第七章 延伸性

本章学习目标

在本章中，您将使用在 AutoLISP 及 Microsoft Visual Basic for Applica - tion(VBA) 中新的程序功能。Visual LISP 给予 AutoLISP 程序新的工具，以开发及推出功能更强大的应用程序。对于 ActiveX/VBA 界面的增强包括了您可以载入一个以上的工程(Project)，并增强了访问图形对象的能力、方法及事件(Event)。

- 在（多文件）设计环境（Multiple Document Environment - MDE）中，载入并运行 AutoLISP 程序。
- 使用 Visual LISP 集成开发环境（Integrated Development Envi - ronment - 缩写成 IDE），编辑并格式化(Formatting) AutoLISP 程序。
- 载入多个 VBA 工程(Project)。
- 在 VBA 集成开发环境(IDE) 中编辑 VBA 程序。

Visual LISP 程序

Visual LISP 使您得以更容易、更快速地开发、调试（Debug）及交付 AutoLISP 程序，其新增的功能更增强了 AutoLISP 的能力。

Visual LISP 程序工具

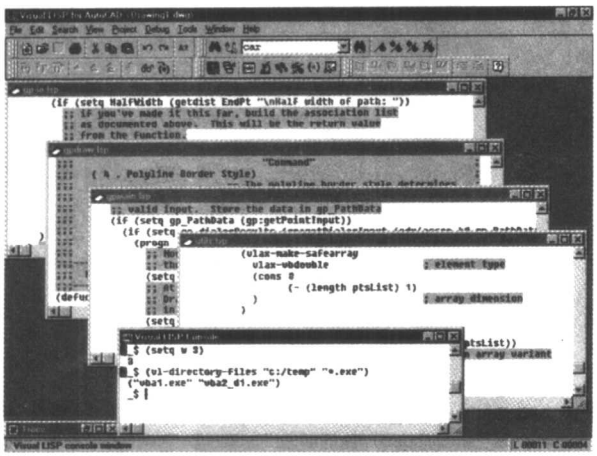
AutoLISP 编辑器（Editor）—协助程序编写者在开发 AutoLISP 程序时，拥有彩色的程序代码、程序代码自动格式化（Formatting）、及智能化侦测内容的辅助说明。

程序调试器—帮助程序编写者寻找并修正程序的错误。

文件编辑—增强程序执行的速度，并且提供安全及有效的传送平台。

灵活的控制台—类似于 AutoCAD 的文字视窗，但是对于 AutoLISP 程序的撰写拥有独特的功能。

工程管理系统—维持多文件的应用程序，其中包括了 AutoLISP、DCL 及 VBA。



Visual LISP 整合开发环境

Visual LISP 对于 AutoLISP 的增强

Visual LISP 对于 AutoLISP 程序语言新增了功能强大的延伸性，而其中的一些功能仅对于 ObjectARX 或 ActiveX/VBA 程序编写者有用。

访问 ActiveX 界面—AutoLISP 程序编写者，现在可以借助对象（Object）技术的力量，使用 ActiveX 界面制作及查询（Query）对象，而产生更快速的应用程序。它也容许 AutoLISP 调用由其他诸如 Visual Basic 等程序语言所撰写的程序。

使用反应器（Reactor）访问应用程序、文件及对象事件（Event）—指定 AutoLISP 程序对于图形事件的反应。每当在图形中的元素(Eliment)被修改、或使用特定的命令时，程序可以自动地执行。

访问系统注册表—程序操控系统的注册表（Registry）项目，如此一来可以更有效地管理使用者环境。

访问文件及文件夹的管理功能—查询、建立与删除文件及文件夹的程序。

函数管理表及字符串的功能—增加了 20 个以上的新函数，其中包括了函数（Routine）的排序（Sorting）及查找。

在多文件设计环境(MDE)中执行 AutoLISP 程序

为了防止在不小心的情况下，在一张图形中执行的 AutoLISP 程序，影响到在另外一张图形中执行的 AutoLISP 程序，因此在每一张图形中拥有各自的内存空间，存放正在执行的 AutoLISP 程序，这样的内存空间称之为 namespaces（名称空间），当您载入一个 AutoLISP 程序时，它将被载入至图形文件的 namespace 中，而其他的图形文件则是无法取用包含在该载入程序中的函数或变量。

如果您仅在一张图形中需要一个 AutoLISP 程序，您可以使用 **APPLOAD** 命令将该程序载入至目前启动的图形中，如果您在所有打开的图形中都需使用该程序，您可以使用下列的方法之一达到此目的。

- 轮流启动每一张图形并使用 **APPLOAD** 命令载入 AutoLISP 程序。虽然这并不是最有效率的方法，但是如果该程序并不经常使用，而且您并不是在每一张图形中都需要执行该程序时，这却是最简单的方法。
- 使用 AutoLISP 的 **VL_LOAD_ALL** 功能，将 AutoLISP 程序载入至打开的图形中。如此将在每一张打开的图形里，载入一个独立分开的 AutoLISP 复制版本，同时也将载入至您目前所启动之 AutoCAD，与陆续所打开的所有图形中，但是该程序并不会载入至您下次重新启动 AutoCAD 时，所打开的图形。
- 使用 ***acaddoc.lsp*** 中的 AutoLISP LOAD 将程序载入。如此一来，将在您每一次建立或打开图形时自动执行，因此 AutoLISP 程序将自动地载入至每一张目前所打开的图形，并且也会载入至您下一次启动 AutoCAD 时所打开的每一张图形中。

注意：*acad.lsp* 仍然存在，它将在您启动 AutoCAD 时执行一次，并不会在您打开新的图形时再执行一次。

- 将程序名称加入至 Startup Suite（启动组）中，在其中所列示的任何文件，都将载入至目前您所打开的所有图形。除此之外，也将载入至未来您启动 AutoCAD 时所打开的任何一张图形中。您可以执行 APPLOAD 命令，使用 Startup Suite（启动组）功能。

不论您使用那一种方法载入 AutoLISP 程序，每一种方法都将在其自己的图形中运作，您可以使用 ***blackboard*** 在程序中共享变量（Variable）值，所谓的 ***blackboard*** 乃是在内存中，程序放置或读取变量的位置。

Visual LISP 的改变

下列为关于 Visual LISP 的改变的一些重点，您可以参考线上辅助说明，其中有更详细的说明及范例。

独立内存空间的应用程序—依据默认值（Default），AutoLISP 程序将载入至图形的内存空间（Namespace），然而您可以将 AutoLISP 编译成一个独立内存空间（Separate-namespace）的应用程序。当您的应用程序是属于独立内存空间应用程序时，它将载入至它自己、独立于图形内存空间的内存空间中，程序中的函数及变数，除非是经由程序出（Export）至图形中，否则将无法在图形中使用。如此可以保护独立内存空间应用程序的符号（Symbol）及函数，避免意外遭到载入至图形中的其他程序的破坏。

虽然独立内存空间应用程序是载入至其自己的内存体中，只有载入该程序的图形可以取用它，如果您在其他的图形中也需使用该应用程序，您必须分别在其他的图形中一一将程序载入。

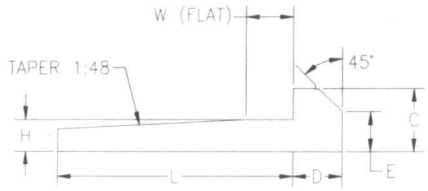
函数及变数的提供—在 Visual LISP 程序库中的所有函数及变数，都将自动地提供给 AutoCAD，因此经编译或未经编译的 AutoLISP 应用程序，都可以获得 Visual LISP 额外功能的好处。由于 Visual LISP 的函数都提供给所有的内存空间，因此您不需在载入包含有 Visual LISP 函数的程序之前，先载入或初始（Initialize）Visual LISP 的程序系统。如果您在程序中使用了 ActiveX 的 Extension 或反应器，您必须分别调用 VL_LOAD_COM 及 VL_LOAD_REACTORS 的方式，初始化这部分的程序库。

当一个程序是载入至图形的内存空间中（此程序不一定要是编译成独立内存空间应用程序的方式），程序中的函数及符号将自动地提供给图形。程序载入之后，其函数及变数在图形中便立即可用了。

程序的编译—当您编译一个 AutoLISP 程序时，您可以将它编译成机器码格式（FAS 文件），或是 Visual LISP 的可执行文件（VLX）。FAS 文件包含了 AutoLISP 程序代码，而 VLX 文件除了包含 AutoLISP 程序代码之外，还包含了诸如 VBA 及 DCL 等的源（Resource）文件。在先前版本的 Visual LISP 中，您也可以将程序编译成 ARX 应用程序，把 Visual LISP 程序程序库包含在一个单一的应用程序中。由于您不需再随着您的应用程序提供程序程序库，因此在新版中您将不再具有这样的选择。

练习 7-1：编辑并执行 AutoLISP 程序

在本练习中，您将打开一个现有的 AutoLISP 程序，然后使用 Visual LISP 集成开发环境（IDE）中的自动格式化及线上说明功能，并且在几个不同的图形中执行程序，它将在您的图形中绘制一把如下图所示的简单钥匙。

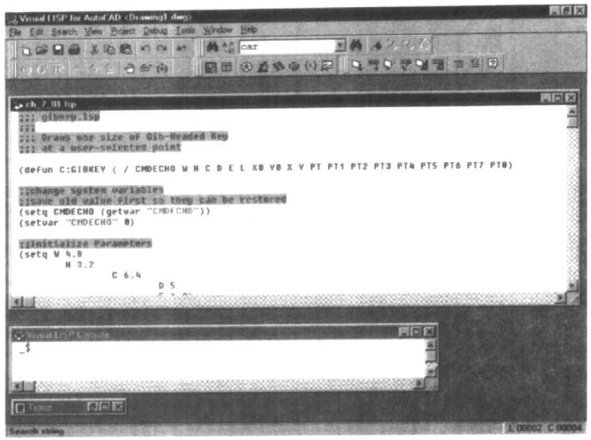


1. 使用默认的图形模板（Template），打开三张新的图形。

2. 在 Tools（工具）菜单中，选取 AutoLISP＞Visual LISP Editor（Visual LISP 编辑器）选项。现在您便处于 Visual LISP 的集成开发环境（IDE）中。

3. 在 Visual LISP IDE 中打开 ***ch_7-01.lsp*** 文件。

   您所打开的程序尚未完成，并且需要重新格式化（Reformat）。您将使用 Visual LISP 的自动格式化程序代码（Code）功能重新格式化，并且在 AutoLISP 程序代码中加入一行，以完成整个程序。



4. 在 Tools 菜单中，选取 Format Code in Editor(设置编辑器代码格式)选项，以重新格式化程序代码。程序代码将根据程序叙述（Expression）的嵌套层次（Nesting Level）自动地缩进（Indent）。程序代码现在变得比较容易解读了。

注意：Visual LISP 的菜单将根据内容而变化，如果您在菜单中未看到您想要选取的项目，请启动包含有程序代码的视窗，然后再次选取 Tools 菜单。

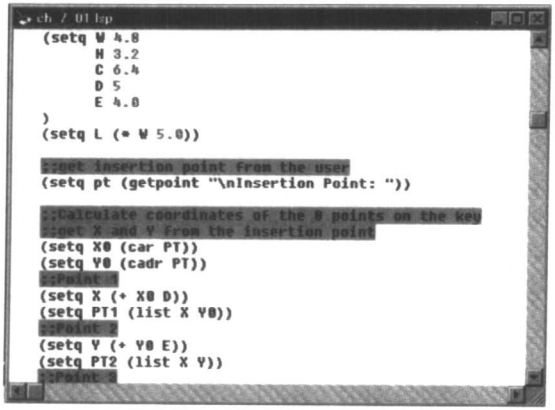
5. 接下来，您将在程序中加入一行程序代码，以便从用户获取插入点（Insertion Point）的资讯。

   - 向下卷动程序代码，直到您看到"get insertion point from the user"这一行注解为止。
   - 在注解的正下方输入下列 AutoLISP 程序代码

   ```
   (setq pt (getpoint "\nInsertion Point: "))
   ```

   当您输入时，编辑器将以不同的颜色自动格式化的变量（User-variables）、内置的函数（Built-in Functions）、字符串（String）及

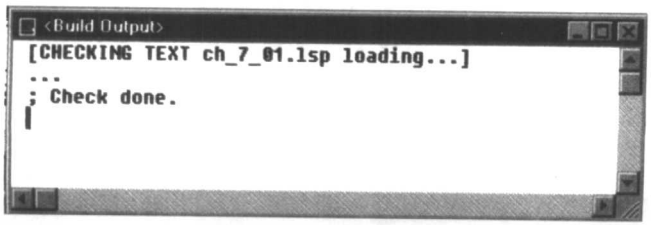
注释（Comment）等。这样的功能可以避免在您编写程序时发生错误，例如：在字串中少了括号，及以内置的变量名称作为使用者变量名称等。



6. 使用 Visual LISP 中自动辨认的帮助（Help）功能，查询 GETPOINT 函数的说明：

   - 亮选（Highlight）***getpoint*** 这个字。
   - 单击 Help（帮助）按键。

7. 在 Tools 菜单中选取 Check Text in Editor（检查编辑器中的文字）选项，以检查程序代码中的句法错误（Syntax Error）。在 Build Output（编译输出）视窗中应出现"Check done"（检查完成）的讯息。如果在其中出现了错误讯息，请检查您所输入的程序代码并进行相同的步骤，再次检查您的程序。



8. 储存您的文件。

9. 在 Tools 菜单中选取 Load Text in Editor(加载编辑器中的文字)，将文件载入至启动的图形中。从 Visual LISP 中将程序载入，与您使用 APPLOAD 命令载入程序是相同的，但是当您在 Visual LISP 的 IDE 中开发并进行程序的测试时，使用前面的方式是比较方便的。

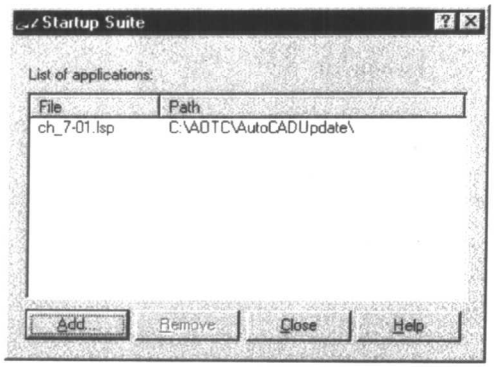
10.  回到 AutoCAD 并在命令行中输入 **gibkey** 命令，以便在启动的图形中执行您刚刚所编译的程序。在您输入了所需要的数据（指定插入点）之后，在您的图形中将出现一把钥匙。

11. 启动另外一张不同的图形并且执行 **gibkey** 命令， 您将看到"Unknown command "GIBKEY". Press F1 for help."（函数未被定义，按 F1 键启动帮助）的信息。这是因为该程序仅载入至启动的图形中，至于其他的图形则无法取用它。

12. 您可以进行下列的步骤，将该程序放置在 Startup Suite（启动组）中

    - 在 AutoCAD 的 Tools（工具）菜单中，选取 Load Application（加载应用程序）选项。
    - 在 Startup Suite（启动组）中，单击 Contents（内容）按键。

    - 单击 Add（添加）按键，选取 ***ch_7-01.lsp*** 文件，然后单击 Add（添加）按键。



- 单击 Close（关闭）按键，回到 Load/Unload Applications（加载/卸载应用程序）对话框。
- 单击 Close（关闭）按键，将 Load/Unload Applications（加载/卸载应用程序）对话框关闭。

一旦 Load/Unload Applications（加载/卸载应用程序）对话框关闭之后，该程序将载入至目前及未来打开的所有图形中。

注意：如果您使用 Visual LISP 新的命令 **VL-LOAD-ALL**，程序仅载入至目前所有打开的图形中，但是并不会载入至您未来再次启动 AutoCAD 时，所打开的图形内。

13. 切换至其他任何已经打开的图形中，然后在命令行中输入 **gibkey** 命令以执行该程序。您应该可以在任何一个图形中执行该命令。

14. 打开一张新的图形，并确认 **gibkey** 命令可以在新打开的图形中执行。

15. 将 AutoCAD 关闭，然后再次启动 AutoCAD 并打开一张新的图形，接着在命令行中执行 **gibkey** 命令。因为透过 Startup Suite（启动组）的功能，系统已经将该 AutoLISP 程序在您启动 AutoCAD 时，自动地载入了。

ActiveX/VBA 程序开发

程序撰写者现在可以使用 ActiveX/VBA 界面，在 Visual Basic 中开发应用程序，这样的界面比起其他的 AutoCAD 程序开发环境，具有更多的好处。

易于使用—程序语言及开发环境易于使用，并且在您安装 AutoCAD 时，便同时安装在您的电脑中。

速度—ActiveX 应用程序比 AutoLISP 或 ADS 应用程序的执行速度更快。

Windows 的协调工作能力—ActiveX 及 VBA 是为了与其他 Windows 应用软件一同使用而设计，它们为跨应用软件之间的沟通提供了绝佳的路径。程序可以与其他诸如 Microsoft Excel、Word 及 Access 等，支持 ActiveX 的应用软件交换数据。

快速的原型开发—在 VBA 中您可以快速地开发程序，即使这些应用程序最终仍将使用其他的程序语言开发，它仍提供了原型（Prototype）应用程序完美的环境。

程序员的基础—ActiveX 及 VBA 技术，为学习 Visual Basic 的程序撰写者，打开了将 AutoCAD 客户化及在 AutoCAD 平台上开发应用程序的大门。

ActiveX/VBA 的新增功能

对于 ActiveX/VBA 界面做了一些改变，使 VBA 程序的功能更强大。经过增强的 VBA 界面可以让您同时载入多个工程（Multiple Project），对于访问 AutoCAD 的对象也做了改进。除此之外，还增加了额外的新对象、方法、性质及事件。

多个模块（Project）—在不需要卸除先前载入模块(Project)的情况下，现在您可以同时载入多个模块，您可以一次选取数个宏（Macro），而不需要为了载入另外一个模块(Project)而先将目前的模块卸除。程序之间可以相互参考（Reference），程序撰写者可以建立共同使用的函数及窗体（Form）的程序库，并且从其他的模块中参考它们。

内置模块—现在，单一的模块可以嵌入至图形中，使 VBA 模块成为图形的一部分。如此可以让您很容易地随着图档传布您的模块，每当您打开图档时，模块将自动地载入，但是您仅能在包含有该模块的图形中执行。

新对象—在 ActiveX 界面中新增了数个新的对象及类（Collection），增强对于图形及应用程序对象的访问。

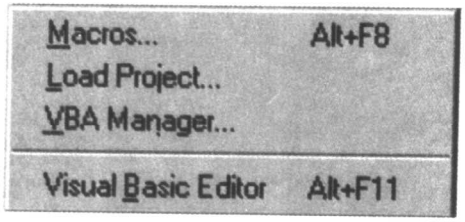
- MenuBar（菜单栏）及 MenuGroups（菜单组）类，增强了对于使用界面的访问。
- 新增的 Visual Basic Extensibility（VBE）对象，及新的 Application 对象方法，允许程序撰写者管理多个工程。
- 新增的 Documents 类中，包含了所打开的各个图形。

新的性质、方法及事件—大部分的对象都有新的性质、方法及事件。

- 大部分的对象都有新的性质、方法及事件。
- 所有的图形对象都有 Modified（修改）事件，它将告知程序该对象是否经过修改。
- 经过增强之后的 Block（图块）对象，现在包含了外部参照（External Reference）的数据。

载入(Load)及执行 VBA 工程(Project)

关于载入及管理 VBA 工程，新增了数个命令及对话框，您可以在 Tools 菜单中选取 Macro（宏）选项，来使用这些新增的命令。



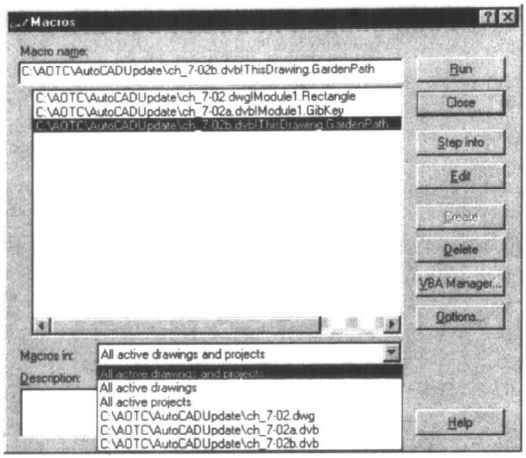
在 Macro 子菜单中包括下列的选项：

- **Macros（宏）**—将显示 Macro 对话框，您可以用它来执行宏。此选项与您在命令行中输入 VBARUN 命令是相同的。
- **Load Project（加载工程）**—将显示 File Open（打开文件）对话框。此选项与您在命令行中输入 **VBALOAD** 命令是相同的。
- **VBA Manager（VBA 管理器）**—将显示 VBA Manager 对话框。此选项与您在命令行中输入 **VBAMAN** 命令是相同的。
- **Visual Basic Editor（Visual Basic 编辑器）**—将显示 Visual Basic 集成开发环境（IDE）。此选项与您在命令行中输入 **VBAIDE** 命令相同。

您可以在命令行中输入 **VBANEW** 命令，建立一个空的 VBA 工程，您也可以透过 VBA Manager 对话框取用此命令。

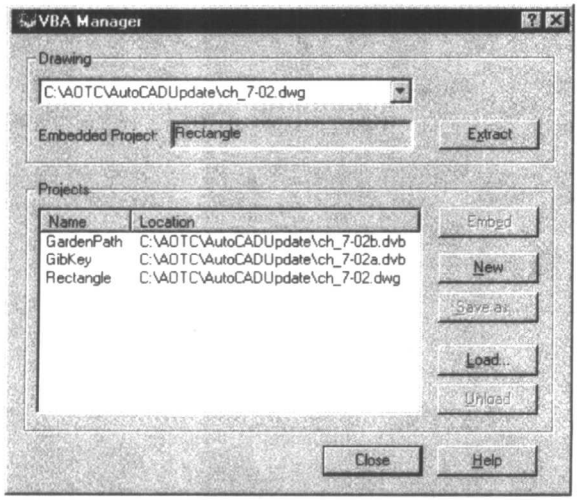
运行 VBA 宏—您可以使用 Macros 对话框，从任何载入的工程中执行宏。依据默认值，包含在工程及打开图形中的所有宏，都将列示在此对话框中。宏的列示内

容，将根据您在 Macros In（宏位置）下拉式列表中，所选取的不同的位置选项而决定。



要执行某一个宏，您可以从列示中选取它，然后单击 Run（运行）按键。

使用 VBA Manager 载入(Load)及嵌入(Embed)工程—您可以使用 VBA Manager 载入及卸载全局（Global）工程、建立新的工程及嵌入（Embed）或提取（Extract）工程。



- 在 Drawing（图形）中列示了您选取之图形中所嵌入(Embed)的工程，您可以单击 Extract（提取）按键，将嵌入至该图形中的工程删除。
- 在 Projects（工程）区域中列示了所有目前载入的 VBA 工程，在 Location（位置）栏位中列示着工程的文件位置，或工程所嵌入之图形的图档位置。

您只要单击位于 VBA Manager（VBA 管理器）对话框中的 Load（加载）按键，便可以将 VBA 工程载入。当您将工程载入之后，该工程将变成全局工程(Clobal Project)，因此您在任何图形中都可以执行。您可以将工程加入至 Load/Unload Applications（加载/卸载应用程序）对话框中的 Startup Suite（启动组）中，以便在您每次启动 AutoCAD 时系统自动地将工程载入。

您可以进行下列的步骤，在 VBA Manager（VBA 管理器）对话框中管理嵌入的工程，或将工程嵌入至图形中。

- 将工程载入。
- 在 VBA Manager 对话框的 Drawing（图形）区域中，选取图形文件。
- 在 Projects（工程）区域中，选取工程名称。
- 单击 Embed 按键。

注意：如果在图形中已经包含有一个嵌入的工程时，Embed 按键将变成无法作用，因此在您嵌入一个新的工程之前，您必须先将现有嵌入的工程提取（Extract）出来。

- 从 VBA Manager 对话框的 Drawing（图形）区域中，单击 Extract（提取）按键。