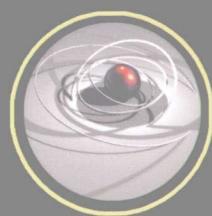


AutoCAD
工程设计书库

AutoCAD 2008

Visual LISP 二次开发入门到精通

周乐来 马婧 编著



 机械工业出版社
CHINA MACHINE PRESS



TP391.72/547

2008

AutoCAD 工程设计书库

AutoCAD 2008 Visual LISP 二次开发入门到精通

周乐来 马婧 编著

图书在版编目(CIP)数据

AutoCAD 2008 Visual LISP 二次开发入门到精通/周乐来, 马婧编著.

—北京: 机械工业出版社, 2008.4

(AutoCAD 工程设计书库)

ISBN 978-7-111-23701-4

I. A… ①周… ②马… ③… ④… ⑤… ⑥… ⑦… ⑧… ⑨… ⑩…

IV. AutoCAD 2008 ② LISP 语言 ③ 二次开发 ④ 计算机辅助设计—应用

中国版本图书馆CIP数据核字(2008)第033152号

机械工业出版社(北京百万庄大街22号 邮政编码100037)

责任编辑: 李 娟

责任印制: 辛 斌

封面设计: 李 娟

北京市中图美编印刷有限公司印刷

2008年2月第1版·第1次印刷

187mm×260mm·1/32印张·434千字

0001—3000册

定价: 39.00元 ISBN 978-7-111-23701-4

定价: 39.00元

机械工业出版社

本书系统地介绍了 Visual LISP 的基础知识和利用 Visual LISP 进行开发的方法与技巧。全书共分为 13 章, 前 4 章介绍 AutoLISP 的语言基础知识, 第 5 章介绍 Visual LISP 的集成开发环境, 第 6 章介绍应用 Visual LISP 调试程序的方法与技巧, 第 7 章介绍如何操作图形数据库对象, 第 8、9 章介绍如何管理、编译应用程序, 第 10 章介绍如何在 Visual LISP 中使用 ActiveX 对象, 第 11 章介绍如何使用反应器, 第 12 章介绍对话框的设计技巧, 最后一章介绍 Visual LISP 程序设计的综合实例。

本书语言简洁, 讲解详略得当, 并附有系统的典型示例。本书实用性强, 便于自学和教学, 是学习 Visual LISP 编程的极佳教程, 是工程设计人员的极佳参考书。

图书在版编目 (CIP) 数据

AutoCAD 2008 Visual LISP 二次开发入门到精通/周乐来, 马婧编著.

—北京: 机械工业出版社, 2008.4

(AutoCAD 工程设计书库)

ISBN 978-7-111-23704-4

I. A… II. ① 周… ② 马… III. ① 计算机辅助设计-应用软件, AutoCAD 2008 ② LISP 表处理语言-程序设计 IV. TP391.72 TP312

中国版本图书馆 CIP 数据核字 (2008) 第 033155 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: 丁 诚 吴鸣飞

责任编辑: 车 忱

责任印制: 李 妍

保定市市中画美凯印刷有限公司印刷

2008 年 5 月第 1 版 · 第 1 次印刷

184mm×260mm·17.75 印张·434 千字

0001—5000 册

标准书号: ISBN 978-7-111-23704-4

定价: 29.00 元

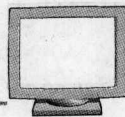
凡购本书, 如有缺页, 倒页, 脱页, 由本社发行部调换

销售服务热线电话: (010) 68326294

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

编辑热线电话: (010) 88379753 88379739

封面无防伪标均为盗版



前 言

AutoCAD 是目前最受欢迎的 CAD 软件之一。该软件自 1982 年问世以来,版本不断升级,功能不断增强,占据着 CAD 领域的主导地位。AutoCAD 之所以在 20 多年的时间里获得如此巨大的发展,除了软件本身具有强大的图形处理功能外,更重要的原因是其开放型结构吸引了众多的 CAD 工作者。AutoCAD 为各行各业的 CAD 软件开发人员提供了功能极其强大的软件平台和丰富的工具,用户可以开发出适合其专业领域的 CAD 应用软件,极大地提高了绘图效率。

AutoCAD 提供的开发工具主要有三种: Visual LISP、VBA (Visual Basic for Applications) 和 Object ARX (AutoCAD Runtime eXtension), 本书主要介绍使用 Visual LISP 进行 CAD 程序设计的基本方法。

本书在详细介绍 Visual LISP 开发工具之前,简要介绍了 LISP 语言和 AutoLISP 语言。LISP (List Processing Language) 是一种计算机的表处理语言,是在人工智能领域广泛应用的一种程序设计语言。AutoLISP 语言是嵌套于 AutoCAD 内部,将 LISP 语言和 AutoCAD 有机结合的产物。使用 AutoLISP 可以直接调用几乎全部 AutoCAD 命令。

Visual LISP 是为加速 AutoLISP 程序开发而设计的软件工具,是一个完整的集成开发环境。Visual LISP 既兼容以前的 AutoLISP 程序,又扩充了许多新的功能,是新一代的 AutoLISP 语言。

在编写过程中,为了使本书更便于理解,作者对全书知识进行了精心的编排和设计。本书实用性较强,从简单到复杂,从基础到应用,层层深入地介绍了 AutoLISP 语言本身的函数结构和语法、Visual LISP 开发工具,让读者在学习本书的过程中能够循序渐进、由浅入深地掌握利用 Visual LISP 进行开发和设计的方法与技巧。

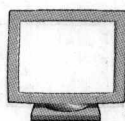
本书由周乐来、马婧编著。

由于作者水平有限,书中如有错误和不当之处,恳请广大读者批评指正。

作者

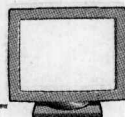
2007.8





目 录

前言	2.4 数据存储结构	18
第 1 章 概述	2.5 数据类型的转换	21
1.1 LISP 简介	2.6 函数	24
1.2 AutoLISP 简介	2.6.1 函数的定义	24
1.2.1 AutoLISP 语言的出现与 发展	2.6.2 函数的调用	25
1.2.2 AutoLISP 语言的特点	2.6.3 函数的递归定义	25
1.2.3 AutoLISP 参考手册	2.6.4 AutoCAD 命令的定义	26
1.2.4 AutoCAD 支持的文件 格式	2.6.5 AutoCAD 命令的调用	26
1.2.5 AutoLISP 程序介绍	2.7 本章小结	26
1.2.6 AutoLISP 程序的内存 分配	第 3 章 AutoLISP 函数	27
1.3 Visual LISP 简介	3.1 基本函数	27
1.4 本章小结	3.1.1 数值函数	27
第 2 章 数据类型、表达式和函数	3.1.2 字符串处理函数	32
2.1 数据类型	3.1.3 表处理函数	34
2.1.1 整型	3.1.4 符号操作函数	37
2.1.2 实型	3.1.5 函数处理函数	40
2.1.3 字符串	3.1.6 错误处理函数	42
2.1.4 表	3.1.7 条件和循环函数	44
2.1.5 文件描述符	3.1.8 应用程序管理函数	46
2.1.6 图元名	3.2 实用工具函数	47
2.1.7 选择集	3.2.1 几何函数	47
2.2 表达式	3.2.2 COMMAND 函数	49
2.2.1 表达式的构成	3.2.3 用户输入函数	51
2.2.2 表达式的求值规则	3.2.4 显示控制函数	56
2.2.3 表达式的求值过程	3.2.5 内存管理函数	58
2.3 变量	3.2.6 文件处理函数	59
2.3.1 变量的命名	3.2.7 设备访问函数	61
2.3.2 变量的数据类型	3.3 本章小结	63
2.3.3 变量的赋值	第 4 章 AutoLISP 程序实例	64
2.3.4 变量的预定义	4.1 修改标注尺寸文本	64
2.3.5 显示变量的值	4.2 绘制红色五角星	65
	4.3 绘制螺旋线	66
	4.4 绘制直齿圆柱齿轮	68
	4.5 中线处画圆	71

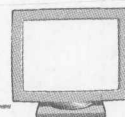


4.6	自动和加粗曲线	72	6.5.3	查看图形数据库的块	115
4.7	本章小结	77	6.5.4	查看选择集	115
第 5 章	Visual LISP 集成开发环境	78	6.5.5	查看外部数据	116
5.1	启动 Visual LISP	78	6.6	本章小结	116
5.2	Visual LISP 环境界面	79	第 7 章	操作图形数据库对象	117
5.2.1	菜单栏	80	7.1	选择集处理函数	117
5.2.2	工具栏	85	7.1.1	选择集构造函数	117
5.2.3	文本编辑器	86	7.1.2	选择集操作函数	122
5.2.4	控制台	88	7.2	实体操作函数	123
5.2.5	跟踪窗口	90	7.2.1	实体名操作函数	123
5.2.6	状态栏	91	7.2.2	实体数据操作函数	126
5.3	加载和运行 AutoLISP 程序	91	7.3	符号表操作函数	132
5.3.1	打开已有的 AutoLISP 程序	91	7.4	扩展数据操作函数	134
5.3.2	进行语法检查	91	7.5	本章小结	135
5.3.3	加载和运行程序	93	第 8 章	应用程序的管理	136
5.4	退出 Visual LISP	93	8.1	管理多个程序文件	136
5.5	本章小结	93	8.2	Visual LISP 工程	137
第 6 章	应用 Visual LISP 调试程序	94	8.2.1	创建一个新的工程	137
6.1	调试功能简介	94	8.2.2	“工程文件”选项卡	138
6.2	程序调试的方法	95	8.2.3	“编译选项”选项卡	141
6.2.1	调试准备	95	8.2.4	工程窗口	142
6.2.2	单步调试	95	8.3	操作工程文件	143
6.2.3	自动执行模式	97	8.3.1	打开一个工程	143
6.3	程序中中断的设置	97	8.3.2	在工程中查找源文件的 内容	144
6.3.1	程序中中断的基本概念	97	8.3.3	在 Visual LISP 应用程序中 包含工程	144
6.3.2	使用断点	98	8.4	本章小结	146
6.3.3	可继续中断循环	100	第 9 章	应用程序的编译	147
6.3.4	不可继续中断循环	101	9.1	程序的编译和链接	147
6.4	数据检验工具	101	9.1.1	Visual LISP 编译器	147
6.4.1	跟踪日志文件	102	9.1.2	加载运行已编译程序	148
6.4.2	监视窗口	103	9.1.3	调用链接函数	149
6.4.3	“跟踪堆栈”窗口	105	9.2	生成应用程序	150
6.4.4	“符号服务”对话框	109	9.2.1	生成新的应用程序	150
6.4.5	“检验”窗口	111	9.2.2	应用程序的更新及修改	151
6.5	查看 AutoCAD 对象	114	9.3	多图档环境下的程序设计	152
6.5.1	查看图形数据库的图元	114	9.3.1	命名空间	152
6.5.2	查看图形数据库的图 形表	115	9.3.2	应用程序的独立命名空间	153

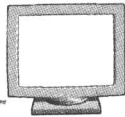




9.3.3	使用图形文档访问函数	153	11.2	创建反应器	185
9.3.4	调用独立变量空间函数	154	11.2.1	创建对象反应器	186
9.3.5	使用图形文档命名变量	155	11.2.2	创建其他反应器	188
9.3.6	在变量空间中共享数据	155	11.2.3	向反应器添加数据	189
9.3.7	自定义错误处理	155	11.3	查询、修改与删除反应器	189
9.4	本章小结	156	11.3.1	检验反应器	189
第 10 章	在 Visual LISP 中使用		11.3.2	使用函数查询反应器	190
ActiveX 对象		157	11.3.3	修改反应器函数	190
10.1	ActiveX 对象简介	157	11.3.4	控制反应器函数	192
10.2	AutoCAD 对象模型	157	11.4	临时反应器和永久反应器	193
10.3	访问 AutoCAD 对象	159	11.4.1	临时和永久反应器的	
10.3.1	查看对象属性	159	转换		193
10.3.2	访问图形对象	161	11.4.2	判断反应器类型	193
10.4	在 Visual LISP 环境中使用		11.4.3	列出当前图形中的永久反	
ActiveX 的方法		163	应器		194
10.4.1	查找所需的函数	163	11.5	反应器的使用规则	194
10.4.2	确定函数的参数	164	11.6	本章小结	194
10.4.3	将数据类型转换为		第 12 章	设计对话框	195
ActiveX 型		165	12.1	对话框简介	195
10.4.4	读取和更新对象属性	169	12.2	对话框控件	196
10.4.5	判断对象是否可以被		12.2.1	定义基本控件	196
访问		171	12.2.2	组合类控件	201
10.4.6	参数带返回值的 ActiveX		12.2.3	其他控件	208
方法		171	12.3	控件的属性	213
10.4.7	集合对象	173	12.3.1	属性数值类型	213
10.4.8	释放对象和空出内存	175	12.3.2	DCL 控件属性	214
10.4.9	转换对象的引用	175	12.4	对话框设计原则与规范	220
10.5	ActiveX 的交互使用	176	12.4.1	通用原则	220
10.5.1	输入类型库	177	12.4.2	通用规范	221
10.5.2	与应用程序建立连接	178	12.4.3	预定义控件的使用规范	222
10.5.3	不引入类型库使用 ActiveX		12.5	对话框控件语言	223
方法		179	12.5.1	对话框文件	223
10.5.4	程序实例	180	12.5.2	DCL 文件结构	224
10.6	本章小结	181	12.5.3	DCL 语法	226
第 11 章	使用反应器	182	12.5.4	对话框出错处理	227
11.1	反应器简介	182	12.5.5	对话框布局	228
11.1.1	反应器类型	182	12.6	对话框驱动程序	232
11.1.2	反应器的回调事件	184	12.6.1	驱动程序调用步骤	232
11.1.3	反应器的回调函数	184	12.6.2	驱动函数	233



12.6.3 对话框的特殊处理	238	线宽	246
12.7 本章小结	241	13.4 参数化绘图实例	248
第 13 章 AutoCAD 中的程序应用	242	13.5 本章小节	260
13.1 设置作图环境	242	附录 AutoCAD 系统变量表	261
13.2 定义字样	244		
13.3 设置图层、颜色、线型和			



第1章 概述

本章内容提要

AutoCAD 是一套向量式计算机辅助制图软件, 它广泛应用于多种领域和行业。用户可以根据自身的专业需求, 对 AutoCAD 进行定制, 使其专业化和用户化。

AutoCAD 提供了程序应用界面, 以 AutoCAD 为基本平台, 让用户或软件开发商利用 Visual LISP 等计算机语言开发特殊的应用程序。

1.1 LISP 简介

LISP (List Processing Language) 语言是人工智能领域中广泛采用的一种程序设计语言, 是一种计算机表处理语言。LISP 语言处理的对象是符号表达式, 因此也被称为符号式语言 (symbolic language)。20 世纪 50 年代末, John McCarthy 发明了 LISP 语言。在几十年的发展过程中, 产生了许多版本的 LISP, 如 MacLISP、InterLISP、ZetaLISP、CommonLISP 等。其中, CommonLISP 是近些年美国的几所大学 (如麻省理工学院、斯坦福大学、卡耐基-梅隆大学等) 和工业界 (如贝尔实验室、HP 公司等) 的研究人员共同研究推出的, 它集成了 MacLISP、InterLISP、ZetaLISP 等版本的优点, 功能较强, 是目前 LISP 语言较完整的版本。

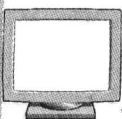
LISP 语言的程序和数据都是以符号表达式的形式来表示的, 因此, 一个 LISP 程序可以把另一个 LISP 程序作为它的数据来处理。LISP 语言语法简单, 数据类型丰富, 编写程序比较方便, 灵活性较高。LISP 语言的内部数据结构非常适合用表来表示, 并且表中还可以嵌套其他的表, 这种构成表的符号表达式放在一对括号中。利用 LISP 语言很容易定义和调用用户编写的函数。这使得 LISP 语言的编程十分灵活。

1.2 AutoLISP 简介

AutoLISP 语言嵌套于 AutoCAD 内部, 它是 LISP 语言和 AutoCAD 有机结合的产物。AutoLISP 是为二次开发 AutoCAD 而专门设计的编程语言。

1.2.1 AutoLISP 语言的出现与发展

早在 20 世纪 80 年代中期, Autodesk 公司的主要创办人 John Walker 就认识到 LISP 语言与 AutoCAD 的协同工作具有潜在可能, 既能够提供非常简单的宏操作, 又能够为类似的操



作提供高级编程语言中广泛的资源。Autodesk 公司从 AutoCAD R2.1 开始引入 AutoLISP 的编程接口。

AutoLISP 采用了和 CommonLISP 最相近的语法和习惯约定,既具有 CommonLISP 的特性,又针对 AutoCAD 增加了许多功能。它可以把 AutoLISP 程序和 AutoCAD 的绘图命令结合起来,使设计和绘图完全融为一体,还可以实现对 AutoCAD 当前数据库的直接访问、修改。AutoLISP 方便了对屏幕图形的实时修改、参数化设计和交互设计,为在绘图领域应用人工智能提供了方便。

AutoLISP 语言最典型的应用之一就是实现参数化绘图的程序设计,包括鼠标拖动程序和尺寸驱动程序等。鼠标驱动程序可以方便用户直接读取 AutoCAD 的输入设备,鼠标拖动应用了 AutoLISP 语言的 GRREAD[<track>]函数,任选项“追踪光标移动”存在且为真时,调整所需的参数值可自动改变屏幕图形的属性。尺寸驱动程序的功能是通过改变实体标注的尺寸值来实现自动修改图形尺寸。迄今为止,大部分参数化程序都是针对二维平面图编制的。实际上,可以利用 AutoLISP 语言实现立体图的参数化绘图。

AutoLISP 语言嵌入 AutoCAD 之后,AutoCAD 就不再只是交互式的图形绘制软件,而成为了真正能够进行计算机辅助设计、绘图的 CAD 软件。

1.2.2 AutoLISP 语言的特点

AutoLISP 具有以下显著特点:

(1) 在普通 LISP 语言的基础上,AutoLISP 语言扩充了许多适用于 CAD 应用的特殊功能,它是一种只能以解释方式运行于 AutoCAD 内部的程序设计语言。

(2) 在 AutoLISP 语言中,许多成分都是以函数的形式给出的,其中没有语句概念或其他语法结构。执行 AutoLISP 程序实质上就是执行一些函数,再调用另一些函数。

(3) AutoLISP 把数据和程序统一表达为表结构,即 S-表达式,就可以把程序当作数据来处理,也可以把数据当作程序来执行。

(4) AutoLISP 语言中的程序运行过程就是对函数求值的过程,是在对函数求值的过程中实现函数的功能。

(5) AutoLISP 语言的功能函数强大,除了具有一般性的功能函数外,还拥有为数不少的控制配合 AutoCAD 的特殊函数。而且 AutoLISP 可直接执行所有 AutoCAD 的指令,并使用所有的 AutoCAD 系统变量。

(6) AutoLISP 语言是一种解释型语言,程序不需要再作编译,“即写即测,即测即用”,可以立即在 AutoCAD 中得到相应的成果。

(7) AutoLISP 语言的主要控制结构采用递归方式。递归方式的使用,使得程序设计简单易懂。

1.2.3 AutoLISP 参考手册

AutoCAD 2008 本身附带 AutoLISP 的说明文件,这些说明文件都是以在线交互的形式提供的。用户在编写 AutoLISP 程序的过程中,可以随时查询在线说明文件,了解函数的用法。

在 AutoCAD 2008 的“帮助”菜单栏选择“帮助”选项,就可以打开“AutoCAD 2008

帮助”窗口，如图 1-1 所示。

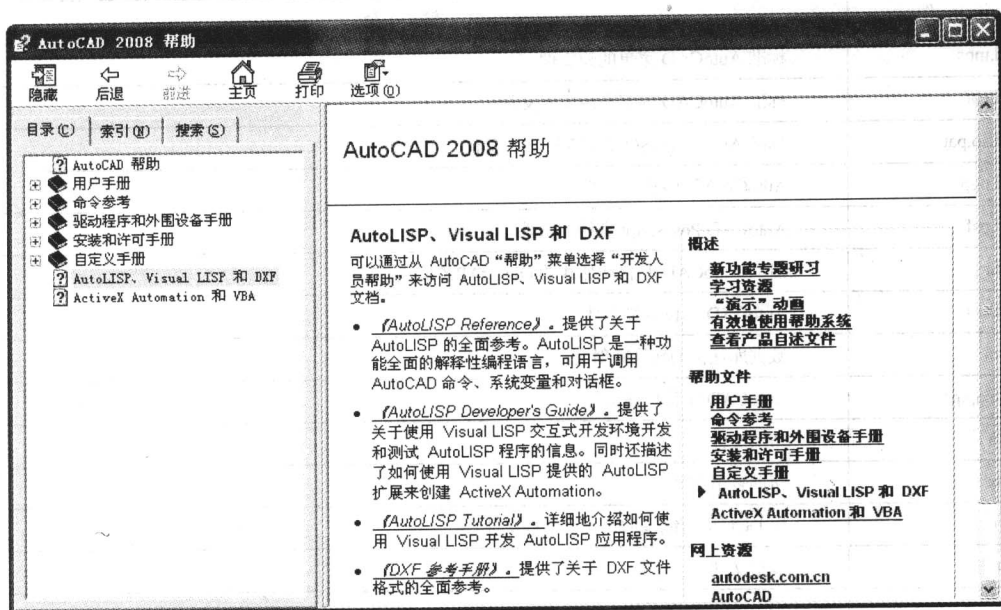


图 1-1 “AutoCAD 2008 帮助”窗口

在“AutoLISP、Visual LISP 和 DXF”栏目中，有四个在线的说明文件，其中：

《AutoLISP Reference》——提供了关于 AutoLISP 的全面参考。此说明文件中，按照字母顺序，说明每个 AutoLISP 函数的用法。

《AutoLISP Developer's Guide》——提供了关于使用 Visual LISP 交互式开发环境开发和测试 AutoLISP 程序的信息。

《AutoLISP Tutorial》——详细介绍了如何使用 Visual LISP 开发 AutoLISP 应用程序。

《DXF 参考手册》——提供了关于 DXF 文件格式的全面参考。

1.2.4 AutoCAD 支持的文件格式

在使用 AutoLISP 进行程序设计时，应该考虑所要使用的文件类型。表 1-1 列出了可编辑的 AutoCAD 支持的文件和文件类型，这些文件都是 ASCII 码的文本文件，用任何文本编辑器都可以创建和扩充这些文件。大多数文件在后续章节会有详细介绍。名字为 acad 或 acadiso 的文件是 AutoCAD 系统定义的文件。在对这些文件进行修改和扩充的时候，必须提前进行备份。

表 1-1 AutoCAD 支持的文件和文件类型

文 件	解 释
acad.lin	标准 AutoCAD 线型库文件
acadiso.lin	标准 AutoCAD ISO 线型库文件
acad.mnl	标准 AutoCAD 菜单调用的 AutoLISP 程序文件
acad.mnu	标准 AutoCAD 菜单的模板源文件



文 件	解 释
acad.mns	标准 AutoCAD 菜单的源文件
acad.pat	标准 AutoCAD 的填充图案库文件
acadiso.pat	标准 AutoCAD ISO 的填充图案库文件
acad.pgp	AutoCAD 程序参数文件
acad.psf	AutoCAD PostScript 支持的文件
acad.rx	启动 AutoCAD 时自动加载的 ObjectARX 应用程序
acad.unt	AutoCAD 单位定义文件
asi.ini	数据库连接的转换映射文件
fontmap.ps	AutoCAD 字体位图文件
*.ahp	帮助文件
*.hdx	帮助索引文件
*.dcl	用 DCL 语言编写的定义对话框文件
*.lin	线型文件
*.lsp	AutoLISP 程序文件
*.mln	多线库文件
*.mnl	同名菜单文件调用的 AutoLISP 程序文件
*.mns	AutoCAD 生成的菜单源文件, 用户可以对它修改或扩充
*.mnu	菜单源文件。包含定义 AutoCAD 菜单的命令字符串和宏语法
*.pat	定义填充图案文件
*.scr	脚本文件
*.shp	定义形/字体的源文件

1.2.5 AutoLISP 程序介绍

本节介绍几个简单的 AutoLISP 程序, 读者可以从中体会 AutoLISP 程序的特性。

例 1-1 AutoLISP 程序入门

```

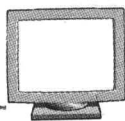
;This program prints a sentence
(defun c:welcome()                ;函数定义
  (prompt "Welcome to the LISP world!") ;显示输出
  (princ);
)

```

本程序的作用是显示输出下面的信息:

Welcome to the LISP world!

其中, defun 是 AutoLISP 系统自带的函数, 加上参数 “c: XXX” 表示向 AutoCAD 系统中增加新的命令 XXX。在 AutoCAD 中加载该 AutoLISP 程序后, 在命令栏里输入 welcome 就可以运行该程序。程序中 prompt 及 princ 均是输出函数, 分别输出字符串和单个



字符。在 AutoLISP 程序中，注释以一个分号“;”开始，并在一行的末尾结束，注释可以出现在程序的任何地方。

例 1-1 的加载步骤如下所示：

1) 新建一个记事本文件，将例 1-1 的程序输入记事本，保存该文件，文件名为 A1-1.lsp（注意扩展名为.lsp）。

2) 在 AutoCAD 2008 中选择菜单栏“工具”→“AutoLISP”→“加载应用程序”菜单命令，弹出“加载/卸载应用程序”对话框，找到步骤 1 创建的 A1-1.lsp，如图 1-2 所示。单击“加载”按钮，此时 AutoCAD 2008 的命令行显示“_appload 已成功加载 A1-1.lsp。”。

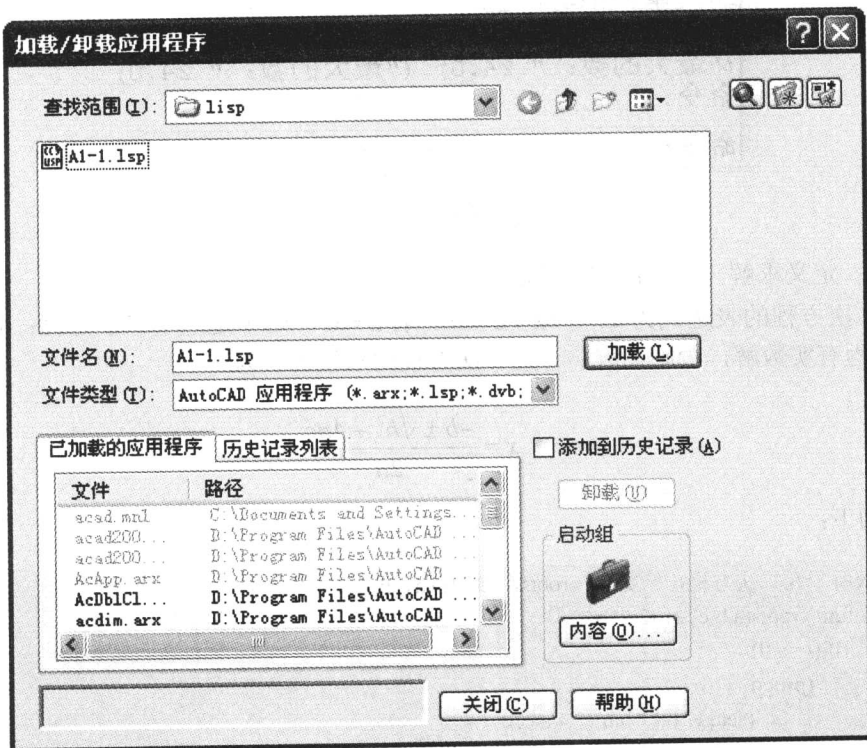


图 1-2 “加载/卸载应用程序”对话框

3) 在 AutoCAD 2008 的命令行输入 welcome，并按回车键，命令行就显示：Welcome to the LISP world!

例 1-2 求最大值程序

```
(defun max(x y) ; 定义函数 max
  (if (> x y) x y) ; 返回 x, y 中的最大值
)

(defun c:work()
  (setq a (getreal "输入第一个数: ")) ; 输入要比较的第一个数
  (setq b (getreal "输入第二个数: ")) ; 输入要比较的第二个数
  (setq c (max a b)) ; 调用函数 max, 将结果返回给变量 c
)
```



(print (list "最大的数: " c))

; 打印 c 的值

)

本程序包含两个函数: work 函数和 max 函数。max 函数的作用是返回 x, y 两数中的最大值, work 函数的作用是获取用户输入的两个数的值, 然后调用 max 函数来求得两数中的最大值, 并显示结果。将该程序加载到 AutoCAD 2008 中后, 执行命令 work, 命令行的显示如图 1-3 所示。

```
命令: work
输入第一个数: 15
输入第二个数: 24
("最大的数: " 24.0) ("最大的数: " 24.0)
命令:
命令:
```

图 1-3 执行例 1-2 的程序

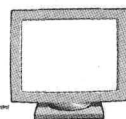
例 1-3 定义求解一元二次方程的函数

一元二次方程的表达式为 $ax^2 + bx + c = 0$ 。若 $a = 0$, 则不是一个二次方程; 若 $b^2 - 4ac \geq 0$, 则方程有实数解, 否则无解。一元二次方程的求根公式为

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

程序如下:

```
;求解一元二次方程, 函数名: roots, 参数: a、b、c。
(defun roots(a b c / t1 t2 x1 x2) ; t1、t2、x1、x2 是局部变量
  (if (/= a 0)
    (progn
      (setq t1 (-(* b b)(* 4 a c)))
      (if(>= t1 0.0)
        (progn
          (setq t2(sqrt t1))
          (setq x1 (/ (- t2 b)(* 2 a)))
          (setq x2 (/ (- 0 b t2)(* 2 a)))
          (print(list x1 x2))
        )
        (print "\n 根是复数.")
      )
    )
    (print "\n 不是一个二次方程.")
  )
  (princ)
)
```



其中, $t1$ 、 $t2$ 、 $x1$ 、 $x2$ 是本程序的局部变量, 它们的定义只在本程序运行期间有效, 若把它们作为全局变量则可能干扰其他程序。`(print(list x1 x2))`若写成`(print x1 x2)`是错误的, 因为 `print` 函数只需一个变元。程序结束前的 `(princ)` 表达式不返回任何值, 其为 `roots` 函数的最后一个表达式, 所以 `roots` 函数将不返回任何值, 这样就不会干扰 `print` 函数的正常打印信息。

假定一元二次方程为 $x^2 - 7x + 12 = 0$, 将该程序加载到 AutoCAD 2008 后, 在命令行输入 `(roots 1 -7 12)` 将返回 `(3.0 4.0)`, 3.0 和 4.0 就是此一元二次方程的两个根。

通过以上几个例子可以看出, AutoLISP 程序具有如下几个特点:

(1) AutoLISP 程序由一个或多个函数组成, 函数又由一个或多个表组成, 甚至函数本身也可以看作一个较复杂的表。因此, 在程序代码中所有的括号必须左右匹配。

(2) 在其他的编程语言中, 有操作符(如+、-、=等运算符号, <、>等关系符号)与函数(`scanf`、`printf`等系统函数及用户自定义函数)的区别, 但在 AutoLISP 语言中, 这两者是同等对待的, 都可以视为函数。

(3) 调用函数时, 必须将函数名放在表中第一个元素的位置, 其参数为随后依次排列的其他元素。表中的函数与参数、参数与参数之间至少要用一个空格来分开。

(4) AutoLISP 同 C 语言一样, 其调用的函数也可以分为内部函数和外部函数, 其中, 内部函数是 AutoCAD 系统自带的, 外部函数是用户自行定义的, 但在调用时没有区别。AutoLISP 语言的本身就是对一个个 AutoLISP 函数的调用, 函数是 AutoLISP 语言处理数据的工具, 学习 AutoLISP 语言, 核心就是要掌握 AutoLISP 函数。

1.2.6 AutoLISP 程序的内存分配

每个程序在执行的时候都要占用一定的系统资源, 而其中最重要的就是内存。AutoLISP 是一种解释性语言, 其程序代码本身及程序中定义的变量、函数在程序中都要占用一定的存储空间。在运行 AutoLISP 程序的时候, 用到了下面两个内存区域。

- 堆 (Heap): 存储所有的函数和变量。程序使用的函数和变量越多, 变量名越长, 函数越复杂, 该区域所占的空间就越大。
- 栈 (Stack): 存储函数的变量和局部结果。标准表中嵌套的表越多, 函数执行的递归次数越多, 该区域所占的空间就越大。

在 AutoCAD 中运行 AutoLISP 的时候, 用户不能扩展系统的堆或栈空间。一般情况下, 这些内存区域能够运行绝大多数程序; 但是如果用户在使用函数递归调用时, 调用自身的次数太多, 以致用完了所有的栈空间, AutoLISP 就会显示如下信息: `Insufficient node space` (没有足够的节点空间), 并终止当前函数的运行。如果在特殊情况下, 执行 AutoCAD 时没有足够的内存来提供上述空间, 则 AutoCAD 不能载入 AutoLISP, 并且显示下列信息: `Insufficient memory-AutoLISP disabled` (没有足够的内存空间, AutoLISP 不可用)。

1.3 Visual LISP 简介

Visual LISP 是 Autodesk 公司在 1997 年的 AutoCAD 14 版本中推出的。它是为加速 AutoLISP 程序开发而设计的软件开发工具, 是一个完整的集成开发环境。前期版本的



第 1 章

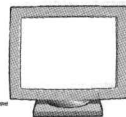
AutoLISP 一般没有提供任何编辑器或编译器，用户大多选择一种自己熟悉的文本编辑器；而 Visual LISP 提供的集成开发环境可以自行编辑、编译、修改及调试 AutoLISP 源程序。Visual LISP 包括文本编辑器、格式编排器、语法检查器、源代码调试器、检验和监视工具、文件编译器、工程管理系统、上下文相关帮助与自动匹配功能和智能化控制台等。Visual LISP 的用户界面良好，用户在短时间内就能够掌握。

Visual LISP 不仅继承了 AutoLISP 程序设计的优点，还允许用 AutoLISP 程序维护 AutoCAD 的资源。在 Visual LISP 集成环境下开发 AutoLISP 程序，不再像以前那样编辑程序时，要用其他系统的文本编辑程序编写程序代码；调试程序时需要用户自己决定在程序的什么位置插入打印语句，以便查看变量的内容；在程序运行正常后还必须将插入的调试代码删除。在 Visual LISP 集成环境下可以便捷、高效地开发 AutoLISP 程序，可以经过编译得到运行效率高、代码紧凑、源代码受到保护的应用程序。

Visual LISP 对 AutoLISP 语言的功能进行了扩展，可以通过 Microsoft ActiveX Automation 接口与 AutoCAD 对象进行交互，可以通过反应器函数扩展 AutoLISP 响应事件的能力。后续章节会详细介绍 Visual LISP 的相关应用。

1.4 本章小结

本节首先向读者简单介绍了 LISP 语言的基本概念、历史和发展趋势；之后详细介绍了 AutoLISP 语言的基本概念，介绍了 AutoLISP 的出现、发展和特点，并且介绍了几个简单的 AutoLISP 程序，帮助读者理解 AutoLISP 的特点；最后简单介绍了 Visual LISP 的基本概念，在随后的章节中，将详细介绍 Visual LISP 语言。



第2章 数据类型、表达式和函数

本章内容提要

本章进一步介绍 AutoLISP 语言的基本要素——数据类型、符号表达式、变量、数据存储结构、数据类型的转换以及函数。其中也会介绍一些程序代码，读者可以通过程序实例对 AutoLISP 有一个整体的理解。

2.1 数据类型

AutoLISP 数据类型丰富，包括：整型（INT）、实型（REAL）、字符串（STR）、表（LIST）、文件描述符（FILE）、AutoCAD 的图元名（ENAME）、AutoCAD 的选择集（PICKSET）等。

2.1.1 整型

整型就是整数。整数由 0、1、2...9、+、- 等字符组成，正号“+”可以省略。在目前通常使用的 32 位机上，整数用 32 位表示（二进制），其范围从 -2147483648 ~ +2147483647。整数的大小与所使用的计算机位数有关。若用户输入的数据超过上述范围，AutoLISP 会自动将整数转换成实数。然而，对两个有效整数进行算术运算时，若结果超出上述范围，那么所得结果将是无效的。

2.1.2 实型

实型数据是带小数点的数，又称为浮点数。实型数用双精度表示，并且至少有 14 位的精度。在 -1 ~ 1 之间的小数，小数点之前的 0 不能省略，否则计算机会误认为点对（见 2.4 节）而出错。例如，-.12 或 .12 是错误的，应该写成 -0.12 或 0.12。实型数的范围比整型数大得多，不容易超界，故用户可以尽量采用实型数。

实型数也可以采用科学计数法表示，即数字后可有一个 e 或 E，其后跟数的指数。例如， 0.123×10^{11} 表示为 0.123e11 或者 0.123E11。但是应注意 e 或 E 之前必须有数字，且指数必须为整数，如 e9、0.25E1.5 均不是合法的指数形式。

2.1.3 字符串

字符串又称为字符常数，是由包括在一对双引号中的一组字符组成的。双引号为字符串的界定符。字符串中的大小写字母和空格符都是有意义的，字符串还可以包含 ASCII 表中的任何字符。

字符串中字符的个数（不包括双引号）称为字符串的长度。字符串可以是空的，即空串