Pen Services for Microsoft® Windows® 95

程序员指南

Microsoft Corporation 著
黄昱 尹娟 译
赵玫 审校

清华大学出版社

目　录

引　言

本书描述如何创建使用 Microsoft® Windows® Pen Application Programming Interface (API)的应用程序。本书分成两个部分。第 1 部分给出有关笔的计算的概述，并描述了 Pen API 的各成分。实例代码对文本进行了补充说明；在后面的章节中还给出了一个完整的实例程序，即实例 recognizer(识别器)(识别器能够把笔笔画转换成字符、符号或图象)。第 2 部分是构成 Pen API 的有关函数、结构、消息和常量的参考手册。在参考手册后，还给出了一些附录，它们提供有关 Pen API 1.0 版与 2.0 版(32 位的笔的服务程序)间的区别的信息，以及其它信息。

Microsoft Windows 95 操作系统中包含了 Pen API 的一个子集，以显示有关的笔数据。这就允许一个基于笔的应用程序，即使在没有专门的笔硬件的环境下，也可以从笔的图形输入板收集笔数据，存储数据，以及在运行 Windows 95 的个人计算机上显示数据。只有在装有由笔的设备的初始设备制造厂家(OEMS)制造的笔硬件的环境下，才会提供全部笔的辅助工具。也就是说，Windows 95 中运行的应用程序至少可以有保证地访问 Pen API 的显示部分；如果还配备有笔的图形输入板，应用程序还能接收笔输入。

本书中所描述的由 Pen API 2.0 版提供的全部光笔**服务**只能在 Windows 95 或更高的 Windows 版本上运行。

本书假定你对 C 语言已比较熟悉，并且对 Windows 编程有一定了解。为了使讨论简洁，本书在正文中不会脱离主题去定义诸如动态链接库(DLL)、回调函数及消息这样的一般性术语。但是，基于笔的计算有一些专用词汇，因此当出现该类词汇时，将在正文中定义这些专供 Pen API 使用的新术语。此外，在本书最后还给出了一个简短的词汇表，其中包括了所有基于笔的计算的专用术语。

组织结构

本书分为下列各章和各附录。

章或附录	说明
第 1 章，笔的应用程序编程接口概述	Pen API 的基本结构。
第 2 章，使用系统默认值启动	如何用最小的编程代价来实现向应用程序加入笔功能。
第 3 章，书写过程	应用程序如何从笔设备获得输入。
第 4 章，上墨过程	应用程序如何收集和修改笔输入数据。
第 5 章，识别过程	将原始笔输入转换成可用字符，如字母或数字。
第 6 章，设计考虑	编写基于笔的应用程序的正确方法、告诫和有关的小窍门。
第 7 章，笔的应用程序的一个实例	PENAPP.C 应用程序实例，解释说明了第 1 章到第 6 章的有关内容。

续表

章或附录	说明
第 8 章，编写识别器	识别器的要求和设计。并用识别器的一个实例 SREC.C 作为模型来说明有关内容。
第 9 章，笔的应用程序编程接口概述	分类列示 Pen API 辅助工具。
第 10 章，笔的应用程序编程接口函数	按字母顺序列示有关函数。
第 11 章，笔的应用程序编程接口结构	按字母顺序列示有关结构。
第 12 章，笔的应用程序编程接口消息	按字母顺序列示有关消息。
第 13 章，笔的应用程序编程接口常量	按字母顺序列示有关常量。
附录 A，1.0 版和 2.0 版笔的应用程序编程接口的区别	对 Pen API 所作的改进。
附录 B，使用 32 位的笔应用程序编程接口。	如何编写 32 位的基于笔的应用程序。
附录 C，修改 SYSTEM.INI 文件	Windows SYSTEM.INI 中采用的有关设置。
附录 D，访问笔的设备驱动程序	应用程序如何调用笔的驱动程序。
词汇表	有关基于笔的术语。

文档约定

本书从开始到结尾都将用到下列文档约定。

约定	说明
粗体文本	黑体字母用来标示一个特定术语，或作为按原形式使用的词语（函数名、关键字、MS-DOS® 命令及命令行选项）的强调标志（例如 **DrawPenData Ex** 或 **Switch**）。必须按所显示的方式输入这些词语和强调标志。大小写字母的用法与平常一样，但不再具有不同含义。例如，既可以在 MS-DOS 提示符后输入 CL、cl，也可以输入 Cl 来激活 C 编译器。
()	在语法叙述中，括号用来把你要传送给某个函数的一个或多个参数括起来。
斜体文本	斜体用来标示一个位置标识符；由你为之提供一个真实值。例如，在下面的语法中，位置标识符 *LpszRecogName* 用来代表一个指向识别器的文件名的指针：**InstallRecognizer**(*lpszRecogName*)；与基于光笔的计算有关的新词汇，当其第一次在文本中出现或定义时，也用斜体来标示。这些词将列示在最后面的词汇表中。
等宽文本	代码实例以非对称字体显示。

if (! RegisterClass (LPWNDCLASS) &wc))
⋮
else
	程序实例中的垂直省略号标示该程序的一部分被省略掉。

续表

约定	说明
…	在某个项后面跟随一个水平省略号标示还可能会出现更多的相同形式的项。
[[]]	两个中括号用于在命令行或语法中把选择性区域或参数括起来。
|	竖杆标示你可以输入该竖杆两侧所显示的任意一个项。在符号图象中，竖标标示可能选择的字符。
{}	大括号标示你必须说明其中的某个项。
小的大写字母	小的大写字母用来标示键或键序列的名字；例如 CTRL+ALT+DEL。如果键名间用逗号而不是加号分开；例如，ALT，F——那么你必须依次按下这些键，而不能同时按下。

可参阅的其它著作和文章

下表所列示的文献提供了有关 Pen API 和有关 Windows 的一般性附加信息。

标题	内容
Microsoft Windows Pen API 2.0 版联机帮助	Pen API 函数、结构、消息和常量的联机参考手册。
Microsoft Windows 软件开发工具(SDK)文献，或相当的文献	有关 Windows 操作系统应用程序编程接口的信息。
Microsoft Windows 设备驱动工具(DDK)文献，或相当的文献	描述了 Windows 设备驱动器的应用程序编程接口。仅当要用到驱动器时才需要。
Duncan，Ray. "Power Programming". PC Magazine. New York，New York：Ziff-Davis 出版公司，1992 年 1 月 14 日—1992 年 5 月 12 日	Pen API 1.0 版基础上的一系列文章。
Petzold，Charles. Programming Windows。第 3 版。Redmond，Washington：Microsoft Press，1992	对 Windows 的一般编程进行了较好的介绍。

系统要求

你可以在下列软件和硬件环境下编写 Pen API 2.0 版的应用程序：

- 一台运行有 Windows 95 或更高版本的个人计算机
- 一个鼠标器，一张图形输入板，或其它 Pen API 支持的指点器
- Microsoft Win32®软件开发工具(SDK)
- Microsoft Windows 95 设备驱动程序工具(DDK)——仅当要建立笔、显示器或键盘驱动器时才需要

- Microsoft C 优化编译器 5.1 版或更高版本，或者 Microsoft Quick C® for Windows 1.0 版或更高版本
- Microsoft 宏汇编器 5.1 版或更高版本——仅当要建立笔、显示器或键盘驱动器时才需要

你还可以使用由其他生产商，如 Borland International，Inc，生产的开发软件。

致谢

在此对曾为本书作出过贡献的人表示特别的感谢，他们有：

作者	Beck Zaratian	Mark Williams
	Don Gilbert	
编辑	David Steinmetz	Barb Ellsworth
	Peter Delaney	David Thornbrugh
程序管理员	Jeff Aamodt	Steve Liffick
	Eric Berman	
笔服务开发组	Eric Onasick	Shishir Pardikar
	Vinayak Bhalerao	Chris Leyerle
	Haresh Ved	Fumitaka Kawasaki
识别器开发组	Mike Van Kleeck	Jim Adcock
	Justin Ferrari	Shamik Basu
	Sung Rhee	Oswaldo Ribas
	Donald Sidoroff	Patrick Haluptzok
	Greg Hullender	
测试小组	Becca Moss	Keith Stutler
	Brian Watson	Randy Shedden
	Xian-Ling Wu	David Flenniken

第 1 章　笔的应用程序编程接口概述

本章给出基于笔的计算的概述，分为两个主要部分。第 1 部分广泛描述了构成笔的应用程序编程接口(API)的各种成分。第 2 部分描述应用程序是如何去访问笔辅助工具来具体实现各种基于笔的特性的。

Pen API 2.0 版的基本结构与 1.0 版仍大致相同，但其风格和设计则有很大变动。即使你曾使用过 1.0 版，你也应仔细阅读本章，以了解 2.0 版中在编程宗旨上的变化。

1.1　Pen API 的结构

表面上看起来十分简单的把数据从笔取到应用程序的步骤实际上包含了许多中间步骤。幸运的是，Pen API 自己承担了其中的大部分工作。通过向应用程序提供访问笔特性的便利途径，Pen API 把程序员从最枯燥无味的光笔数据识别中解放出来。同时，其灵活的设计也允许应用程序控制大多数低级的笔输入。

当阅读这一节时，请记住：Pen API 结构上的复杂性并不意味着在创建基于笔的程序时也存在相应的困难。你将会发现编写一个聪明的基于笔的软件并不比编写其他 Microsoft Windows 应用程序困难。

图 1.1 解释说明了应用程序与 Pen API 的主要成分间的相互作用关系。

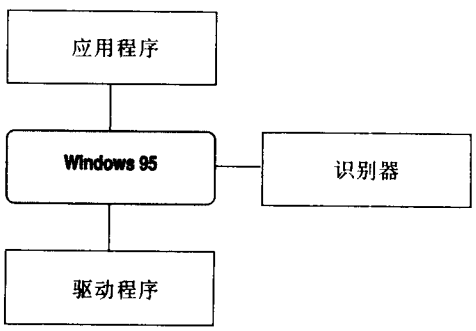

图 1.1　Pen API 的成分

下面 4 节描述图 1.1 中的各成分，其中从主 Windows 成分开始。每一节中都包括一个与图 1.1 相结合的图，提供相应成分的详细分解图示。伴随的正文则描述该成分，并解释它是怎样与其他成分一起互相作用的。

1.1.1　Windows

如图 1.2 所示，Microsoft Windows 95 的基于笔的辅助工具的核心由两个库构

成——PKPD. DLL 和 PENWIN. DLL。PKPD. DLL 文件用于 Windows 95 光笔服务的墨数据管理。这就使得应用程序在任意一种 Windows 95 安装方式下，都能显示和管理墨数据，甚至在未安装笔硬件的情况下也能如此。在第 9 章"笔的应用程序编程接口概述"中确定了由 PKPD. DLL 提供的有关笔的辅助工具。

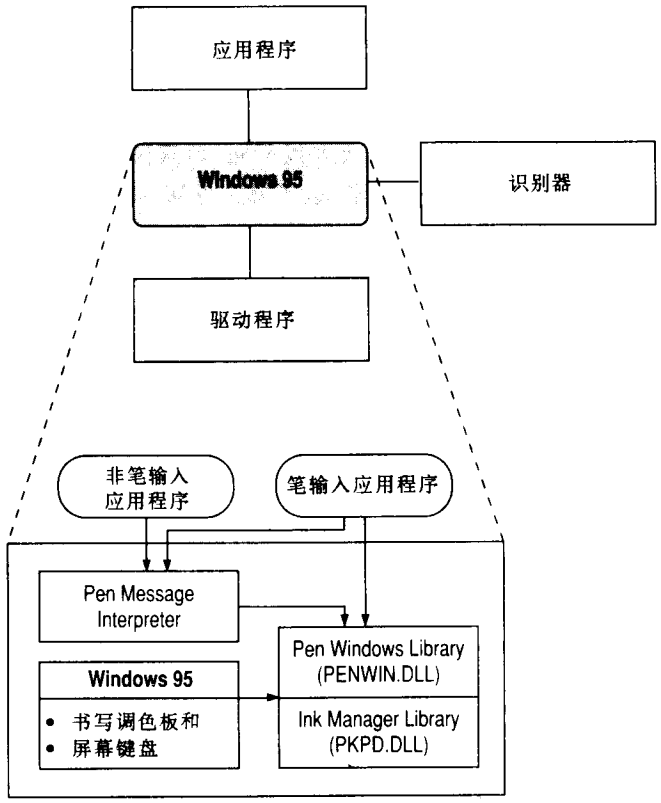

图 1.2 Windows 成分的详细图示

仅当配备有由初始设备制造厂家(OEM)生产的光笔硬件时，才可使用 PENWIN. DLL 文件，它提供诸如收集、修改和识别墨数据之类的附加笔的辅助工具。在使用输入和识别特性前，应用程序首先应当检测是否有 PENWIN. INI 文件，然后再决定是退出执行状态，还是根据情况调整其相应性能。

在 Pen Windows 1. 0 版中，如果应用程序想通知系统把所有编辑控制转换为手写输入的编辑(hedit)控制，它需要调用 **Register Pen App**。而在 Pen API 2. 0 版中，则不必如此；应用程序中的所有编辑控制都被自动转换为手写输入的编辑控制。如果是诸如基于 Windows 95 的应用程序这样的标明版本的应用程序，转换自动完成；而如果是诸如基于 Windows 3. 1 的应用程序这样的标明版本的应用程序，与 Pen Windows 1. 0 版一样，它也需要调用 **Register Pen App** 来完成转换。

重要的是，一定要了解，任何应用程序要想成功地使用 PENWIN. DLL 中的函数，所在的计算机都必须在启动 Windows 时取得相应的笔的辅助工具，并在关闭 Windows 时

结束相应的笔的辅助工具(也就是说,PENWIN. DLL 必须在 SYSTEM. INI 文件的[Boot]部分中的驱动程序行中注明)。对 PKPD. DLL 库中的函数则不必如此,该库在 Windows 95 系统中可以被自动地选用。要了解有关 SYSTEM. INI 文件要求的信息,请参见附录 C"修改 SYSTEM. INI 文件"。

由于该要求,PENWIN. DLL 永远也不会由应用程序静态链接到未安装笔的辅助工具系统上。因此,它的函数必须用函数指针来进行调用。典型的例子是,当初始化笔输入应用程序时,将调用带有 **SM_PENWINDOWS** 参数的 **GetSystemMetrics** 函数,若成功返回,就提供了一个到要使用的库的句柄。对于该应用程序将用到的各个 PENWIN. DLL 函数,应用程序还必须在之后调用 **GetProcAddress** 函数(带有库的句柄和函数名作为参数),并把返回的函数指针保存起来以便在以后需要调用该函数时使用。要了解该方法的例子,可参见 HFORM 应用程序实例。

如果不链接 PENWIN. LIB,就不能确认当一个应用程序在未安装 PENWIN. DLL,但在其路径中包含有 PENWIN. DLL 的系统上运行时,是否能取到 PENWIN. DLL。在系统启动时未取得的 Pen 成分不能保证正确地运行。注意这同时适用于 16 位和 32 位的库。

被指定只在装有笔的辅助工具系统上运行的应用程序可以直接链接到 PENWIN. LIB。这类应用程序应当在启动时检测是否提供了笔的辅助工具;若发现没有提供笔的辅助工具,应立刻退出执行状态。应注意的是,为了便于阅读,本书中的大部分例子采用的是通常的链接方式,而没有采用更安全的函数指针方式。将由开发者自己来决定如何为每一个应用程序选择访问 PENWIN. DLL 函数的最佳途径。

Pen Message Interpreter 向非笔输入应用程序提供基本的笔的辅助工具。该类应有程序虽然没有直接利用 Windows 笔的辅助工具,但目前它们代表了基于 Windows 的软件的主流。Message Interpreter 允许通过截取手写输入和其他笔事件并把它们转换成相应的键盘和鼠标消息以便在非笔输入程序中使用笔。应用程序本身并不知道笔的存在和所发生的笔输入事件。

在截取手写输入的过程中,仅当 Message Interpreter 检测到在非笔输入应用程序中有　型指针或插入点时,它才起作用。由于应用程序一般只在提示你在书写区中输入时才显示　型指针,因此 Message Interpreter 可以为大多数非笔输入程序提供可靠的服务。然而有少数基于 Windows 的非笔输入应用程序不使用标准的　型指针来进行提示,从而会导致 Interpreter 的检测方法失效。虽然 Interpreter 仍旧允许笔在该类应用程序中当作鼠标来使用,但是它已不能解释手写输入。

当 Message Interpreter 为 Windows 3.1 版以前的版本所开发的应用程序提供服务时,它也有可能失效。这些应用程序在设计时没有考虑笔,因此不可能很好地与笔一起配合工作。例如,为 Windows 3.0 版所编写的应用程序的编辑区域常常显得太小而无法用笔在其中进行书写。老的应用程序存在的最后一个问题是,Message Interpreter 无法从应用程序中获得有关的上下文信息,以确定当前所期待的是何种输入。这会降低识别的准确率。

Message Interpreter 对于程序员来说,只是他们学术兴趣所在,这是因为它只与非笔输入应用程序有关。本书的其余部分主要集中讨论如何编写笔输入应用程序和直接使用

Pen API 的动态链接库(DLLs)。

1.1.2 驱动程序

图 1.3 中显示了两种在 Pen API 系统中起作用的驱动程序。大多数驱动程序都符合下面两种模式：使用 Windows 可安装驱动程序接口的可安装设备驱动程序和控制与硬件进行相互作用的虚拟设备驱动程序。

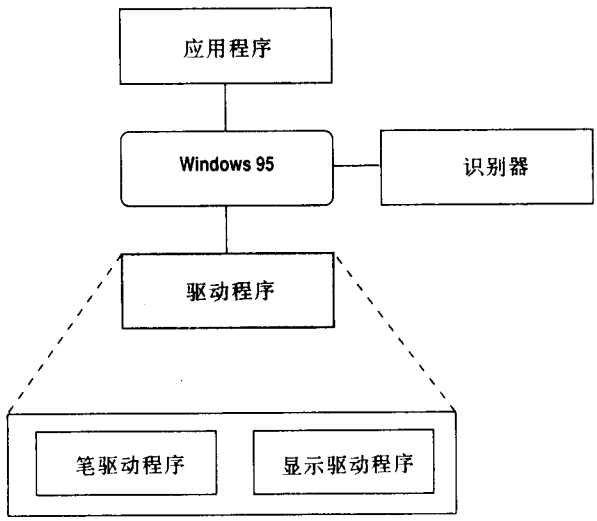

图 1.3 驱动程序成分的详细图示

笔的驱动程序

笔可安装设备驱动程序(Windows 提供的是文件 PENC.DRV。)与虚拟光笔驱动程序(VPENDC.VXD)相互作用，并把笔的移动数据传送给 Windows。由于有时候可能需要对潦草的笔迹进行识别，因此在此对笔输入设备作如下限制：

- 笔驱动程序每秒必须至少能对笔的位置进行 60 次报告。速度将决定所报告的笔的轨迹是否已足够精确，以便支持基于矢量的识别器进行识别。它还将使得墨迹(根据笔的移动而显示出来的一条点象素的轨迹)以通常的书写速度自然而平滑地出现在屏幕上。要了解有关识别器的更多的信息，请参见本章后面的“识别器”。
- 光笔驱动程序在报告笔位置时必须能在每英寸间报告至少 200 个点。这将确定墨迹是否足够清晰，以便让识别器对笔在数字化的屏幕表面的轨迹做出准确的判断。
- 无论设备的分辨率如何，笔设备所报告的图形输入板上的笔位置的坐标的最小尺寸必须达到 0.001 英寸。该约定将确保 Windows 识别器模块和应用程序所看到的墨迹具有同样的度量单位。

显示驱动程序

显示驱动程序负责与显示硬件和 Microsoft Windows 的图形设备接口(GDI)模块协

调工作。显示驱动程序应当支持墨迹的显示，以便能让用户在笔移动时看到反馈的痕迹。从技术上来说，Pen API 并不需要从显示驱动程序那里获得对墨迹显示的支持。但是，通过让用户看到笔留下的墨迹，系统变得更为实用和方便了。

系统支持两种类型的显示驱动程序：Display Control Interface (DCI)驱动程序(又叫 DCI Providers)和非 DCI 驱动程序(如老的 VGA 驱动程序和 8514 驱动程序)。对 DCI Providers 来说，它能完全支持笔接口，没有额外的要求。

要在非 DCI 驱动程序上支持墨迹显示，显示驱动程序必须满足下列要求：

- 能支持 **GetLPDevice** 函数，以便向 **Windows** 提供一个标识笔硬件的值。
- 能支持 **InkReady** 函数，供 Windows 调用以通知驱动程序笔正在移动中，而且 Windows 已准备好显示墨迹。**InkReady** 必须能在系统中断期间处理调用。
- 提供一个笔形状的指针。

由 Windows——而不是显示驱动程序——显示墨迹。当 Windows 通过 **InkReady** 函数收到通知时，驱动程序将回调 Windows 以绘制出墨迹。

要了解有关显示驱动程序的详细信息，可参考 Windows 95 的设备驱动程序工具(DDK)。

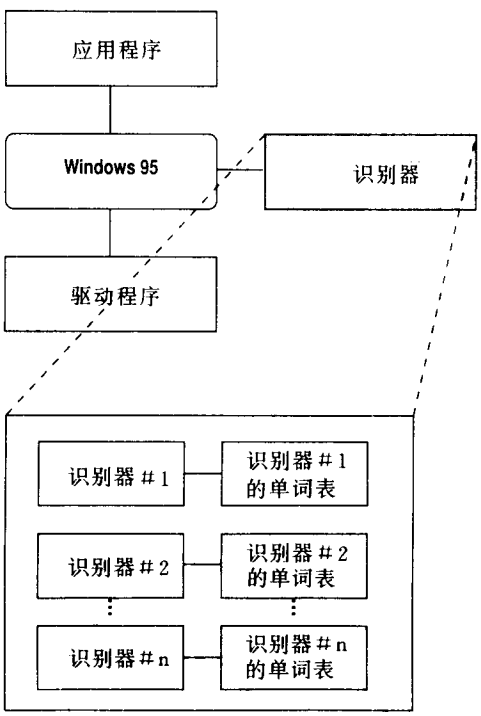

图 1.4　识别器成分的详细图示

识别器

识别器是一个带有许多函数的 DDL 文件。这些函数用来确定一定形状的笔的笔画代

表了什么符号。如图 1.4 中所示，Windows 允许多个识别器并行操作。例如，一台识别器可能专门用于识别英文字母，另一台则用于识别数学符号，还有一台用于识别几何图形，等等。

每一个笔的识别器都能访问任意数据的 word lists(单词表)。单词表向识别器提供了进一步确认其估计正确性的途径。例如，如果识别器不能确认某个手写的单词到底是"boy"还是"looy"，通过在单词表中找到其中一个词语而不是另一个词将有助于识别器作出更可信的选择。

虽然应用程序可选用许多识别器，但只能有一个系统默认的识别器。这就是 Windows 自动安装并调用的默认识别器。要使用其他的识别器，应用程序首先必须专门安装好它们。(要了解怎样安装多个识别器，请参见第 5 章"识别过程"。)在所安装的大多数 Microsoft 笔的辅助工具使用的 OEM 图形输入板中，Microsoft Handwriting Recognizer (GRECO. DLL)被当作默认的系统识别器。Microsoft Handwriting Recognizer 可以识别所有欧洲字母、数字以及标点符号，特别是英文、法文和德文字母。通过在 Windows 注册中引入新文件，应用程序可以设定一个不同的系统识别器。附录 A 中解释了怎样设定一个新的默认识别器。

1.2 从应用程序访问 Pen API

如图 1.5 所示，能接收用户输入的应用程序可划分为两类：笔输入和非笔输入的应

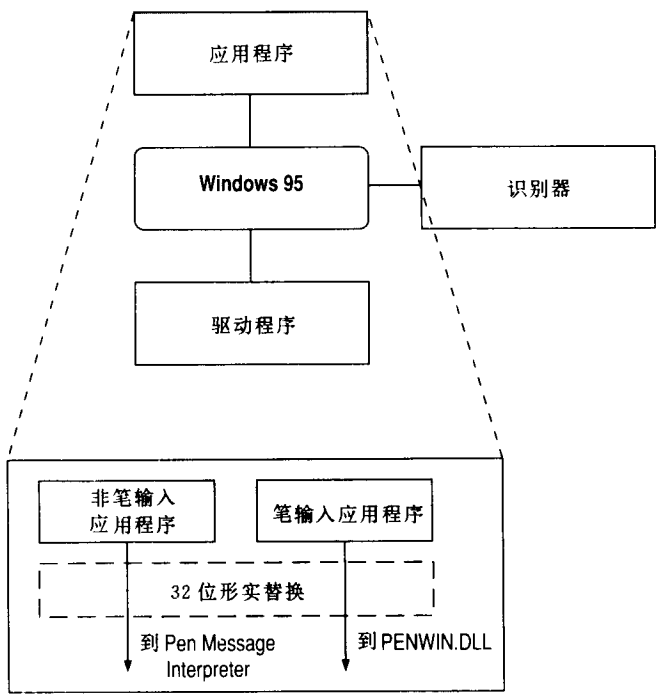

图 1.5 应用程序的详细图示

用程序。非笔输入的应用程序，如其名字所隐喻的那样，在编写时只考虑到接受键盘或鼠标输入，而未能意识到 Windows 笔的辅助工具的存在。但是，如果有笔的辅助工具，在非笔输入的应用程序中，Windows 95 既可以把它当作鼠标使用，也可以用它来进行文本输入。要了解在未考虑接受笔输入的应用程序中 Windows 是怎样使用笔的详细信息，请参见第 2 章“使用系统默认值启动”中的“非笔输入的应用程序”。

Pen API 是被设计为用于小型的手执系统，内存和能力都较为有限，因此其 API 由 16 位的函数构成。因此，Windows 为 32 位的应用程序提供了一个形实替换以调用 API。形实替换自动地将 32 位的函数参数和结构数据转换成 16 位的数据。在调用 Pen API 前，应用程序必须保证其数据适合于较小的数据尺寸。要了解有关使用 32 位 API 的信息，请参见附录 B。

第 2 章　使用系统默认值启动

笔的应用程序编程接口(API)尽可能多地处理基于笔的计算的复杂性。一个较完善的 API 应当让开发者把精力集中在设计上,而不能让他为细节问题担忧。本章描述了怎样依靠系统提供的缺省 Pen API 服务来创建一个基于笔的应用程序。为了简单起见,本章中所使用的术语"应用程序"包括基于 Windows 的程序和动态链接库(DLLs)。

2.1　非笔输入应用程序

Microsoft Windows 95 支持在非笔输入应用程序中使用笔。对于这类应用程序,Windows 提供了一种方法,用笔来模拟鼠标和键盘数据。它通过两种途径来实现这一点。

第一种途径是利用 Pen Message Interpreter,它已经在第 1 章 1.1.1 节"Windows"一节中描述过了。第二种途径包括了两个实用程序"支程序",分别叫做 Writing Palette (WRITEPAL.EXE)和 Screen Keyboard(SK.EXE),它们都作为已安装的应用程序由系统提供。Writing Palette 允许用户在 Message Interpreter 未能检测到输入提示符的情形下输入手写文本。例如,当在某个窗口中运行 MS-DOS 文本编辑器时,用户就可以通过 Writing Palette 实用程序来输入手写文本。Pen API 把手写文本转换成字符,并把结果显示在输入窗口中。如果需要的话,用户还能再改正文本,并当改正项被识别出来时单击 OK 按钮。Windows 将以一系列 WM_KEYDOWN 和 WM_KEYUP 消息的形式向非笔输入文本编辑器传送这些字符,就好像它们是通过键盘输入的。

Screen Keyboard 支程序将显示一幅典型的键盘的图象,用户可以用笔在该 on-screen keyboard(屏幕键盘)上单击键来进行输入。每按下一个键,相应的按键消息就被发送出去。这种方法不需要进行识别,因为没有包括手写输入。

2.2　笔输入应用程序

Pen API 允许开发者按不同的程度来使用基于笔的计算。对于那些希望通过小量的编程工作就在应用程序中实现许多重要的笔功能的开发者,Pen API 提供了 **DoDefaultPenInput** 函数。**DoDefaultPenInput** 在函数中配备了一个由许多复杂的 API 元素构成的集合。如其名字所隐含的那样,它允许应用程序依赖系统来作出与笔输入有关的决定。在时间和精力允许的情况下,开发者可以逐步地增强基于笔的应用程序的性能。

当接收到光笔设备产生的 WM_LBUTTONDOWN 消息,并进行相应系统调用时,**DoDefaultPenInput** 将逐级发出一系列消息。这些消息反映了识别过程的各个步骤,每一个消息都是对该过程中将发生的下一步骤的通知。应用程序可以在进行每一步骤前预先完成一些动作,也可以忽略消息而由 **DefWindowProc** 函数提供缺省服务。