Microsoft® Programming Series

ONE 3.5" DISK INCLUDED

Microsoft®

WINDOWS™ 3.1

Developer's Workshop

- Internationalizing Applications
- Source Code and Executable Programs
- Pen Computing
- Virtual Device Drivers

Microsoft PRESS®

北京希望电脑公司

科学出版社

编程经验集

北京希望电脑公司计算机系列丛书

Microsoft 专家
Windows 3.1 编程经验集

Dr. J. 巴特勒 B. 奇费托 J. C. 克雷格
W. S. 霍尔 R. 帕奇 A. 辛哈 著

东阳生 李竹华 译

尤晓东 希望 校

科学出版社

1993

(京)新登字 092 号

内 容 简 介

本书是为具有一定 Windows 编程经验的程序员所写，共分六章，每章讨论 Windows 一个方面的问题。

第一章为 Windows 软件国际化，讨论了为国际市场编写应用程序需要考虑的问题。第二章讲述图形方面的内容，即比例缩放、坐标空间和变换。第三章说明如何编写 Windows 的笔计算程序。第四章讨论网络基本输入输出系统的程序设计。第五章为如何编写虚拟设备驱动程序。第六章讲述作为传统程序设计工具补充的 Visual Basic。

本书对于从事软件开发的程序设计人员和计算机专业的师生、科技人员具有很好的阅读参考价值。

需要本书的用户，可与北京 8721 信箱联系。电话 2562329，邮码 100080。

Dr. John Butler, Bob Chiverton, John Clark Craig
William S. Hall, Ray Patch, Alok Sinha
WINDOWS 3.1
Developer's Workshop
Microsoft Press, 1993

Microsoft 专家
Windows 3.1 编程经验集
Dr. J. 巴特勒 B. 奇费托 J. C. 克雷格
W. S. 霍尔 R. 帕奇 A. 辛哈 著
东阳生 李竹华 译
尤晓东 希望 校
责任编辑 那莉莉
科学出版社出版
北京东黄城根北街 16 号
邮政编码：100717
北京市通县兰空印刷厂印刷
新华书店北京发行所发行 各地新华书店经售
*
1993 年 8 月第 一 版 开本：787×1092 1/16
1993 年 8 月第一次印刷 印张：17 1/4
印数：0—10 000 字数：420 000
ISBN 7-03-003867-3/TP・297
定价：66.00 元

Microsoft Windows 3.1 Developer's Workshop

Dr. John Butler, Bob Chiverton, John Clark Craig
William S. Hall, Ray Patch, and Alok Sinha

译 者 的 话

为了引进国外最新计算机技术，促进我国计算机应用与开发的水平，北京希望电脑公司首次向美国微软公司出版社(Microsoft Press)购买了一批计算机图书的版权。这些图书涉及Windows，Windows for Workgroups、软件开发工具、数据库管理系统、程序设计语言、多媒体技术和标准计算机词典等领域，是目前最新的计算机技术资料，它们不久将与读者见面。

MICROSOFT WINDOWS 3.1 DEVELOPER'S WORKSHOP 一书是为有经验的 Windows 程序员而写，共分六章。第一章由本地化专家 William S. Hall 编写，讨论为国际市场编写应用程序要考虑的重要而复杂的各个方面的问题。第二章是 Microsoft Systems Journal 特约编辑 Bob Chiverton 所写，讲述图形方面的内容。第三章是原 Microsoft Windows 开发者和笔式计算专家 Dr. John Butler 所写，说明如何编写 Windows for Pen Computing 程序。第四章是 Microsoft 开发人员 Alok Sinha 写的，讨论当应用程序需要通过网络进行通讯时，开发人员应该做些什么？第五章由 Microsoft 开发人员 Ray Patch 编写，说明虚拟设备驱动程序(VxD)的功能和特点。最后一章是 The Microsoft Visual Basic Workshop 一书的作者 John Clark Craig 所写，阐述 Visual Basic for Windows 怎样才能成为传统程序设计工具的强有力补充。

在本书的翻译出版过程中，自始至终得到了美国微软公司中国地区总裁杨绍纲先生、微软北京代表处田本和先生和林资山先生的热心指导和帮助，在此谨向他们表示诚挚的谢意。

1993 年 7 月　北京

目　录

引　言

《MICROSOFT WINDOWS 3. 1 DEVELOPER'S WORKSHOP》一书是为有经验的 Windows 程序员而写，这些程序员对下面这些领域有兴趣：国际化、图形、基于笔的程序、网络程序、虚拟设备驱动程序和 Visual Basic 模型。如果我们比较一下本书涉及的内容和未涉及的内容，对读者来说也许会有些用处，虽然这样说起来很绕舌。首先，本书不是 Windows 程序设计的入门读物，Charles Petzold 经典的《Programming Windows》目前已经是第三版了，其中对于如何使非 Windows 程序员成为真正的 Windows 程序员提供了很好的学习过程，而《MICROSOFT WINDOWS 3. 1 DEVELOPER'S WORKSHOP》假定读者已经具备了上面几本书中所述的知识，因此我们慎重地选择了 Petzold 书中所含内容之外的话题。

其次，本书并不完全源于 Richard Wilton 的《Windows 3 Developer's Workshop》一书，但如标题所述，它又与那本书密切相关，可以称为姊妹篇。两书都是"developer's workshop"，因为它们都深入探讨了有经验的程序员所关心的一些话题，目的都是为了建立复杂的 Windows 程序而提供附加的工具。但 Wilton 讨论了调试、定制的控制、DDE 等等，我们把这些归为 Windows 程序设计内容，本书没有使用这种组织原则。我们按作者进行组织，他们都对所写的主题有很丰富的实践经验，因而每篇文章都是独立的。读者在本引言中可以看到没有使用"本书是如何组织的"这样的标题，这样，读者就可以先阅读自己最感兴趣的那篇文章。

最后的结论是：《MICROSOFT WINDOWS 3. 1 DEVELOPER'S WORDSHOP》一书不是包括所有"Windows 高级技术"的大册子。Windows 是一个丰富的、复杂的、易于扩充的操作系统，它能"包容一切"，我们希望在下一本《Developer's Workshop》中继续这一工作。

一、本书内容

本书包括六篇文章，每一篇由不同的开发人员编写。

第一篇文章由本地化专家 William S. Hall 编写，它针对为国际市场编写应用程序要考虑的重要而复杂的各个方面。许多源于美国的软件产品，现在从海外的收入已经超过总收入的一半；Windows 正在迅速地成为一种国际标准；而且 Windows 在设计和实现本地化产品方面给开发人员提供了强大的支持。Hall 告诉读者如何在这些地区使用 Windows 的功能，以及如何了解和避免各种困难。

第二篇文章是《Microsoft Systems Journal》特约编辑 Bob Chiverton 所写，讨论图形方面的内容。特别是，Chiverton 深入 GDI 设备变换，说明了坐标变换、比例变换、缩放、旋转和平移。该文还解释了 Win32 变换如何扩充 16 位的 Windows GDI。

第三篇文章是原来的 Microsoft Windows 开发者和笔式计算专家 Dr. John butler 所写，说明如何编写利用 Windows for Pen Computing 的程序。Butler 细致地描述了如何修改一个简单而典型的 Window 程序，使之完全能使用笔的过程。在整个过程中，他讨论了程序设计中的要点，并说明了操作系统提供的工具，以使得该过程比较容易理解。他在许多情况下说明了笔比键盘更好用，甚至比鼠标更好用。

第四篇文章是 Microsoft 开发人员 Alok Sinha 写的，它说明了这样一个问题：当应用程序需要通过网络进行通讯时，开发人员该做些什么？Sinha 介绍了在 Windows 中使用广泛支持的 NetBIOS 协议，把内容分成 NetBIOS 控制块、同步和异步命令以及内存映象。其中包含一个示范程序，对所有内容作了实际说明。

第五篇文章由 Microsoft 开发人员 Ray Patch 编写，说明虚拟设备驱动程序(VxD)的功能和特点，对只在小地方使用的程序有时是必要的。Patch 详细介绍了 VxD 的机制，说明怎样编写 VxD 服务(VxD 为另外的虚拟设备驱动程序提供服务)，并解释了如何从 VxD 访问基于 MS-DOS 的程序和 TSR。他建立的程序可作为实际 VxD 的一个工作模板。

最后一篇文章是《The Microsoft Visual Basic Workshop》的作者 John Clark Craig 所写，说明 Visual Basic for Windows 怎样才能成为传统程序设计工具的强有力补充。Craig 消除了认为 Visual Basic 只是功能不太强的写作工具的观点；他说明可以很容易地建立界面、模型化 Big App、访问 DLL 中可用的函数、以及增强现有的应用程序。

二、本书附带磁盘上的内容

本书随带了一片 3.5 英寸、1.44 MB 的磁盘，其中包含了本书中使用的所有源代码和可执行(EXE)文件，程序存放在与书中文章相对应的子目录中。磁盘上的全部程序已经使用 Microsoft C/C++编译过，它们也可以使用 Microsfot C 6 版进行编译。所有程序都是很有用的。

Dean Holmes
Acquisitions Manager
Microsoft Press

第一章　国际化 Windows 软件

1.1 简　介

1.1.1 一个可怕的事实

让我们来考虑一下下述情况：你作为一个管理 Microsoft Windows 操作系统某一项工程的软件开发人员，致力于为自己的应用程序扩展本国之外地区的市场。根据以往对于 Windows 的训练，你知道应该将字符串放入资源文件中，以便将产品进行国际化。由于没有真正理解这种方法的含义，你尽职地把大多数的字符串放入资源文件中，同时建议自己的同事也这样做。然而，随着产品出口压力的增加，越来越多的字符串继续嵌入到源代码中。你有时合理地认为其中的某些字符串只是用作“系统串”，但仍然继续极度节省空间地进行串的处理，编写精巧的程序，利用合并操作来重新使用字符串，设计仅够容纳所用字符串的串缓冲（必须将程序放入 4KB ROM 中的时代已经过去）。

你仍然像以前一样编写代码，使用 C 库中原来的函数来转换字符串中的大小写，进行比较，以及显示日期和时间，对于如何显示浮点数做隐含的假设。（并不是每个人都使用十进制数吧？）你将从编辑控制框中取得的串直接传递给 MS-DOS，使用逗号分隔各个域来输出数据。而开发系统中这一部分的同伴讨厌 Windows 的对话框编辑器（这样就产生了更大的冲突——至少在旧的版本中是这样的）。你通过手工方式精心地制作出大量的控制，控制的标识符放在包含函数原型、全局变量和其他定义的头文件中，对话框模板加入到包含字符串和菜单的资源文件中。具有这种艺术爱好的开发人员忙于设计包含正文的漂亮图标和位图。

最后基本产品出来了，此时需要将该产品进行翻译。当翻译人员开始处理资源文件时，他发现有些信息段的含义无法根据上下文来确定，因而请求帮助（还记得前面提到过的精巧的串处理程序吗），这就需要进行解释。例如，说明字符串“was not found”用于跟文件名相结合，开发人员已经忘记了还使用该串来完成其他的任务。翻译人员按照语法特性与文件名一起翻译这个串，而遗漏了动词的含义。

终于，翻译工作完成了，接着编译这些资源，并加入到程序中。但执行该程序时，程序却立即崩溃了。经过一些分析工作之后，我们发现翻译后的许多信息比程序中假定的长度要长，从而超过了固定的缓冲区大小。经过很长时间的调试，程序终于能够运行了。菜单部分被正确地翻译，看起来也很好，但充其量也只是看着还不错，时间和日期的显示格式在使用地区说来是不能接受的，某些信息的显示中还包含了两种语言（因为你忘记了一些嵌入的字符串）；动词和题目不与人和数一致，形容词和名词具有不同的词性和数，消息中包含特殊的单词顺序；图标中显示未翻译的正文，有一些则包含了本国人不喜欢或不能理解的图形；对话框显得太拥挤，难以使用，某些域就被截掉了；对于该国语言来说数据的分类也不正确；重音字母上大小写之间的映射不成功；带分隔符显示当前值的方式与本地用户期望的方式相

反;输出的数据不能被移入为目标语言编写的软件中,因为固定的域分隔符(逗号)作为数的整体和小数部分的分隔符是错误的。

最后,你认识到为了满足翻译版本的要求,不得不建立两套不同的源程序——一套用于原始语言,另一套用于新的语言。这样便使得费用和代码的维护困难加倍。开发另一个国家语言版本的计划只能暂停,直到原始产品完全得到更新。另外,还需要安排更多的质量保证周期,这影响其他的计划,也影响许多其他的方面。因此,你下决心不能再在下一个产品中重复这些错误了。

1.1.2 为什么要国际化

为什么还要考虑产品的翻译呢?可以用下面的一句话来进行回答:"货币上都是有文字的,但并不总是印有同一种语言的文字"。换句话说,通过发行应用程序的不同语种版本,可以极大地拓展潜在的市场区域。现在有许多软件公司发现其收入的很大一个部分来自国际市场,这意味着每个人薪水的相当一部分是由不说同一母语的人们支付的。至少应该向用户提供满足他们的文化标准和语言需要的产品,以表示最起码的尊敬和礼貌。

那么如何来理解看似复杂的产品开发,而使之又能适应语言和文化的惊人变化呢?最为重要的,怎样以最有效的方式达到这一目标呢?在本文中,我希望能够为比较轻松地进入国际市场提供背景知识。读者能学到字符集、键盘、适合本国国情的内容以及 Windows 和 MS-DOS 间的相互作用。读者会发现用来编写代码的工具与语言和地区无关,还要学习使用户接口独立的必要性。最后,读者会发现完成必要的国际化恰恰是高效软件开发工作的另一个部分,即使读者从来没有进行过翻译到另一种语言的工作,读者还会发现自己已经开发了一个易于扩充和维护的产品。

1.1.3 几个术语

先让我们统一一下本文中使用的几个术语的含义:"允许国际化"(enabling),"本地化"(localizing)和"改型"(retrofitting)。假设读者刚刚完成了一个 Windows 程序的最初版本,为了便于讨论,我们假定开发者位于美国,他的第一个版本是致力于美国的市场,我们称该版本为"原始产品"。

现在提出以下的问题:该产品的设计和编码是否能够使之打入国际市场?更具体地说,如果将该程序翻译为另一种语言,是否只需花最小的费用就可完成,而不影响原始产品?如果是这样,我们就可以说该产品具备了对国际语言的支持能力,我们称它为是**允许国际化**的。

如果产品是允许国际化的,则可以将它**本地化**。本地化了的产品使用某个国家的语言、习俗和文化。完成本地化的工作应该能够不修改除资源文件和帮助文件外的任何一行源代码。反过来说,如果我们发现需要重写某些源代码,则就是作了**改型**。改型工作的费用是极为昂贵的。如果需要改型,则设计的源程序代码不是与地区和语言无关的。当然,如果读者是第一次进行该工作,就甚至会不知道自己是如何产生这种错误的,希望后面的章节能引起读者的注意。

1.1.4 为什么程序未被允许国际化

通常程序的第一个版本是不适于国际化的，结果发生了前面所说的可怕情况。软件管理员和程序员都致力于建立允许国际化的产品，但需要就完成日期和开发过程进行权衡。将一个字符串嵌入代码中似乎比从资源文件装入要容易得多(当然后者更加稳定，是不是)，编写例程来读取和使用 WIN. INI 中国际选项的日期和时间格式要比使用 C 的库函数要麻烦得多，将正文串看成可能是包含双字节的亚洲语言文字看起来是很麻烦的。

在某些情况下，人们只是没有注意到程序的哪些部分需要引起注意。字库名看起来像是系统信息，因而将该串留在源代码中比从资源文件中获取会有什么坏处呢?但如果是为日本的市场编写代码，读者就会发现如果不修改代码来改变字库名的话，就不能显示出字符。

在其他情况中，读者可能会觉得不需要允许国际化的代码。例如，一些公司开发人员生产实用程序产品，该软件可以没有用户界面，而只是一些工具集合，但可能要求程序不能被允许国际化。有一个真实的故事，不久以前，一家为 Windows 开发产品的公司决定将一个用于字符串处理的库投放市场，其中的许多例程用汇编语言编写，它补充了 Windows API 中的很多欠缺之处，还弥补了 C 库中缺乏使用长指针函数的不足，这似乎正是程序员梦寐以求的东西。但这家公司忘记了该库完全依赖于 Windows 的语言和字符集。可以看出，使用该库无法根据字符集的代码点顺序将正文排序，无法不考虑具体的语言习惯来处理字符。任何人只要使用该公司的库来开发 Windows 应用程序，就立即使自己的产品失去了国际化的可能性。

程序未被允许国际化的其他原因是各种各样语言唯我独尊的通病："如果想用我们的产品，就来学习英语(或法语、日语)"。或者是有些人相信只有"专家"(莫非也只说英语吗)才会使用程序，这是那些开发专用或纵向应用程序的人们易犯的错误。然而，由于在 Windows 中能简单地建立和使用程序，因此，自动地拓展了用户。以前只由专家使用的程序在 Windows 中可以由一个没有经验的职员，甚至是不能对原始语言进行翻译的销售人员来使用。

我们不能忽视对于"洋东西"的个人偏见，因为它会妨碍软件开发的国际化。许多很有水平的开发人员通常很少有欲望进行国际化或学习其他的语言，他们常常因为忽视这些方面而陷入困境，认为一旦自己的产品本地化就会失去对它的控制。但只要这些开发人员意识到自己受这种感觉的影响，是很容易克服的。使用一种自己不熟悉语言版本的 Windows 就会有信心，他会惊异地发现能够毫无困难地使用 Windows 的芬兰语或日语版本，即使没有这些语言的任何知识。当然有时候键盘的使用会难住他，而且不能识别某些字母的重音，但不需太长的时间就能掌握这些重音，而且能以相当快的速度自学日语写作系统的基础知识。

本文可以帮助读者决定编写出与语言和地区无关的内部代码必须经过的步骤，经过这些努力之后，就可以充满信心地编写出易于本地化的软件，编写允许国际化的代码将会成为另一种程序设计技巧。

1.1.5 允许国际化、本地化和改型的代价

大量的允许国际化规则只是好的软件工程实践。因而，即使不进行本地化，也能在代码的开发和维护方面受益匪浅。但是程序员和管理员仍然必须学习允许国际化的原理，它在编程的时间和复杂性上不是没有代价的。编写的代码必须提供对与语言无关特性的访问，而且

用户接口必须独立，以便于翻译。

但允许国际化工作只是一次性花费，一旦完成之后，该产品就可被翻译为其他语言，而不影响原始产品。允许国际化的代价可以分散为需要建立的每一个本地化版本。

允许国际化之后，则本地化的主要代价变为一次翻译工作。必须记住不仅要翻译软件，还必须翻译在线帮助、文档和包装(假定已经考虑到提供国际发行，且计划给海外用户提供支持)。通常将一个允许国际化的产品软件进行正确翻译是本地化过程中最容易和最快的工作。

最后，还要记住从产品开发的开始就使软件允许国际化是更容易和更廉价的。每次重新看没有被允许国际化的程序时，就不得不修改工作代码。在效果上就是建立了一个需要另一质量保证周期的新版本产品。这不仅影响了国家语言产品，还提高了原始版本的开发费用。从欧洲语言转换为亚洲语言的修改是很麻烦的，所以事先就计划好比以后再来修改要更好(大家都知道这一点，但就是不这样做)。

1.1.6 学会国际化游戏

能够在 Microsoft Windows 环境下编写软件是很幸运的，因为它给编写允许国际化的软件提供了一个理想的平台。它有内嵌的工具来以资源的形式独立用户接口；一个帮助机制可用于根据任何帮助文件显示信息，而不管该文件使用哪种语言；该环境提供了与国家相关的信息；字符集是多语种的；键盘外型支持许多国家；可以使用与语言相关的功能；手边的转换例程可用于 MS-DOS 与 Windows 之间的相互交互。

当然，需要学习如何使用这些工具。跟通常学习一种程序设计技巧一样，同时需要学习理论和实践，在后面的内容中，两者交织在一起。首先让我们来仔细看一下字符集。字符集是理解国际化的基础，因为它们完全定义了在一个特定环境中能支持的语言和地区。然后再来看支持键盘的理论和实践，与语言无关的语言支持方式以及国家信息的特征。文中还需要常常提及 MS-DOS，因为它是运行 Windows 的底层平台，国际化的许多内容是从 MS-DOS 继承而来的，而且两个系统相互交互的许多方法影响国际化支持。文章结束时讨论处理用户接口，描述了一个库，该程序包含在本书随带的磁盘上。最后有一些参考文献可以帮助读者增加对本领域软件工程的理解，还可以帮助找到有关语言和文化的信息。

没有任何东西能替代经验，马克·吐温简明地概括了这一点：将猫提着尾巴抓回家的人，他的经验比光是听说此事的人要多出六七十倍。本文最后的文章和参考文献能够减少无知的因素，但并不能消除用户第一次国际化一个产品将遇到的所有问题。

最后提一下本文的方向：在说明国际化时，本身的表示应独立于语言和文化背景。当然，由于 Windows 在欧洲和美国的广泛使用，以及我本人的语言和文化背景，可能会导致偏向这些方向，但我试图涉及亚洲语言，主要是日语，而且也没有忽视将 Windows 推广到中欧和东欧地区。但是对于从右向左语言的有趣部分没有加以讨论，也没有考虑对 Unicode 的影响。Unicode 是新的 16 位字符编码标准，它在 Windows 的将来版本，即著名的 Windows NT 中使用。

1.2 字符集入门

1.2.1 字符集

一个字符集是符号的有序排列。字符集中一个给定符号的位置称为它的代码点。不要把字符集的概念与字模的概念混淆，一种字模是一个字符集，具有给定大小和字型，按照一种适合于屏幕上显示或适合于输出设备的格式使用。因而，一个字符集是一种字模中与字型一样的一个属性。相同的字型可用于任意的字符集，同一字符集可以按许多字型出现。

要理解一个特定的系统怎样处理国际化，开始时应先研究一下它所支持的字符集。它们决定了输入设备（例如键盘和笔）如何处理输出、可以被显示和打印的字模，以及可以实现本地化的国家和地区。

在很久以前（大约是 1985 年），Windows 只有一个字符集。本质上等同于国际标准化组织（ISO）8859-1，Windows 使用的字符集能提供对许多欧洲语言的支持。此后 Windows 的原始字符集被丰富了另外的符号，并且新的字符集被引入到销往世界上其他区域的 Windows 中。如今的 Windows 版本支持美洲、东欧、西欧、前苏联的斯拉夫语、土耳其语、阿拉伯语、Hebrew、中文、朝鲜语、日语，等等。

当然，用户希望自己的软件利用这些市场上的机会，同时又需要避免为每种不同的语言建立独立的软件版本。因此有一项任务是编写用户程序，以使之独立于基本的字符集。例如，如果不使用 Windows API 的工具映射小写和大写字母或字间隔，则用户程序（按照规则翻译后即可正确运行于俄语版、土耳其语版、捷克语版，甚至日语版本下）就需要做一些麻烦且可能费用昂贵的修改。

为了知道和理解规则及其必要性，需要为个人计算机上使用的许多种书写系统作一个评价。为帮助获得这种评价，本文在一些细节上检查了 Windows 和 MS-DOS 中存在的几个字符集。读者会很快发现自己的母语极大地影响着对于语言各方面的看法，还会发现为了编写与语言无关的软件，需要抛弃对于语言的许多假设，转而使用 Windows 环境下可用的工具。

作为开始的第一步，先来看一下在 MS-DOS 中使用的一些字符集。两个操作系统以许多有趣的方式相互作用。Windows 使用 MS-DOS 文件系统，MS-DOS 程序能运行在 Windows 中，甚至运行在一个窗口中。Windows 应用程序能够移入 MS-DOS 程序创建的数据。除了通常这两个系统使用极为不同的字符集外，这些相互作用只是传递注释。所以这两个系统之间的约定是必要的，特别是对 Windows 的表示层有影响。

1.2.2 MS-DOS 中的字符集

读者是否知道在 MS-DOS 版本 5 的 U.S. 版中，可以处理 Czech, Danish, Dutch, English, Finnish, French, German, Hungarian, Italian, Norwegian, Polish, Portuguese, Serbo _ Croatian, Slovak, Slovenian, Spanish, Swedish 及更多语言中的正文？当然如果没有能够输出那些语言中特殊字符的打印机，则只能在屏幕上看到所做的工作。在给拼写检查程序装入新的词表之前，该检查程序的使用可能是很令人厌烦的，至少存在这种可能性。

MS-DOS 通过使用许多字符集来获得多语种能力，每一种称为一个代码页。每种给定代