户要从窗口的标题栏选择最小化命令按钮或从窗口的弹出菜单选择 Minimize（最小化）命令。窗口最小化后，Minimize 命令变为无效，Restore 命令变为有效。

选用还原命令按钮（仅用于最大化窗口）或窗口弹出菜单的 Restore 命令，用户可以使最大化或最小化的窗口还原。窗口还原后，窗口弹出菜单的 Restore 命令变为无效，Maximize 命令（如果适用）和 Minimize 命令变为有效。在还原最大化窗口后，标题栏上以最大化命令按钮替代还原命令按钮。通过选择任务栏上的应用程序图标、从窗口弹出菜单选择 Restore 命令、选择还原命令按钮（只用于最大化窗口）或使用 Alt+Tab 键和 Shift+Alt+Tab 键，用户都可以还原窗口。

在应用程序内，所有可调整大小的窗口都应该含有大小控制柄。如图 1.1 所示，大小控制柄是一个特殊的控制柄，同时用于水平和垂直方向调整窗口。在使用大小控制柄时，用户用指向设备拖放控制柄，以调整边框时使用的同样方式重新调整窗口。大小控制柄应该永远放在水平滚动条和垂直滚动条的相交点，即窗口的右下角。如果窗口有状态栏，大小控制柄应当放在状态栏的右边。窗口一次只能有一个大小控制柄。

滚动窗口

当窗口太小、无法完全显示一个数据对象时，窗口应该支持滚动，使用户能移动窗口的视图，看到数据的其他部分。如图 1.1 所示，应用程序应该使用一个滚动条（含有滚动箭头的矩形控件）、一个滚动块（小方块）、一个滚动轴。

根据所查看的数据对象，窗口可以包括一个垂直滚动条（如图 1.1 所示）、一个水平滚动条或者两者皆备。如果数据对象太宽，可使用水平滚动条；如果太高，则使用垂直滚动条。如果不要求用滚动条查看数据对象，则不会含有滚动条。然而，如果有滚动条的窗口在调整大小后不再需要滚动条，可以继续显示它。连续显示滚动条会使屏幕的外观更简洁。

滚动条的两端会出现箭头，指向窗口或视图移动的方向。用户选用一个箭头时，窗口朝箭头的方向移动一个滚动单位。滚动单位是应用程序限定的量度单位，与窗口显示的数据对象相适应。举例来说，在选用滚动箭头后，文本文件一次滚动一行，而图片则按一定像素数移动。在整个滚动范围内，滚动单位应该保持一致。当窗口不能继续向一个方向滚动时，应当禁用与该方向相对应的箭头。

滚动块沿着滚动条移动，指示窗口当前显示的数据对象的大概位置。例如，如果窗口当前显示数据对象的开头部分，滚动块会在滚动条的顶部；如果显示数据对象的中间，滚动块移到滚动条的中间。从 Windows 95 开始，滚动块较以前的 Wnidows 版本有了改变。现在，滚动块按照窗口查看的数据对象数量和数据对象的总体大小的比例显示大小。例如，如果窗口可以显示数据对象的整个内容，滚动块大小会调整为滚动条的整个长度。如果窗口可以显示一半数据对象，滚动块的大小是滚动条的一半。滚动块还可以用于另一个功能：用户可以拖动滚动块，滚动到数据对象内的近似位置。

滚动轴还可以作为部分滚动接口。用户用指向设备点击滚动轴时，窗口将滚动与可视区域相同的大小。滚动方向依选择滚动块哪一边的滚动轴而定。例如，如果用户点击垂直滚动条滚动块下面的滚动轴，窗口将向下滚动。当使用滚动轴滚动时，应该从当前的屏幕上看到应用程序数据对象的一小部分。例如，如果窗口正在查看文本文件，而且用户向下滚动，在完成滚动后，在窗口底部看到的行会变成在窗口顶部可以看到的行。

到当前为止，我们已经完整地介绍了与用户滚动窗口有关的滚动技术。有时，窗口应该自动滚动。下面列出窗口应该自动滚动的常见情况：

- 当用户把选择对象拖过窗口边缘时，窗口应该朝拖动的方向滚动。
- 当用户把对象拖动到靠近可滚动区域的边缘，对象可以放在窗口内部，则朝相应的方向滚动窗口。
- 当用户在窗口边缘输入文本或在窗口边缘的位置里放置一个对象时，窗口应该滚动，使用户能把注意力集中在正在使用的对象区域内。滚动的程度取决于应用程序的上下文。例如，用户生成在窗口内键入的新的一行文本时，一次应该滚动一行。在水平滚动时，滚动单位应该大于单个字符的宽度，以免连续滚动。此外，用户移动靠近窗口边的图像对象时，滚动单位以对象的大小为基础。
- 如果在选择对象或光标移动时进行操作（如查找），则会滚动窗口来显示新的选择对象或光标的位置。另一个示例是输入形式——在光标向当前窗口视图没有显示的区域移动时，窗口应该随着光标滚动。

应用程序还应该使用键盘上的导航键来支持滚动。如果窗口含有光标（如文本编辑器），用户按动窗口旁边的导航键，窗口会朝相应的方向滚动。没有光标的窗口（如图形图像查看器），总要用导航键滚动。Page Up 键和 Page Down 键相当于用户选用滚动轴。然而，在有光标的窗口里，

光标也会移动。

不要用相邻的控件使滚动条拥挤。有时，紧靠滚动条放置控件（如拆分框）十分方便。然而，相邻的控件太多，会使用户难以滚动。如果需要大量控件，最好选用标准工具栏。

拆分窗口

把一个窗口拆分成两个单独的视图，可显示有关数据对象的不同信息，这样对于应用程序来说非常有好处。例如 Windows Explorer（见图 1.6）。窗口内的这些单个视图称为窗格。

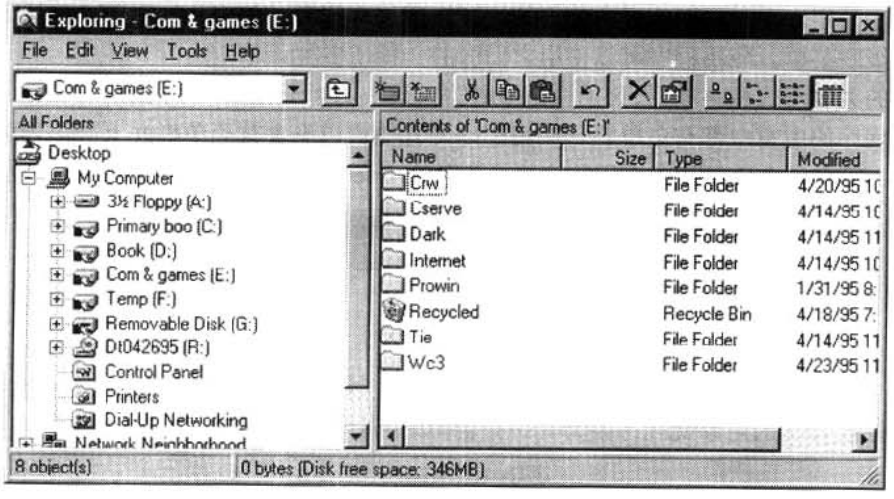


图 1.6　带拆分条的窗口

虽然可以拆分窗口同时查看多个数据对象的内容，但最好还是用 MDI 应用程序结构来同时显示多个文件，可以使用户更清楚地了解单个文件。

默认情况下，或者作为用户可配置的选项，窗口可以设置为拆分成窗格。如果窗口允许用户把窗口配置为窗格，在相应的位置应该有一个拆分框（在垂直滚动条的顶部或水平滚动条的左边）。拆分框是与滚动条末端相邻的小矩形（通常是可调整边框的大小），它可以拆分或调节窗口。

用户把拆分框拖动到预定位置来拆分一个窗口。当用户在拆分框或拆分条的热点位置放置指针时，指针的图像应该改变，以便向用户提供反馈。当用户拖动拆分框或拆分条时，同时用指针移动了拆分框和拆分条的轮廓（见图 1.7）。在完成拖动时，显示从窗口一端延伸到另一端的拆分条来限定窗格。拆分条的大小至少应该等于调整边框的大小。

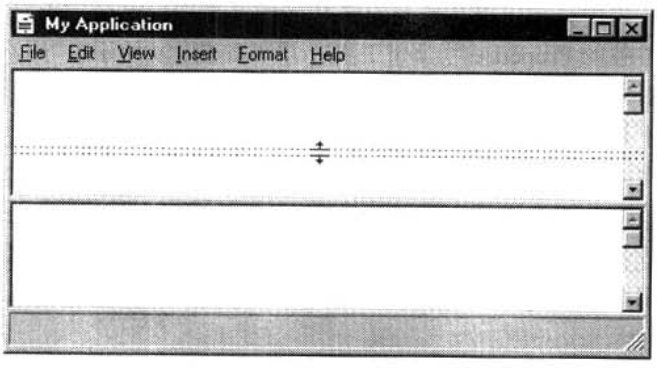


图 1.7　移动拆分条

为了拆分窗口，提供键盘接口，并包括 Window（窗口）或 View（视图）菜单的 Split（拆分）命令。用户选用 Split 菜单项时，使用箭头键从中间拆分窗口并允许用户放置拆分条。按回车键在当前位置设定拆分条。Esc 键中止拆分模式。

1.3.2 菜单

应用程序都使用某种菜单同用户交互。至少，用户永远可以使用主窗口的弹出菜单。其他菜单视应用程序而定，用户可以选择所使用的选项。应用程序可以使用几种菜单，包括下拉菜单、弹出菜单和级联菜单。应用程序可以使用它们中的一种或多种菜单。

最常见的菜单形式是菜单栏。菜单栏是横跨窗口顶端的矩形区域，在标题栏的下方（见图 1.8）。大多数应用程序的主窗口都有菜单栏。应用程序的其他窗口里含有菜单栏并不好，因为这样做会使用户混淆。所有包含菜单栏的应用程序至少应该有 File（文件）菜单栏和 Help（帮助）菜单栏。

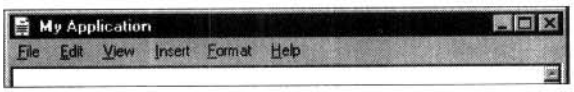


图 1.8 菜单栏

即使一个应用程序包含菜单栏，在适当的位置也应该有弹出菜单（见图 1.9）。弹出菜单使用户能有效地同屏幕上的对象交互，因为其中只含有与指针下面的对象相关的那些菜单项。

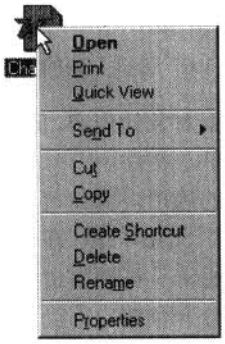


图 1.9 弹出菜单

应用程序应当用鼠标右键在应用程序里显示弹出菜单。在弹出菜单的左上角按鼠标右键，这样来使用指针的位置。弹出菜单可以含有与对象相应的菜单项。然而，不是每个对象属性都有菜单项，菜单应该含有单独的 Properties（属性）菜单项（如图 1.9 所示）。用户选择 Properties 菜单项时，将显示一个有对象属性的属性表对话框。不要只是通过弹出菜单进行访问操作。注意，弹出菜单不一定只限于下拉菜单所显示的项目。

在弹出菜单中确定菜单项的顺序时，要运用以下标准：

- 首先放置对象的主要命令，如 Open（打开）、Play（播放）、Print（打印）。然后是对象的移动命令、对象支持的其他命令、What's This?（这是什么）命令（如果支持的话）。
- 移动命令的顺序应该是 Cut（剪切）、Copy（复制）和 Paste（粘贴），以及其他特殊的 Paste（粘贴）命令。
- 在适当的情况下，Properties 命令应该是菜单上的最后一项。

1. 菜单接口

用户可以用指向设备（如鼠标、光笔和键盘）访问菜单。用指向设备进行菜单导航是很简单的。用户使用指向设备单击菜单来选择需要的项目。如果选择的菜单项是有下拉菜单的菜单标题，将显示下拉菜单。用户把指向设备移到菜单项上时，此项会突出显示，呈现出当前菜单的选择。如果用户按下用来选择菜单的鼠标键，在放开鼠标键时，选定的菜单项就是所要选择的菜单项。如果用户通过释放鼠标键来选择菜单，下次按鼠标键时就决定了选择那个菜单项。

下拉菜单的键盘接口使用 Alt 键激活菜单栏。在激活菜单后，用户用箭头键在菜单中移动，通过按回车键选择菜单项。使用 Esc 键可关闭打开的下拉菜单，另外，该键可使菜单栏取消活动状态。在按住 Alt 键的同时（或者按了 Alt 键以后）按一个字母键，会显示与该字母键关联的下拉菜单（在菜单项中是加下划线字符）。顺序按字母键会显示相关的菜单项。所有菜单项都应当有一个限定的访问字符。有关限定访问字符的内容见第 6 章。

在下拉菜单中可以给菜单项指定快捷键（也称加速键）。反复访问的命令应该有相关的快捷键。在用户按快捷键时，不显示菜单，而是立即执行菜单命令。表 1.3 列出了一些常见菜单项的快捷键和访问字符。访问字符是加下划线字符。

2. 菜单标题

下拉菜单和级联菜单都应有菜单标题。下拉菜单的菜单标题是出现在菜单栏里的项。级联菜单的菜单标题是父菜单项的名字。选择代表全部菜单内容的菜单标题，能清楚地表达菜单里全部项目的目的。菜单标题不使用空格；Windows 把空格解释为一个标题的结束和另一个标题的开始，这样，菜单栏里将会有两个菜单标题。此外，应该避免使用少见的复合词。

3. 菜单项

菜单中包含单独的选项，称为菜单项。菜单项可以是词语、图形，或者图形和词语的组合。菜单项应该提供给用户有关菜单项特点和效果的可视线索。图 1.10 显示了有不同类型菜单项的菜单。

菜单项分成相关的组，并用菜单分隔符分开。标准的分隔符是一条横线，如图 1.10 所示。不要把菜单项用作分隔符，这样不够标准，会使用户搞混用户界面。在特定上下文中无法使用的或不适合的菜单项应该被禁用或删除。留下可启用的菜单项，并用消息框告诉用户该项目无效，这样做并不好，不会立即提供给用户反馈。相对于删除菜单而言，最好还是禁用菜单，因为这样会使用户界面保持稳定。然而，如果菜单项不能用，并且不再需要，就要删除它。如果禁用了一个菜单里的全部菜单项，就要禁用菜单标题。如果应用程序状态栏包括反馈信息，可表明此命令不可用并说明原因。在禁用自定义菜单项时，要遵守后面“可视化设计”一章中提出的标准。

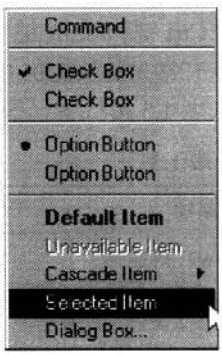


图 1.10　菜单项类型

菜单项类型

不同类型的菜单项可用来向用户提供有关菜单项的特性及其执行功能的直观反馈。如果菜单项命令在执行前要求附加的信息，菜单项应该跟有省略号（…）。这类命令（如 Save As…命令）通常显示一个对话框，来收集需要的信息。然而，并不是每个显示对话框或其他辅助窗口的命令都需要在菜单标题中有省略号。例如，用户完成 Properties 命令对话框后，不再要求进一步的参数或动作来完成命令，所以不需要省略号。引起消息框显示的命令也不需要省略号。

除了执行命令外，菜单项也可以切换对象或窗口的状态或属性。一个示例是 View 菜单，其中含有触发工具栏和状态栏视图的菜单项。这些菜单项可能与其他菜单项相互排斥，或者独自存在。如果其状态是独立的，则在状态启动为活动状态时，菜单前面标出复选标志（如图 1.10 所示）。如果状态同其他菜单项相互排斥，活动菜单项由一个选项按钮标记来指明（如图 1.10 所示）。相互排斥状态的示例是对齐文本，可以左对齐、居中或右对齐。

菜单可以限定可用快捷方式选择（如双击或拖放）的默认项目。例如，如果用户双击图标时默认的命令是 Open（打开），Open 命令将是默认菜单项。在指示默认菜单项时，它的外形显示成粗体文本。

菜单项标签

所有的菜单项都应该包括说明文本或图形标签（无论用哪一种，都应该保持一致）。在编写菜单项时，应该遵守以下标准：

- 一个菜单内使用唯一的项目名。虽然菜单项名可以在不同菜单里重复，代表相同或不同的动作，还是要避免这种情况。否则会使用户搞混。
- 要使用简短的词。如果需要，可以使用多个词，但冗长的菜单项名称会使用户难以浏览菜单。
- 在菜单内给每个菜单项限定唯一的访问键。常见的菜单项使用表 1.3 里的规定。可能的话，可以给常见的命令限定一致的访问键。
- 菜单项名应该遵守用大写字母开头的规则。除了在多词名称开头或结尾出现的冠词、连词和介词外，每个词的首字母均应大写。
- 即使文本属性说明特定的文本样式，也要避免用不同的文本属性形成单独的菜单项名。

菜单项快捷键

下拉菜单项也可以显示与命令相关的键盘快捷键。一般情况下，在具有快捷键的菜单的最长项目后，快捷键在第一个制表符的位置向左对齐。不要使用空格进行对齐，在菜单项名里放置一个制表符'\t'字符，适当地分开快捷键。弹出菜单中不应有快捷键。弹出菜单已经是快捷形式了。

Ctrl 键和 Shift 键作为修饰键，它使用“+”来与实际的键组合起来使用。例如，Copy（复制）命令有常见的快捷键 Ctrl+C。不要使用 Alt 键，因为这样会引起用户界面的问题。

表 1.3 提供了一些常见菜单项和与之相关的快捷键。

表 1.3　　菜单项快捷键和访问字符

菜单项	快捷键
New（新建）	Ctrl+N
Open（打开）	Ctrl+O
Save （保存）	Ctrl+S
Save As（另存为）	
Print（打印）	Ctrl+P
Print Preview（打印预览）	
Page Setup（页面设置）	

续表

菜单项	快捷键
Send（发送）	
Exit（退出）	
Undo（撤消）	Ctrl+Z
Cut （剪切）	Ctrl+X
Copy （复制）	Ctrl+C
Paste （粘贴）	Ctrl+V
Paste Special（选择性粘贴）	
Paste Link（粘贴链接）	
Clear （清除）	Del
Select All （全选）	Ctrl+A
Find （查找）	Ctrl+F
Find Next （查找下一个）	F3
Replace （替换）	Ctrl+H
Links（链接）	
Object Properties （对象属性）	Alt+Enter
Object（对象）	
Help Topics（帮助题目）	
About<Application Name>（关于<Application Name>）	

4. 常用菜单

所有应用程序都有一些下拉菜单，但这并不是说所有应用程序都需要有下拉菜单。如果一个应用程序要使用常用的下拉菜单，应遵守本节的标准。

File（文件）菜单

File 菜单用于对文件进行基本操作。图 1.11 显示了常见的命令和它们应该放置的地方。如果窗口关闭时对象仍处在活动状态，应用程序不支持 Exit 命令，可用 Close（关闭）命令替换 Exit 命令。

Edit（编辑）菜单

Edit 菜单应该包括通用的编辑命令，例如 Cut、Copy、Paste、OLE 对象命令，另外还有表 1.4 中的命令。图 1.12 显示了 Microsoft WordPad 的一个典型的 Edit 菜单。

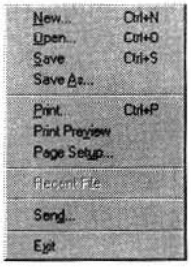


图 1.11　File 菜单

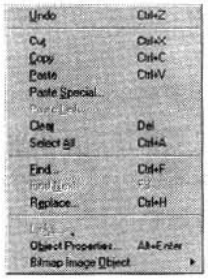


图 1.12　Edit 菜单

表 1.4　　可选的 Edit 命令

命令	功能
Undo（撤消）	撤消最后的编辑动作
Repeat（重复）	重复最后的编辑动作
Find（查找）	查找文本
Replace（替换）	查找并替换文本
Delete（清除）	清除当前所选的对象
Duplicate（复制）	创建当前所选对象的副本

View（视图）菜单

如图 1.13 所示，View 菜单包括改变窗口中数据视图的命令，例如，Zoom（放大）或 Outline（大纲）。窗口中还可以含有控制界面要素显示的命令（如工具栏、标尺、状态栏）。

Window（窗口）菜单

Window 菜单用于 MDI 应用程序对 MDI 子窗口进行管理（见图 1.14）。Window 菜单可以包括与 MDI 子窗口有关的命令，如 Cascade（层叠）、Tile（平铺）、Close All（关闭所有窗口）。Window 菜单的最后一项是为当前 MDI 子窗口保留的。

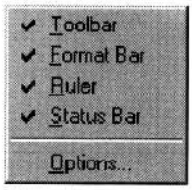


图 1.13　View 菜单

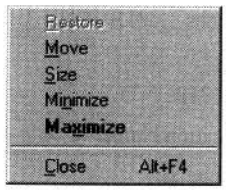


图 1.14　Window 菜单

Help（帮助）菜单

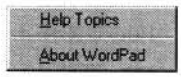


图 1.15　Help 菜单

Help 菜单项用来帮助用户使用应用程序（见图 1.15），其中包括一个 Help Topics（帮助主题）命令，用来提供对应用程序帮助文件 Help Topics 浏览器的访问。应用程序可以交替使用一组菜单命令访问 Help Topics 浏览器的特定页面，例如 Contents（目录）、Index（索引）、Find Topic（查找主题）。本菜单还可以放置其他用户帮助命令。

为了提供对应用程序版权和版本信息的访问，在 Help 菜单的底部包括一个 About application name 菜单命令。用户选择这个命令时，会显示一个窗口，其中有应用程序名、版权信息、版本号和其他应用程序属性。不要在这个菜单命令的末端使用省略号（…），因为这样会导致窗口不能接收用户的输入。

窗口弹出菜单

窗口弹出菜单替换了以前 Windows 版本中的系统菜单。用户在窗口标题栏按鼠标右键可访问这个菜单。为了向后兼容，用户也可以选择显示小图标的标题栏区域，按鼠标左键或同时按 Alt+空格键来访问窗口弹出菜单。

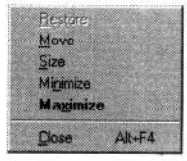


图 1.16　窗口弹出菜单

常见的窗口弹出菜单包括 Close（关闭）、Restore（还原）、Move（移动）、Size（大小）、Minimize（最小化）和 Maximize（最大化）命令（见图 1.16）。应用程序还可以包括应用于窗口或视图的其他命令，例如 Split

（拆分）命令。子窗口也包括一个窗口弹出菜单，但通常只有 Move（移动）、Close（关闭）等命令。

图标弹出菜单

以图标形式显示的数据对象的弹出菜单包括对于对象类型的操作。当光标在图标上时，用户按鼠标右键可访问图标弹出菜单。

包括图标的应用程序提供图标弹出菜单。例如，系统提供桌面上图标的弹出菜单。在文档中嵌入 OLE 对象的应用程序提供对象的弹出菜单。包容器应用程序使用对象的命令来提供图标弹出菜单。图 1.17 显示了一个典型的表示应用程序的图标弹出菜单。图 1.18 显示了一个表示文档的图标弹出菜单。

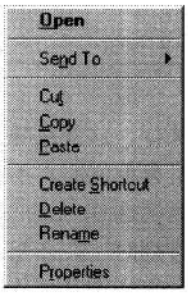


图 1.17　应用程序图标弹出菜单

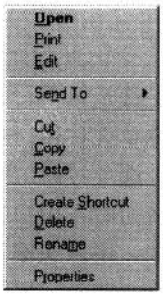


图 1.18　文档图标弹出菜单

1.3.3　输入和控件

控件是允许用户与应用程序交互并向应用程序提供输入的图形对象。一些控件仅仅对用户提供反馈或帮助解释用户界面。各种类型的控件都有其唯一的可视外观和操作。Windows 也允许应用程序定义新的控件和现有控件的变量。在设计控件时，要遵守本章后面的“可视外观”　节中的标准。本节介绍 Windows 应用程序中使用的常见控件类型。

1. 按钮

按钮和菜单项一样，是启动动作或改变属性的控件。Windows 应用程序可以使用三种基本类型的按钮——命令按钮、选项或单选按钮，或者复选框。

命令按钮

命令按钮是矩形控件，在按动它时会执行动作，通常，命令按钮上面有说明文本，有时是图形文本。对话框通常包括的一些常见命令按钮有 OK（确定）、Cancel（取消）、Help（帮助）按钮。

用户选择命令按钮时，要把指针放在按钮上，按动鼠标左键或轻叩光笔。如果按钮有输入焦点，按空格键也可选择命令按钮。在按下并放开命令按钮后，会调出与按钮有关联的命令。

命令按钮文本应该指示按钮执行的动作。按钮文本应该遵守菜单项使用的大写约定，和菜单项一样，对要求附加信息的命令使用省略号（…）作为可视帮助。如果按钮用于放大辅助窗口来显示附加的选项，在按钮标签的末端会含有一对大于符号（>>）。

Windows 中的所有命令按钮都应当有同样的外观，遵守同样的标准，以便用户可以从外观来区别命令按钮。表 1.5 显示了命令按钮可拥有的不同的可视状态。