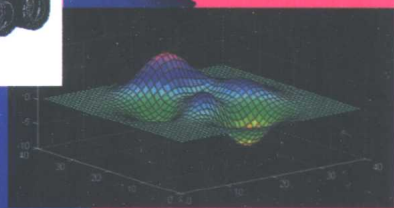
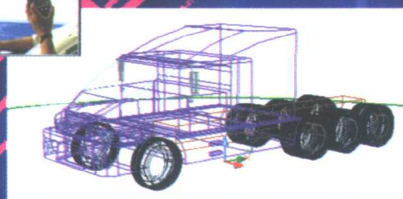
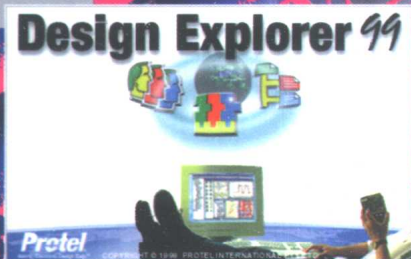
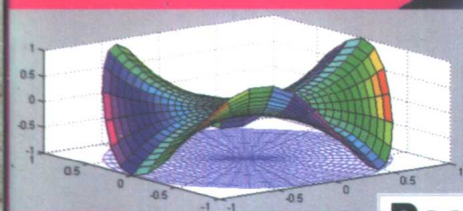
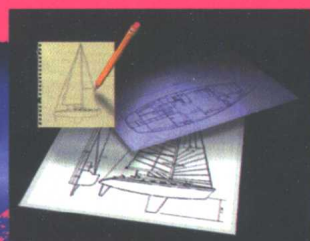


计算机工程设计与应用开发丛书

AutoCAD 2000 开发工具应用详解

清源计算机工作室 编著



机械工业出版社
China Machine Press

计算机工程设计与应用开发丛书

AutoCAD 2000 开发工具应用详解

清源计算机工作室 编著



机械工业出版社

本书主要讲解如何使用 AutoLISP 和 Visual LISP 工具进行应用程序开发, 全书共分 15 章。第 1 章至第 6 章主要讲述基本的 AutoLISP 开发知识和对话框设计与管理知识; 第 7 章主要讲述了 AutoLISP 函数, 包括基本的 AutoLISP 函数和扩展的 AutoLISP 函数; 从第 8 章至第 15 章, 主要讲述如何使用 Visual LISP 集成开发环境进行 LISP 应用程序开发, 其中包括 Visual LISP 开发界面的使用、Visual LISP 编写代码的使用, 以及应用程序的调试, 另外还详细讲述了 ActiveX 的使用方法以及反应器特征。本书内容主要以结合实例的方式, 由浅入深, 系统而全面地讲述了使用 AutoLISP 和 Visual LISP 进行二次开发的基本知识, 用户可以从中学到许多 AutoCAD 2000 为二次开发用户在 Visual LISP 中增加的新的、强大的功能。

本书适用于使用 AutoCAD 进行产品设计, 并使用 AutoLISP 和 Visual LISP 进行应用程序二次开发的工程技术人员和软件开发人员。本书还可供高等院校和培训学校相应专业的师生参考使用。

机械工业出版社 (北京市百万庄大街 22 号 邮政编码 100037)

责任编辑: 边 萌 封面设计: 姚 毅

责任印制: 何全群

三河市宏达印刷厂印刷·新华书店北京发行所发行

2000 年 4 月第 1 版第 1 次印刷

787mm×1092mm 1/16·22.25 印张·532 千字

0 001—5000 册

定价: 40.00 元 (ICD, 含配套书)

ISBN 7-900043-07-1/TP·07

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

本社购书热线电话 (010) 68993821、68326677-2527

前 言

随着计算机技术的发展, 计算机软件在工程设计领域的应用越来越广。在机械、电子、建筑等行业, 应用计算机软件进行产品设计的 CAD 软件也非常丰富, 使产品设计人员能够高效率地进行各自领域的产品分析、设计等工作。这些应用于工程设计领域的 CAD 软件有 AutoCAD、Protel、MATLAB 等。这些软件极大地提高了机械、电子等行业的产品设计质量与效率, 是目前 CAD 领域应用最为广泛的软件, 是工程设计领域中最有用的辅助设计软件。为了帮助工程设计人员学习这些软件, 快速掌握这些软件的使用与开发技术, 我们特编写一套“计算机工程设计与应用开发丛书”。AutoCAD 2000 主要应用于机械产品设计和开发, 以及 AutoCAD 2000 的二次开发; Protel 99 主要应用于电子原理图的设计、电路板的设计和绘制, 以及电子逻辑分析和仿真等; MATLAB 主要应用于工程方面的数学计算、自动控制系统的分析, 以及图形与图像处理等。

本套书主要面向工程设计人员, 涉及的知识不但包括软件应用知识, 还包括专业基础知识, 以及软件在相关领域的二次开发技术知识, 是机械、电子领域技术人员的最佳参考书。本套书在介绍软件的使用过程中, 结合丰富的实例进行讲解, 使读者能快速掌握相关的知识。本套书不但讲述软件的基础应用知识, 还讲述了软件的中、高级应用知识, 是一套面向中、高级读者的全面而系统的参考书。本套书包括《AutoCAD 2000 命令与实例详解》和《AutoCAD 2000 开发工具应用详解》; 《Protel 99 原理图和 PCB 设计》和《Protel 99 仿真与 PLD 设计》; 《MATLAB 基础及其应用》和《MATLAB 高级应用——图形及影像处理》, 基本覆盖了三种软件在机械领域和电子领域的应用。全套书均附有光盘, 以实例为主, 将软件的实际应用很生动地展现在读者面前。

本套书由清华大学和科学院从事该领域工作多年的博士生和硕士生进行编写, 具有贴近读者的特点, 而且书中列举了大量的实例, 是使用这三种软件的工程设计人员不可多得的参考书。

清源计算机工作室

2000 年 3 月

编者的话

AutoCAD 2000 是 Autodesk 于 1999 年推出的最新版本软件, AutoCAD 软件集图形处理、产品设计、图形数据管理以及网络技术于一体, 成了连接世界的产品设计平台, 在 CAD 发展领域具有深远的意义。AutoCAD 2000 不但可以用来进行产品设计绘图, 而且还提供了二次开发工具, 可实现产品参数化设计等二次开发, 并可以针对用户的需求实现产品的二次软件开发, 大大提高了产品设计的效率。AutoCAD 2000 提供了多种开发工具, 如 AutoLISP 及 Visual LISP、ObjectARX 2000 和 VBA 等, 这些都是基于 AutoCAD 2000 平台的二次开发软件。

本书主要讲述如何使用 AutoCAD 2000 优秀的二次开发工具 AutoLISP 及 Visual LISP 进行程序设计, 以及开发基于 AutoCAD 平台的 CAD 软件。AutoLISP 作为一种 LISP 编程语言, 是 AutoCAD 2000 内嵌的编程工具, 是 AutoCAD 2000 整体的一部分。AutoLISP 易于使用并很灵活。使用 AutoLISP, 你可以编写适合于图形处理的很有效的宏程序及函数。另外, AutoCAD 2000 为使用 AutoLISP 语言提供了一个非常有用的可视化开发工具—Visual LISP 2000。使用该开发工具可以更高效地开发 LISP 应用程序。Visual LISP 应用现代化的 LISP 引擎以支持多文档、对象和事件, 并且提供强大的集成开发环境 (IDE)。利用 Visual LISP 支持 ActiveX 的优点, 可以开发运行效率更高的应用程序, 而且可以生成更加安全的可执行程序 (VLX 和 FAS)。

本书围绕如何使用 AutoLISP 和 Visual LISP 工具开发应用程序进行讲解。全书共分 15 章, 第 1 章至第 6 章主要讲述基本的 AutoLISP 开发知识和对话框设计与管理知识; 第 7 章主要讲述 AutoLISP 函数, 包括基本的 AutoLISP 函数和扩展的 AutoLISP 函数; 从第 8 章至第 15 章主要讲述如何使用 Visual LISP 集成开发环境进行 LISP 应用程序开发, 其中包括 Visual LISP 开发界面的讲解, Visual LISP 代码的编写, 以及应用程序的调试, 另外还详细讲述了 ActiveX 的使用方法以及反应器特征。附录中列出了 AutoLISP 错误代码, 以供参考。本书系统而全面地讲述了使用 AutoLISP 和 Visual LISP 进行二次开发的基本知识, 用户可以从中学学习到许多 AutoCAD 2000 为二次开发用户在 Visual LISP 中增加的新的、强大的功能。

在阅读本书时, 读者可以先学习前面的 AutoLISP 基础和对话框设计部分, 也可以先学习第 8 章至第 12 章的 Visual LISP 集成开发环境的基本部分, 然后再从前到后顺序学习, 对于某些 Visual LISP 调试和窗口选项的设置, 可以直接学习第 15 章。

本书适用于使用 AutoCAD 进行产品设计, 并使用 AutoLISP 和 Visual LISP 进行应用程序二次开发的工程技术人员和软件开发人员。本书还可供高等院校和培训学校的相关专业师生参考使用。

本书由江思敏主编, 郑巍、阴向阳等同志参与编写了部分章节。由于水平有限, 时间仓促, 书中缺点和不足在所难免, 敬请广大读者批评指正。

编者

2000 年 3 月

目 录

前言

编者的话

第 1 章 AutoLISP 基础	1
1.1 概述	1
1.2 AutoLISP 表达式	1
1.3 AutoLISP 数据类型	2
1.3.1 整数	3
1.3.2 实数	3
1.3.3 字符串	3
1.3.4 表	4
1.3.5 选择集	4
1.3.6 实体名	4
1.3.7 VLA 对象	4
1.3.8 文件描述符	5
1.3.9 符号和变量	5
1.4 AutoLISP 程序文件	7
1.4.1 格式化 AutoLISP 代码	7
1.4.2 注释	8
1.4.3 颜色译码	8
1.5 AutoLISP 变量	9
1.5.1 显示一个变量的值	9
1.5.2 预定义的变量	10
1.5.3 Nil 变量	10
1.6 数据处理	10
1.7 字符串处理	12
1.8 基本输出函数	14
1.8.1 显示信息	14
1.8.2 字符串控制字符的使用	15
1.8.3 统配符匹配	16
1.9 关系和条件处理	17
1.10 表处理	20
1.10.1 点表	25
1.10.2 点对	26
1.11 符号和函数处理	27
1.11.1 符号处理	27

1.11.2	函数处理	29
1.11.3	使用 defun 定义一个函数	31
1.11.4	向 AutoCAD 增加命令的 C:XXX 函数定义	31
1.11.5	函数的局部变量定义	33
1.11.6	函数变元	34
1.12	错误处理	35
1.12.1	使用 *error* 函数	37
1.12.2	获取错误并继续执行程序	38
1.13	应用程序处理	39
1.13.1	应用程序处理函数	39
1.13.2	装载 AutoLISP 应用程序	42
1.13.3	装载 ARX 应用程序	43
第2章	AutoLISP 与 AutoCAD 通信	45
2.1	查询和命令函数	45
2.1.1	执行 AutoCAD 命令	45
2.1.2	系统及环境变量	48
2.1.3	配置控制	48
2.2	显示控制	48
2.2.1	命令行交互输出	49
2.2.2	控制菜单	49
2.2.3	控制图形和文本窗口	51
2.2.4	控制低级图形	51
2.3	获取用户输入	52
2.3.1	用户输入函数 getxxx	52
2.3.2	用户输入函数条件的控制	54
2.4	几何实用函数	56
2.4.1	目标捕捉 (Object Snap)	56
2.4.2	文本区域 (Text Extents)	57
2.5	转换	60
2.5.1	字符串转换	60
2.5.2	角度转换	62
2.5.3	ASCII 码转换	63
2.5.4	单位转换	64
2.5.5	坐标系转换	66
2.6	文件处理	68
2.6.1	写字符到屏幕或文件中	68
2.6.2	写字符串到屏幕或文件中	69
2.6.3	读一个字符	69

2.6.4	读一个字符串	70
2.6.5	打开与关闭文件	70
2.6.6	文件的查找	70
2.6.7	搜索选择指定扩展名的文件	71
2.6.8	帮助文件处理	72
2.7	设备访问与控制	74
2.7.1	存取用户输入	74
2.7.2	数字化仪的校准	74
第3章	操作 AutoCAD 对象	77
3.1	选择集处理	77
3.1.1	选择集过滤表	78
3.1.2	在 AutoLISP 和 ObjectARX 之间传递选择集	82
3.2	对象处理	83
3.2.1	实体名函数	83
3.2.2	实体数据函数	88
3.2.3	实体数据函数与图形屏幕	93
3.2.4	旧格式多义线与 Lw 轻多义线对象	94
3.2.5	非图形对象的处理	95
3.3	扩展数据 (XDaTa)	96
3.3.1	扩展数据的组织结构	97
3.3.2	注册应用程序	98
3.3.3	扩展数据检索	99
3.3.4	附加扩展数据于实体	100
3.3.5	使用扩展数据内存管理	101
3.3.6	扩展数据的句柄	101
3.4	扩展记录 XRecord 对象	101
3.5	访问符号表与字典	102
3.5.1	符号表	102
3.5.2	字典条目	104
第4章	设计对话框	105
4.1	对话框部件	105
4.2	使用 DCL 定义对话框	107
4.2.1	base.dcl 和 acad.dcl 文件	107
4.2.2	引用 DCL 文件	107
4.2.3	DCL 语法	108
4.3	使用 Visual LISP 显示对话框	110
4.3.1	预览错误处理	111
4.3.2	DCL 文件语义核查	112

4.4	调整对话框的布置	113
4.4.1	对话框实例	113
4.4.2	建立控件组	114
4.4.3	控件间的间距	115
4.4.4	右端和底端的空间	116
4.4.5	加框行和列周围的空间	116
4.4.6	自定义退出按钮文本	117
4.5	设计指导	118
4.5.1	美观和功效性	118
4.5.2	前后一致的设计和清晰明了的语言	118
4.5.3	用户控制	119
4.5.4	容错处理	120
4.5.5	提供帮助功能	120
4.5.6	对残疾用户的考虑	120
4.5.7	单词字母的大写	121
4.5.8	缩略语的使用	121
4.5.9	布局	121
4.5.10	控件的大小和位置	121
4.5.11	将控件置为无效	122
4.5.12	对话框的嵌套使用	122
4.5.13	隐藏对话框	122
4.5.14	提供默认数值	122
4.5.15	键盘输入	123
4.5.16	预定义控件和控件组的指导	123
4.5.17	错误处理	126
第5章	管理对话框	127
5.1	使用 AutoLISP 程序控制对话框	127
5.1.1	快速入门	127
5.1.2	一个对话框打开时的函数使用限制	129
5.2	动作表达式和回调	129
5.2.1	动作程序与回调函数	129
5.2.2	动作表达式	130
5.2.3	回调原因	131
5.2.4	默认值和 DCL 动作	132
5.3	处理控件	132
5.3.1	初始模式与值	132
5.3.2	在回调时改变模式和值	132
5.3.3	处理互锁组	133

5.3.4	处理滑动条	134
5.3.5	处理编辑框	135
5.4	嵌套对话框	135
5.5	隐藏对话框	136
5.5.1	隐藏对话框	136
5.5.2	要求口令	137
5.6	列表框和下拉列表	138
5.6.1	列表操作	139
5.6.2	处理列表部件	140
5.7	图像控件和按钮	141
5.7.1	生成图像	141
5.7.2	处理图像按钮	143
5.8	特定应用数据	143
5.9	DCL 错误处理	144
5.10	对话框函数概要	145
5.10.1	函数调用顺序	145
5.10.2	对话框的简单块定义	146
第 6 章	可编程对话框参考	147
6.1	控件属性	147
6.1.1	属性类型	147
6.1.2	受限制的属性	148
6.1.3	用户自定义属性	148
6.2	预定义属性概要	148
6.3	DCL 属性分类	150
6.4	对话框控件函数概要	157
6.4.1	预定义活动控件	157
6.4.2	控件组	157
6.4.3	用于修饰和说明的控件	157
6.4.4	文本组	158
6.4.5	对话框的退出按钮和错误控件	158
6.4.6	限制使用的控件	158
6.5	DCL 控件分类	158
6.6	可编程对话框函数概要	170
6.6.1	对话框的打开和关闭函数	170
6.6.2	控件和属性处理函数	170
6.6.3	列表框和下拉列表处理函数	170
6.6.4	图像控件处理函数	170
6.6.5	应用程序特定数据处理函数	171

第7章 AutoLISP 函数概述	172
7.1 分类概述	172
7.2 基本函数	172
7.2.1 应用程序处理函数	173
7.2.2 数学函数	173
7.2.3 字符串处理函数	174
7.2.4 等量和条件函数	175
7.2.5 表操作函数	175
7.2.6 符号处理函数	176
7.2.7 函数处理函数	177
7.2.8 错误处理函数	177
7.3 实用工具函数	178
7.3.1 转换函数	178
7.3.2 设备访问函数	178
7.3.3 显示控制函数	178
7.3.4 文件处理函数	179
7.3.5 几何实用函数	180
7.3.6 查询和命令函数	180
7.3.7 用户输入函数	180
7.4 选择集、对象和符号表函数	181
7.4.1 扩展数据处理函数	181
7.4.2 对象处理函数	181
7.4.3 选择集处理函数	182
7.4.4 符号表和词典处理函数	183
7.5 内存管理函数	183
7.6 Visual LISP 提供的 AutoLISP 扩展函数	183
7.6.1 ActiveX 集成操作函数	184
7.6.2 ActiveX 数据转换函数	184
7.6.3 ActiveX 方法激活函数	185
7.6.4 ActiveX 对象处理函数	185
7.6.5 ActiveX 特性处理函数	185
7.6.6 曲线测量函数	185
7.6.7 字典函数	186
7.6.8 对象处理函数 (Visual LISP 向 AutoLISP 扩展)	186
7.7 反应器函数	187
7.8 VLX 名称空间函数	189
7.9 名称空间通信函数	189
7.10 Windows 注册函数	190

第 8 章 熟悉 Visual LISP 环境	191
8.1 启动 Visual LISP	191
8.2 熟悉 Visual LISP 用户界面	192
8.2.1 Visual LISP 文本编辑器	193
8.2.2 其他 Visual LISP 窗口	193
8.3 了解 Visual LISP 菜单	193
8.3.1 可变的菜单内容	194
8.3.2 Visual LISP 菜单概述	194
8.4 掌握控制台窗口	195
8.5 使用 Visual LISP 文本编辑器	196
8.6 装载和运行 AutoLISP 程序	196
8.6.1 运行选择代码行	198
8.6.2 使用扩展的 AutoLISP 函数	198
8.7 退出 Visual LISP 环境	198
第 9 章 使用 Visual LISP 开发程序	200
9.1 开始组织	200
9.2 使用控制台窗口	200
9.2.1 控制台特性简述	201
9.2.2 使用具有多个图形的控制台窗口	203
9.2.3 使用控制台快捷菜单	203
9.2.4 记录控制台窗口的动作	204
9.3 使用文本编辑器	205
9.3.1 编辑一个文件	206
9.3.2 使用文本编辑器快捷菜单	207
9.3.3 使用文本编辑器快捷键	208
9.3.4 移动和拷贝文本	210
9.3.5 文本搜索	211
9.3.6 给文本做书签标记	213
9.4 使用控制台和编辑器代码帮助	214
9.4.1 Visual LISP 颜色译码	215
9.4.2 使用 Apropos 特征	216
9.4.3 让 Visual LISP 帮助用户完成词输入	219
9.4.4 为 AutoLISP 函数获取帮助	221
9.5 使用 Visual LISP 将源代码格式化	221
9.5.1 Visual LISP 格式化样式	222
9.5.2 应用格式化选项	224
9.5.3 应用 Visual LISP 注释样式	227
9.5.4 保存和恢复格式化选项	228

9.5.5	格式化器的约束	228
9.5.6	格式化快捷键	229
9.6	检查语法错误	229
9.6.1	检查括号的平衡	230
9.6.2	使用颜色译码检查语法错误	231
9.6.3	使用 Check 命令查找语法错误	232
9.6.4	查找程序中语法错误位置	232
第 10 章	调试程序	234
10.1	Visual LISP 调试特征	234
10.2	调试实例	234
10.2.1	设置中断程序运行的断点	235
10.2.2	单步执行程序	236
10.2.3	监视一个表达式的计算结果	237
10.2.4	继续程序的执行	238
10.2.5	以 Animate (活动) 模式运行程序	238
10.3	使用 Visual LISP 调试特征	238
10.3.1	启动一个调试过程	239
10.3.2	中断环	239
10.3.3	使用断点	241
10.4	使用 Visual LISP 数据检查工具	243
10.4.1	使用监视窗口	244
10.4.2	跟踪堆栈窗口	244
10.4.3	使用符号服务对话框	249
10.4.4	使用 “Inspect (检查)” 窗口	251
10.4.5	查看 AutoCAD 图形实体	254
第 11 章	建立应用程序	258
11.1	编译和链接程序	258
11.1.1	使用编译器	258
11.1.2	从一个文件编译一个程序	258
11.1.3	浏览一个编译实例	260
11.1.4	装载并运行编译程序	261
11.1.5	链接函数调用	262
11.2	创建应用程序模板	262
11.2.1	创建一个新应用程序	262
11.2.2	装载和运行 Visual LISP 应用程序	267
11.2.3	改变应用程序选项	267
11.2.4	重新创建一个应用程序	268
11.2.5	更新一个应用程序	268

11.3 为多文档环境设计	268
11.3.1 名称空间	268
11.3.2 在自己的名称空间中运行一个应用程序	270
11.3.3 共享名称空间之间的数据	273
11.3.4 处理多文档环境中的错误	274
11.3.5 在一个多文档环境下使用 AutoLISP 的限制	275
第 12 章 维护 Visual LISP 应用程序	276
12.1 管理多个 LISP 文件	276
12.2 定义一个工程	277
12.2.1 设置工程文件属性	278
12.2.2 使用工程窗口操作工程包含文件	280
12.3 使用已定义的工程	282
12.3.1 打开一个工程	282
12.3.2 在工程源文件中查找字符串	282
12.3.3 在 Visual LISP 应用程序中包括一个工程	283
12.4 优化应用程序代码	284
12.4.1 定义创建选项	284
12.4.2 选择一个编译模式	286
12.4.3 选择一个链接模式	286
12.4.4 安全优化	287
第 13 章 使用 ActiveX	289
13.1 在 AutoLISP 中使用 ActiveX 对象	289
13.2 掌握 AutoCAD 对象模型	289
13.2.1 对象特性	290
13.2.2 对象方法	291
13.2.3 对象集合	291
13.3 访问 AutoCAD 对象	291
13.3.1 使用检查工具查看对象属性	292
13.3.2 应用程序对象分支	293
13.3.3 过程概述	293
13.3.4 性能考虑	294
13.4 以 ActiveX 方法使用 Visual LISP 函数	295
13.4.1 确定用户需要的 Visual LISP 函数	295
13.4.2 确定如何调用一个函数	296
13.4.3 转换 AutoLISP 数据类型为 ActiveX 数据类型	296
13.4.4 查看和更新对象属性	300
13.4.5 使用返回变元的值的 ActiveX 方法	303
13.4.6 列出一个对象的属性和方法	303

13.4.7	使用集合对象	305
13.4.8	释放对象和内存	308
13.4.9	转换对象引用	308
13.4.10	处理由 ActiveX 方法返回的错误	309
13.5	使用 ActiveX 与其他应用程序相互作用	311
13.5.1	输入一个类型库	311
13.5.2	建立与一个应用程序之间的联系	313
13.5.3	编写一个实例应用程序	313
13.5.4	没有输入一个类型库使用 ActiveX	316
第 14 章	附加反应器到 AutoCAD 图形	318
14.1	反应器概述	318
14.2	了解反应器类型和事件	318
14.3	定义回调函数	320
14.4	创建反应器	322
14.4.1	使用对象反应器	323
14.4.2	附加数据到反应器对象	324
14.5	在多个名称空间使用反应器	324
14.6	查询、修改和移去反应器	325
14.6.1	检查反应器	326
14.6.2	使用函数调用查询反应器	326
14.6.3	修改反应器	326
14.6.4	移去反应器	327
14.7	暂时的和持久的反应器	328
14.8	反应器用户指导	329
第 15 章	Visual LISP 环境和格式设置	330
15.1	窗口属性设置	330
15.1.1	语法颜色选项	330
15.1.2	当前配置	331
15.1.3	设置当前窗口为原型	332
15.1.4	设置所有窗口为原型	332
15.1.5	字体	332
15.2	环境选项设置	332
15.2.1	一般选项	332
15.2.2	Visual LISP 格式化选项	335
15.2.3	页面格式选项	337
15.3	保存设置	338
附录	AutoLISP 错误代码	339

第 1 章 AutoLISP 基础

AutoLISP 作为一种 LISP 编程语言, 是 AutoCAD 内嵌的编程工具, 是 AutoCAD 整体的一部分。它易于使用并很灵活, 使用 AutoLISP 可以编写适合于图形处理的很有效的宏程序及函数。本章主要介绍 AutoLISP 编程语言的基本概念。因为 AutoLISP 代码无需编译, 所以用户可以直接在命令行中输入代码并立即看到结果。通过练习本章的样例, 用户可以快速学习并掌握 AutoLISP 的基本知识。

1.1 概述

AutoLISP 是基于 Common LISP 发展而来的, 它不但具有与 Common LISP 相一致的句法和语法规则, 而且具有许多适合于 AutoCAD 的专用函数, 可以实现 AutoLISP 程序与 AutoCAD 绘图命令的结合, 使设计与绘图成为一体; 还可以通过 AutoLISP 程序实现对 AutoCAD 图形数据的直接访问、修改等处理, 以便对图形进行绘制、编辑等实时处理, 实现图形的交互设计绘制。使用 AutoLISP 进行 AutoCAD 二次开发, 重要的是先掌握 AutoLISP 的语法规则及如何使用 AutoLISP 函数进行程序设计, 然后再结合 AutoCAD 的绘图命令就可以实现图形的参数化绘制等开发目的。

AutoLISP 是一种人工智能语言, 它是一种基于符号处理的、面向对象的语言, 而不是过程性的语言, 在处理语句的过程中是通过对对象进行求值实现的。在 AutoCAD 的开发工具中, AutoLISP 是唯一的一种非编译的、解释性的语言, 所以在运行 AutoLISP 应用程序时, 只需直接调用其编写的函数, 就可以实现目标操作, 不需要一个专门的编译工具。如果再结合 AutoCAD 的绘图命令及相关的内部函数, 不仅可以用来开发设计、计算、绘图、DCL 对话框等应用程序, 而且能生成 Windows 类对话框。

利用 AutoLISP 进行 AutoCAD 的二次开发, 可以帮助用户充分利用 AutoCAD, 从而大大节省绘图时间并提高工作效率。AutoLISP 的一个典型应用是实现图形参数化设计。在工程设计中, 图形的绘制常存在大量的重复劳动, 在设计进程中, 还存在着大量的修改设计, 而通过实现参数化设计, 则可以使工程设计人员减少不必要的重复劳动, 提高其设计效率。

本章重点介绍 AutoLISP 函数的使用和 AutoLISP 应用程序的编写, 为学习后面的程序设计开发打好基础。

1.2 AutoLISP 表达式

在命令行中输入文本时, AutoCAD 将该文本与内部的可用命令名列表作比较, 以解释该文本。如果输入的文本与列表中的某项相匹配, AutoCAD 执行该命令。当 AutoCAD 接收到 AutoLISP 代码时, 它将该代码传递给 AutoLISP 解释器。

AutoLISP 解释器的核心是计算器。该计算器先读取一行代码, 对它求值, 然后返回一个结果。该代码必须符合 AutoLISP 表达式的格式要求, 它可以从文件中读取, 也可

以由用户从 AutoCAD 命令行中输入。

所有 AutoLISP 表达式的格式都如下所示：

(function arguments)

每个表达式都以一个左括号开始，由一个函数名和一个该函数的参数组成，并且每个参数都可以是一个表达式。表达式以右括号结束。每个表达式都返回一个可由外层表达式使用的值。如果没有外层表达式，则 AutoLISP 将该值返回给 AutoCAD 命令行。例如，如下代码调用了三个函数。

(fun1 (fun2 arguments)(fun3 arguments))

第一个函数 fun1 有两个参数，另两个函数 fun2 和 fun3 各有一个参数。函数 fun2 和 fun3 被函数 fun1 所包含，因此它们的返回值作为参数传递给 fun1。函数 fun1 由这两个参数计算函数值，并将该值返回给命令行。

如果在 AutoCAD 命令提示中输入 AutoLISP 表达式，AutoLISP 将计算该表达式并显示结果，然后重新显示命令提示。下例展示了 *（乘）函数的用法，该函数接受一个或多个实数作参数。

Command: (* 2 27)

54

因为此函数没有外层表达式，所以它将结果返回给命令行。嵌套在其他表达式中的表达式将它们的结果返回给外层表达式。下例用 +（加）函数的结果作 *（乘）函数的一个参数。

Command: (* 2 (+ 5 10))

30

如果输入的闭（右）括号数量不对，AutoLISP 将显示如下提示：

n>

其中，n 是一个整数，表明还有几层左括号尚未匹配。如果出现此提示，用户必须输入 n 个右括号后才能对表达式求值。如

Command: (* 2 (+ 5 10

2>))

30

常见错误是忽略了文本字符串中的双引号 (")。在这种情况下，右括号被解释为字符串的一部分，而不会对 n 产生影响。要改正此错误，可先按 ESC 键取消该函数，然后重新输入正确的表达式。

1.3 AutoLISP 数据类型

AutoLISP 计算器按括号中代码的次序和数据类型来处理表达式。在用户充分利用 AutoLISP 之前，必须了解数据类型之间的区别以及如何使用它们。下面详细讲述 AutoLISP 的数据类型。