第零章　专业工具盘及环境配置

写《AutoCAD 专业开发工具》一书的目的旨在使它便于使用。为此，尽量提供了支撑文件、工具及文档，并使它们易于存取和使用。因此，在使用本书前无需对系统进行过多繁锁配置。但在通读本书前，先阅读本章，会使以后的工作更顺利。

0.1　专业工具盘(PT-DISK)

书中所有练习使用的源程序、实用工具及图形，均收录在随书发行的 PT 盘中。这些文件被压缩成 4 个可执行的自解文件。为完成本书的所有练习，必须安装 PT-DISK 文件及 LISPSQL 文件。

0.1.1　安装

按下列步骤安装 PT 工具盘中的练习文件：

安装 PT 练习文件

将 PT 工具盘插入驱动器 A。若不使用 A 驱，则替换成当前盘符。

C:\>**MD\PT** ↵	创建 PT 子目录
C:\>**CD\PT** ↵	进入 PT 目录
C:\PT>**A:\PT** ↵	将 PT 盘中的文件解压缩至 C:\PT 目录

PT 盘中包括一个文本编辑器 WCEDIT、对话框创建器 PROTOBOX，以及由 AutoLISP 支持的 ASI 命令文件 LISPSQL，它们均保存在 PT 盘中自己的目录下。这些文件的细节在本章后面介绍。

0.1.2　许可权

在 PT 工具盘中的文件 LICENSE.TXT 提供了使用 PT 工具盘中资料的许可权。这个许可权并不包括 PT 盘中的 WCEDIT、PROTOBOX 以及 LISPSQL。以上程序有自己特定的许可权限。LICENSE.TXT 如下所示：

> The AutoCAD Professional's API Toolkit——PT-DISK
> 1993 New Riders Publishirg
>
> Version 12.00 for Release 12 PC-DOS/MS-DOS
> Developed by Kurt Hampe and Jim Boyce.
>
> Permission to use, copy, modify, and distribute this software for any purpose is hereby granted, provided that no fees are collected directly or indirectly and that the above copyright notice and this permission notice appear in all

copies and in all supporting documentation.

(对本软件的任何目的的使用、拷贝、修改、颁发均是允许的，所有这些均不收费，并允许在所有的拷贝及支撑文档中复制以上许可信息)。

0.1.3 声明

如下的声明包含在PT盘的文件DISCLAIM.TXT中。此外，WCEDIT、PROTOBOX以及LISPSQL在它们的文档中可能有更进一步的阐述及声明。

This software is provided AS IS without warranty of any kind, either express or implied, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. Neither the publisher nor its dealers or distributors assume any liability for any alleged or actual damages arising from the use of this program. (Some states do not allow the exclusion of implied warranties, so the exclusion may not apply to you.)

0.2 WCEDIT

WCEDIT是专为AutoLISP编程设计的文本编辑器。做为一个编程环境，WCEDIT提供许多其它通用编辑器所不具备的特点。应按照下面的步骤安装WCEDIT(有关WCEDIT更进一步的信息，请查阅安装后的有关文档)。

安装 WCEDIT

将PT盘插入驱动器A。若使用其它驱动器，则替换成相应的盘符。

C:\>**MD\WCEDIT** ↵	创建子目录WCEDIT
C:\>**CD\WCEDIT** ↵	进入WCEDIT子目录
C:\>WCEDIT>**A:WCEDIT\WCEDIT** ↵	将WCEDIT的文件由A盘解压缩至C:\WCEDIT目录

安装完WCEDIT后，按以下步骤进行：

1. 编辑AutoCAD的启动批处理文件，把子目录C:\WCEDIT加入ACAD环境变量所指定的目录中。例如：

```
SET ACAD=C:ACAD;\C:\ACAD\SUPPORT;\C:\ACAD\ADS;
C:\ACAD\ASE;C:\ACAD\FONTS;C:\WCEDIT.
```

2. 在"Command:"提示下执行批处理文件进入AutoCAD图形编辑器：

```
Command:(load"WC") φ
```

WCEDIT显示一段信息(包括机器码)并提示输入密码。

3. 与1-800-272-ELSA联系或将机器码送与ELSA，注册WCEDIT并获取密码。在解锁WCEDIT时输入密码，以后就不用再次输入了，除非配置文件损坏或系统状态改变。

安装完WCEDIT并在Command下键入(load "WC")后，可在"Command:"提示下键

入"wc"使用 WCEDIT。

0.3　PROTOBOX

PROTOBOX 是一个用以产生对话框 DCL 文件的实用程序。按以下步骤安装：

安装 PROTOBOX

将 PT 盘插入 A 驱。若不使用 A 驱，则替换成相应的盘符。

C:\>**MD\PROTOBOX** ↵	创建子目录 PROTOBOX
C:\>**CD\PROTOBOX** ↵	进入子目录 PROTOBOX
C:\PROTOBOX>**A:\PROTOBOX\PROTOB** ↵	将 PROTOBOX 文件由 A 盘解压缩至 C:\PROTOBOX 目录

安装完 PROTOBOX 后，按以下步骤操作：

1. 编辑 AutoCAD 的启动批处理文件，在环境变量 ACAD 所指向的目录中加上子目录 C:\PROTOBOWX。例如：

```
SET ACAD=C:\ACAD;C:\ACAD\SUPPORT;C:\ACAD\ACIS;
C:\ACAD\ASE;C:\ACAD\FONTS;C:\PROTOBOX
```

2. 查阅随 PROTOBOX 所带的文档寻找有关注册 PROTOBOX 的信息。
3. 查阅文档寻找 PROTOBOX 的使用指南。

0.4　LISPSQL

LISPSQL 是一个 ADS 应用程序，它提供 AutoLISP ASI 功能。PT 盘中装的是 AutoCAD 12 c2 版的 LISPSQL。

注释： 为完成第三部分 AutoLISP ASI 编程的练习，必须安装完整的 LISPSQL。

按下列步骤安装 LISPSQL：

安装 LISPSQL

将 PT 盘插入 A 驱动器。若不使用 A 驱，则替换成相应的盘符。

C:\>**CD\PT**	进入 PT 子目录
C:\PT>**A:\LISPSQL\LISPSQL** ↵	将 LISPSQL 文件由 A 盘解压缩至 C:\PT

安装完毕后，阅读文档 LISPSQL.DOC。第三部分将介绍如何使用 LISPSQL.EXP。

0.5 AUTOEXEC.BAT 的组织

使用本书不必对 AUTOEXEC.BAT 进行修改，安装完 AuotCAD 后，AUTOEXEC.BAT 已进行了必要的修改。

Kurt Hampe 所使用的 AUTOEXEC.BAT 如下所示：

```
@ECHO OFF
PROMPT $P$G
PATH C:\;C:\DOS;C:\PHAR386\BIN;C:\WATCOM;C:\ACAD;
C:\WINDOWS;C:\MACH32;C:\MOUSE;C:\WORD
SET TEMP=C:\WINDOWS\TEMP
LH /L:2,31816 C:\MOUSE\MOUSE
LH /L:2,13981 C:\DOS\SHARE.EXE
```

0.6 CONFIG.SYS 的组织

使用本书无须对 CONFIG.SYS 进行修改，安装完 AutoCAD 12.0 以后，系统已做了必要的修改。

Kwrt Hampe 所使用的 CONFIG.SYS 如下示：

```
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE NOEMS HIGHSCAN
BUFFERS=30,0
FILES=99
DOS=UMB
LASTDRIVE=H
FCBS=16,0
DEVICEHIGH /L:2,12048=C:\DOS\SETVER.EXE
DOS=HIGH
STACKS=9,256
SHELL=C:\DOS\COMMAND.COM C:\DOS\ /e:1024 /p
DEVICEHIGH /L:1,44496 =C:\DOS\DBLSPACE.SYS /MOVE
```

提示： 注意，在 CONFIG.SYS 中 SHELL 命令行中有/E:1024 的选项。这一选项为环境变量分配定量内存。在 AutoCAD12、Windows、编译器、连接器同时工作时大约需 10～15 个环境变量。当加载环境变量(特别是通过批处理文件)时，若 DOS 提示"out of environment space"，则应在 SHELL 后加/E 选项。若已加/E 选项，则增大/E 选项的数值，以便为环境变量分配更多的内存(以字节为单位)。

0.7　AutoCAD 启动批处理文件的组织

使用本书需对 AutoCAD 的启动批处理文件进行必要的修改。为使用第三部分的源程序、练习及样例，需设置环境变量 EXCHANGE，并指向 PT 子目录。此外，为使用第十四章的程序、练习及样例，需设置环境变量 AVECFG、RDPADI、RHPADI 以及 AVERDFILE。

Kurt Hampe 所使用的 AutoCAD 启动批处理文件如下：

```
SET ACAD=C:\ACAD;C:\ACAD\SUPPORT;C:\ACAD\FONTS;
C:\ACAD\ADS;C:\ACAD\SAMPLE;C:\ACAD\PT
SET ACADCFG=C:\ACAD
SET ACADDRV=C:\ACAD\DRV
SET EXCHANGE=C:\PT
SET AVECFG=C:ACAD
SET RDPADI=C:\ACAD\DRV\RCPSVADI.EXP
SET RHPADI=C:\ACAD\DRV\RHEXPORT.EXP
SET AVERDFILE=640,111,1,1,256,15,16
C:\ACAD\ACAD=1=2
SET ACAD
SET ACADCFG
SET ACADDRV
SET EXCHANGE
SET AVECFG
SET RDPADI
SET RHPADI
SET AVERDFILE=
```

0.8　编译器和连接器

本书所使用的 ADS 应用程序均已经过编译和连接，可执行文件与源程序代码一起由 PT 工具盘提供。如果用户欲编译并连接自己的应用程序，则必须配备自己的编译器及连接器（见第一章）。

使用编译器及连接器需设置一系列环境变量，并对 AUTOEXEC.BAT 及 CONFIG.SYS 进行修改。AutoCAD 提供与 ADS 库文件及头文件一起进行编译和连接的批处理文件（见第一章）。

PT 工具盘中用于编译和连接 ADS 应用程序的编译器/连接器是 WATCOM C/386 9.0。WATCOM C/386 是当前唯一的保护模式编译器，它不需使用 Phar-Lap 连接器就可编译和连接 ADS 应用程序。

Kurt Hampe 使用的 WATCOM C/386 编译器/连接器批处理文件在下面列出，它来源于 AutoCAD 的 WATCOM C/386 9.0。

```
@ECHO OFF
@ECHO MSDOS BATCH FILE TO BUILD WATCOM C/386 ADS SAMPLE PROGRAMS
SET SAVWAT=%WATCOM%
SET SAVPTH=%PATH%
SET SAVINC=%INCLUDE%
```

```
SET WATCOM=C:\WATCOM
SET PATH=%WATCOM%\BIN;%WATCOM%\BINB;%PATH%;
SET INCLUDE=%WATCOM%\H;..

IF EXIST %WATCOM%\LIB386\DOS\CLIB3S.LIB GOTO L1
ECHO YOU MUST EDIT W90SAMP.BAT AND CHANGE THE VARIABLE WATCOM
GOTO DONE
:L1

IF .%1==. GOTO L2
@ECHO COMPILING %1.C
  wcc386p %1 -fpi287 -3s -s -oailt -zq
  wlinkp system ads file %1 library wcads90,
  c:\acad\api\wcapi90.lib, c:\acad\ase\asipw.lib option quiet
GOTO DONE

:DONE

SET INCLUDE=%SAVINC%
SET PATH=%SAVPTH%
SET WATCOM=%SAVWAT%
SET SAVINC=
SET SAVPTH=
SET SAVWAT=

@ECHO
@ECHO W90SAMP COMPLETED
@ECHO ON
```

0.9 硬件配置

本书对计算机硬件无特殊要求，只要能使用 AutoCAD 即可。然而，用户的应用程序可能需要更多的系统资源。因此，每部分的最前面一章均对所需的系统硬件支持进行讨论。

本书中涉及到的应用程序均在多种计算机系统测试过。下面是 Kurt Hampe 所使用的计算机。

- 486DX2-50 带 256K 缓存及 32 位局部总线结构，32 位的局部总线 SVGA 图形卡，250M 硬盘及 32 位局部总线硬盘卡，8M RAM 及 15 英寸平面直角 SVGA 显示器。
- 386DX-25 带硬件缓存，16 位 SVGA 图形卡，80M 硬盘及 16 位硬盘卡，4M RAM，14 英寸 SVGA 显示器及一个调制解调器。

Kurt Hampe 使用的是下列打印机：

- 一台带 2.5M 内存的 HP LaserJet IIP Plus。
- 一台 9 针点阵式打印机。

第一部分　AutoCAD开发系统

第一章　AutoCAD开发系统概述

本章对AutoCAD开发系统(ADS)及开发ADS应用程序的基本概念、方法及需求进行介绍。本章所介绍的每一个条目的细节及语法格式，将在第二章和第三章中论述。

第二章"ADS编程"对ADS应用程序的结构进行了剖析，并介绍如何编写完整的AutoCAD ADS应用程序。第三章"ADS技术"将通过实例说明如何用ADS来完成预期的目的。

1.1　AutoCAD开发系统(ADS)

AutoCAD开发系统(ADS)是一个开发AutoCAD外部功能的C语言编程环境，它包括一组C函数库及头文件。一个ADS应用程序是一系列经过编译和连接的C函数，这些函数作为AutoCAD的外部函数由AutoLISP控制。ADS应用程序不能作为独立的程序使用，它只能作为一组被AutoLISP调用的外部函数。因此，AutoCAD执行ADS应用程序和执行AutoLISP应用程序一样。

有时ADS被看成一种只面向高水平的开发者的开发工具。但这种看法是不正确的，AutoCAD为用户处理了绝大部分ADS的潜在复杂性。ADS并不难使用，如果掌握了C语言，就可以使用ADS，而且使用ADS带来的便利远远超过学习和使用它所带来的障碍。

为了清晰和连贯起见，本书的第二、三、四部分均在介绍AutoLISP的基础上对ADS进行讨论。

注释： 一般来说ADS是一个C语言开发环境，但Windows允许使用Visual Basic作为对C的补充。本书的非Windows部分以C语言为讨论焦点。第五部分有对Windows特性详尽的介绍。

1.1.1　应用程序接口(API)

应用程序接口(API)是一系列AutoCAD引入的规则和标准，它允许函数对AutoCAD的核心命令和函数发出请求。AutoCAD的附加功能，例如扩展实体造型(AME)，为ADS提供了C语言函数库(第十三章有关于AME API的介绍)。由于这些函数没有置于AutoCAD内也没有放在标准的ADS函数库中，故它们必须通过API与AutoCAD共同工作。

AVE Render及其函数内置于AutoCAD，这意味着它不需要外部函数库，实际上，AutoCAD中的渲染函数指的是渲染命令。第十四章有使用渲染函数的详尽介绍。

无论函数是内置于AutoCAD中还是由函数库提供，均允许编写应用程序与AutoCAD直接连接。这些函数均是API的一部分。

1.1.2 应用程序的 SQL 接口(ASI)

应用程序的 SQL 接口(ASI)是 AutoCAD 扩展 SQL 应用程序接口(AutoCAD SQL Extensio's API)的简称。ASI 用于区分 AutoCAD 扩展 SQL 的函数和由其他函数库提供的函数。对编程者而言,ASI 只是指代某些特定的 API 函数。

1.2 ADS 和 AutoLISP 的比较

在学习 ADS 之前,应该对 ADS 和 AutoLISP 之间的区别以及各自的利弊有一定的了解。

1.2.1 AutoLISP

AutoLISP 是一种用于 AutoCAD 环境的解释语言。在介绍 OS/2 上的 AutoCAD10 及 DOS 上的 AutoCAD11 中的 ADS 以前,所有的 AutoCAD 应用程序均用 AuotLISP 编写。AutoLISP 有以下的优势及弊端:

优势

- 解释语言。代码不必经过编译即可执行。代码的每一行均由解释程序先翻译成机器码再执行。因为程序即使有错误也可运行到错误行,故用户可以方便地跟踪以寻找代码中的错误。
- 与 AutoCAD 通信简便。AutoLISP 解释器是 AutoCAD 运行环境的一个组成部分,应用程序无须控制特定的函数与 AutoCAD 通信。这样既简化了通信,同时也减少了代码长度。
- 内置于 AutoCAD。运行 AutoLISP 不必购买额外的资源,如编译器和连接器。
- 允许出错。AutoLISP 对代码中的错误较为宽容。AutoLISP 应用程序出错后并不影响 AutoCAD 的运行,很少像 ADS 应用程序那样导致系统崩溃。
- 高级语言。AutoLISP 是一种高级语言。绝大多数的数据类型及结构的细节已由解释器替用户处理了,在应用程序中用户可省去它们的定义。
- 可移植性。AutoLISP 代码可移植到任何 AutoCAD 平台上。

弊端

- 速度。由于 AutoLISP 是一种解释型语言,故用 AutoLISP 编写的应用程序比编译语言编写的应用程序慢。
- 内存用量大。由于是一种高级语言,AutoLISP 为用户处理了绝大部分的数据类型及结构的细节。因此 AutoLISP 对内存需要量大。在应用程序中,由于加大内存需求强度,势必造成运行速度慢处理能力降低的后果。
- 使用系统资源有局限。AuotLISP 不允许应用程序对操作系统及硬件直接进行操作,不能充分利用系统资源。
- 安全性差。解释的语言代码可用文本编辑器观看,因此源代码得不到保护。不过 AutoCAD 提供了对 AutoLISP 代码加密的实用程序。

- 返回值。AutoLISP 函数只有一个返回值。应用程序经常用大量的全局变量以从函数取回“返回值”，这样会降低源程序的易读性及增加应用程序的复杂性。

1.2.2 ADS

在许多方面，使用 ADS 编程和使用 AutoLISP 一样，应对它们有足够的了解。但是，具体到语法、函数、变量定义及函数的参数和结果缓冲器，ADS 和 AutoLISP 有许多不同。由于 ADS 使用 C 语言，它继承了 C 语言的特点，其优势和弊端列表如下：

优势

- 编译语言。ADS 应用程序的执行代码是经过编译连接产生的机器码，执行速度较快。
- 灵活性大。C 语言对系统资源的分配控制能力很强。另外，C 语言允许定义灵活的数据类型来满足需要。
- 直接对操作系统及硬件进行操作。由于使用 C 语言，ADS 允许应用程序对操作系统和硬件做低层次的直接操作。
- 安全性。经编译产生的代码在文本模式下无法阅读。这些代码可由反汇编及跟踪技术破译，但能做到这一点，并能读懂结果的人必定可以自己从头编写这个应用程序。
- 库函数。C 语言允许用户建立自己的函数库，以便于以后的应用程序直接调用
- 结构。C 语言是一种结构化语言，允许用户编写支持不同算法的模块。
- 中级语言。C 是一种编译型中级语言，它比 AutoLISP 节省内存。
- 返回值。C 允许对返回值进行更好的控制，因为函数可返回指针以指向内存变量。
- AutoCAD 开发工具。Autodesk 公司提供的 AutoCAD 开发工具包含完整的样板应用程序、实用程序及附加文档资料。这些是使用 ADS 编程的充分理由。如果决定用 ADS 来开发用户的应用程序，则应购买 AutoCAD 开发工具。

弊端

- 背景知识。在使用 ADS 将应用程序集成于 AutoCAD 之前，必须对 C 语言有一定的了解。
- 昂贵。由于代码必须经过编译，故在测试函数前要么使用实模式 C 编译器及其内置连接器，要么使用保护模式的 C 编译器及 Phar-Lap 连接器。要想充分使用系统资源，必须使用保护模式的 C 编译器。并非所有的 C 编译器和连接器与 AutoCAD 兼容，因此必须选择能与用户的 AutoCAD 平台共同工作的 C 编译器及连接器来开发用户自己的应用程序。
- 可移植性。如果只使用符合 ANSI 标准的 C，则 ADS 代码在任何 AutoCAD 平台上能保证源程序级的兼容。这些源码必须在新平台上重新进行编译和连接，而且新的平台必须提供编译器及应用程序使用的函数库(包括用户自己的库)。
- 编程复杂。由于 C 是编译型、中级、结构化语言，故用户在使用变量及函数前必须先进行定义。这要求用户对整个编程环境有相当的了解。
- 限制过少。前面所述 C 的灵活性，同时也可能变成编程的障碍，这取决于用户的编程是否严谨。例如，C 对数组越界，对未定义具体内存位置的指针进行读写，编译时均

认为是合法的，但却可使运行结果不可捉摸，甚至导致系统崩溃。

1.2.3 对比 ADS 与 AutoLISP

我们已清楚地看到，ADS 与 AutoLISP 有许多不同，也都有各自的利弊。如果对 C 并不熟悉，就要下决心学会它。如果没有或无法得到能与自己的 AutoCAD 平台共同工作的 C 编译器，则无法使用 ADS。如果只是编写一些简单的应用程供自己使用，并无将它们传播开来的念头，则 AutoLISP 可能会满足你的需要。

如果应用程序经常需要大量的人机交互，或图形数据库花费需要大量时间等待 AutoCAD 或用户反应，那么在这些应用中，代码的执行速度所带来的效率差别就微乎其微了。

如果应用程序为商业目的而编写，或需特殊的功能而 AutoLISP 无法实现（例如直接控制硬件），或代码的执行速度至关重要，则应选用 ADS。初始的时间耗费（熟悉系统）及购买编译器和连接器的开销会很快收到回报。

提示： 许多用户先用 AutoLISP 编写应用程序原以检验他们的思想，然后再将它们改写成用于传播的 ADS 应用程序。

注释： 大多数 C 编译器均包含自己的调试程序，具体的形式、用法及便利之处因编译器不同而异。Phar Lap 连接器也提供调试工具。这有助于修改、优化应用程序，直到用户满意。

1.3 使用 ADS 的条件

本书主要阐述 ADS 编程对用户及系统的要求。本书各部分的最前面一章均对各个专题下的 ADS 编程条件进行了讨论。

1.3.1 编程及软件知识

ADS 编程要求用户对编程及软件有一定的背景知识，以下是几个要点：

- 用户必须对应用程序所涉及到的 AuotCAD 技术有全面的了解。
- 用户必须精通 C 语言。ADS 大量地使用 C 语言的高级特性，例如联合（unions）、链表（linked lists）及指针（pointers）。
- 用户必须知道怎样使用 C 编译器及连接器（为 AutoCAD 所支持）。本章 1.5 节阐述 ADS 应用程序类型时将对编译器及连接器进行详细介绍。
- 用户必须理解 ADS 应用程序是如何集成到 AutoCAD 环境中去的，第二章将对此进行详尽剖析。

1.3.2 硬件环境

ADS 编程对硬件并无特殊要求，只要满足 AutoCAD 的使用要求即可。用户所使用的编译器可能对系统的内存有一定要求，但这些并不会超过 AutoCAD 对系统的要求。根据应用程序的性质，编译器会协调对内存的使用，然而，如果运行耗费内存的 ADS 应用程序——特别是有大量图形数据时，用户应考虑将系统的内存增加到超过运行 AutoCAD 所需的最少

内存。理论上内存越大越好。

1.3.3 软件环境

ADS编程需要几种软件的合作。首先是AutoCAD本身，如果想利用AutoCAD的某项特殊功能(例如AME)，则必须完整地安装这部分。其次需要AutoCAD支持的C编译器，如果使用WATCOM C以外的保护模式C编译器，则必须配Phar-Lap的连接器。最后，必须安装应用程序所需的所有库函数，它们分别由编译器(提供标准的库)及AutoCAD(提供ADS，ASI等库)提供。有关AutoCAD支持的编译器的信息请参见1.5节。

警告： 有些函数库AutoCAD只提供保护模式版本，如果用户没有保护模式的编译器及连接器，则无法编写相关的应用程序。

注释： 本书的第一部分假定用户已正确安装了ADS文件，若没有安装或未调通，请在阅读后面的材料前先完成这部分工作。

1.4 了解AutoCAD的ADS接口

ADS应用程序通过AutoLISP与AutoCAD及用户通信。下面将列出装入并执行ADS函数的步骤。这些步骤是一个基本的介绍，有关详尽信息将在2.1.4节讨论。如果这一系列步骤中某一步失败了，则余下的步骤也无法正常工作。

1. ADS应用程序被调用内存——通过XLOAD命令。
2. ADS应用程序通过ADS_INIT初始化与AutoLISP的通信。ADS与AutoLISP的通信由AutoCAD处理。
3. ADS应用程序使用ADS_LINK和新建立的通信链，告诉AutoLISP应用程序它已准备好接受函数请求。
4. 应用程序通过ADS_DEFUN定义自己的外部函数。外部函数是应用程序中定义的函数，它们可在命令行下执行或被AutoLISP应用程序调用。
5. ADS应用程序中的函数在需要时被调用并执行。
6. ADS应用程序通过XUNLOAD从内存中卸载。这时应用程序可通过ADS_UNDEF手工地取消自身外部函数的定义，如果需要的话还执行一些特定的操作，例如关闭文件或中止另一个应用程序。

1.4.1 与AutoLISP连接

ADS应用程序与AutoLISP的通信一旦建立，就允许在命令行或AutoLISP应用程序中调用ADS定义的外部函数。ADS应用程序必须通过ADS_INIT、ADS_LINK和一些以ADS打头的函数建立ADS和AutoLISP的通信。AutoCAD处理建立链的大部分细节，而通信的大部分细节则由处于后台的AutoCAD来处理。掌握了以上列出的编写ADS应用程序的核心步骤后，则可自由地编写应用程序的外部函数了。

提示： 作为编写ADS应用程序的指南，AutoCAD提供一个样板文件——TEMPLATE.C。TEMPLATE.C在AutoCAD目录下的ADS子目录中。在PT盘中将提供一个

CORE.C 的文件，它与 TEMPLATE.C 功能类似，为 ADS 应用程序的框架，只是更结构化一些。

1.4.2 运行 ADS 应用程序

在 ADS 应用程序调入内存且定义外部函数以后，用户可以通过命令行或 AutoCAD 应用程序调用 ADS 外部函数。

和 AuotLISP 一样，若想在 Command：提示下直接输入外部函数并执行它，则在定义 ADS 外部函数时，函数名前必须冠以“C:”。如果这样做了，则在 AutoLISP 应用程序中调入 ADS 外部函数时也必须在函数名前冠以“C:”。

在下面的例子中，名为“TEST”的 ADS 函数由 ADS_DEFUN 定义并被命令行及 AutoLISP 应用程序调用。这个例子只是一个示意，2.3 节将介绍一个完整的 ADS 程序。

```
ads_defun ("c:test",0);
Command: test ↵
(setq result (c:test))
```

1.5 ADS 应用程序的分类

ADS 应用程序既可是实模式的也可以是保护模式的应用程序。下面将对两种类型的 ADS 应用程序及相应的编译器进行介绍。

提示： AutoCAD 平台支持的编辑器种类在 README.ADS 中列出，这个文件在 AutoCAD 目录中。

README.ADS 中列出的是一般信息，在 AutoCAD 目录下的\ADS\DOCS 子目录中有关于特定编译器较详尽的介绍。

注释： 理论上任何标准的 ANSI C 编译器均可和 ADS 一块工作。但不同的 C 编译器由于自身的操作实现，有自己特定的方式。本章所列出的编译器是由 Autodesk 公司测试过的，并提供相应的函数库、头文件及文档资料。由于我们是为使用 AutoCAD 和 ADS 而选择编译器，故用户应使用 AutoCAD 支持的编译器。

1.5.1 实模式

实模式的应用程序为 16 位的，装载和运行均在常规内存中。实模式应用程序扩展名为 EXE，所使用的编译器是为 80286 或更高的处理器而设计的。AutoCAD12 支持下列实模式编译器：

- Borland C 2.0 或更高
- Microsoft C 5.0 或更高

AutoCAD 目录下的\ADS\DOCS 子目录中有关于编译器和连接器设置更进一步的信息。实模式 C 编译器均内置连接器，可直接生成运行代码。

在\ADS 子目录中，AutoCAD 提供执行 Borland C 和 Microsoft C 编译器的批处理文件。

表 1.1 列出了 AutoCAD386 c2 版的批文件，用户可以修改它们以加入自己的函数库或命令行参数。

表 1.1　实模式编译器批处理文件

批文件	编译器
MSC6 SAMP. BAT	Microsoft C
TBC3SAMP. BAT	Borland C

实模式编译器有以下优势和弊端：

优势

- 实模式编译器比保护模式版的更为便宜。
- 实模式编译器能得到广泛的供应与支持。
- 实模式编译器不需额外购买连接器(因为置有连接器)。
- 实模式的编译器有 ADS 文档及其他众多的第三方资料支持。

弊端

- 实模式应用程序无法使用扩展内存。由于常规内存有限(640K 左右)，限制了实模式应用程序的规模。
- 实模式应用程序是 16 位的，它无法充分利用 80386 和 80486 的 32 位体系结构。因此，实模式应用程序较保护模式版本的应用程序慢。
- AutoCAD 不提供实模式下的 AME 和 ASE API 函数库。
- 实模式应用程序所分配的内存为 AutoCAD 保护并管理的程度不如保护模式的应用程序。因为实模式应用程序对系统内存的配置很敏感。
- 必须调整 NISTACK 与 SAVER DOS 扩展开关。NISTACK 用于限制可同时打开的实模式应用程序个数。SAVER 将内存寄存器由保护模式转换成实模式，以便进行符号调试。请参见 REALMODE. TXT 设置扩展开关的信息。

1.5.2　保护模式

保护模式应用程序是 32 位应用程序，装载并运行于扩展内存中。产生保护模式代码的编译器是为 80386 或更高的处理器而设计的。保护模式的可执行程序具有扩展名 EXP。除 WATCOM C 外，保护模式编译器必须使用 Phar-Lap4. 1 或更高版的连接器。AutoCAD386 12 支持以下编译器。

- MetaWare High C 1. 5 或更高版本
- MetaWare High C/C++3. 0 或更高版本
- Zortech C++ 3. 0 开发版或科学工程版
- WATCOM C9. 0/386 或更高版本

警告： 不能用 Zortech C++3. 0 标准版，它只能产生 16 位的实模式代码。

注释： 如果使用 WATCOM C9. 0/386 并选择 Phar-Lap 连接器，请参阅 Phar-Lap Installation instructions for 386|DOS-Extender SDK 手册，以下列出须用 Phar-Lap 实用程

序 WLIB.EXE 来更新的有关 WATCOM 的文件手册。

在\ADS\DOCS 子目录下，有使用以上保护模式编译器的详尽文档——WATCOM.TXT，ZTC.TXT和 HIGHC.TXT。有关使用 Phar-Lap 连接器的信息，记录在\ADS\DOCS 子目录下的文档 PHARLAP.TXT 中。

在\ADS 子目录中，AutoCAD 提供执行保护模式编译器的批处理文件。表 1.2 列出了 AutoCAD386 c2 使用的批处理文件。用户也许需要编辑这些文件，以加入自己的函数库及命令行参数。

表 1.2 保护模式编译器批处理文件

批文件	编译器
W90SAMP.BAT	WATCOM C/386 9.0
Z30CAMP.BAT	Zortech 3.0
M30SAMP.BAT	High C 3.0
MAKESAMP.BAT	High C 1.7X

保护模式的 C 编译器有以下优势和弊端：

优势

- 保护模式应用程序使用护展内存，这使得保护模式应用程序运行相当迅速(内存量大，减少分页和覆盖的使用)。
- 保护模式应用程序为 32 位，充分发挥了 80386 或 80486 的 32 位体系结构的优势。应用程序的保护模式版比实模式版要快。
- AutoCAD 提供 AME 及 ASE API 的保护模式函数库。
- 保护模式应用程序分配的内存可被 AutoCAD 很好地保护和管理。如果为 AutoCAD 和保护模式应用程序提供足够的内存并进行良好配置，保护模式应用程序不会出问题。

弊端

- 保护模式编译器比实模式编译器贵。
- 保护模式编译器并不总能得到国际上广泛的提供和支持。
- 除 WATCOM C/386 外，所有保护模式编译器需购置 Phar-Lap 4.1 连接器。
- 对一般的应用程序来说，保护模式编译器不如实模式编译器应用广泛，缺乏第三方资料文档。

1.6 使用 C++

C++比标准 C 提供若干便利，例如封装、继承及重载。不幸的是，AutoCAD 不提供 C++的 ADS 库，因此用户定义的函数无法重载。用户仍可使用 C++编译器及标准 C++库，但许多 C++结构的优点无法用到。如果在 ADS 应用程序中使用 C++，则必须写自己的 ADS C++库。

1.6.1 ADS C++ 的未来

到本书写完为止，Autodesk 公司既没着手开发 ADS C++库，也没有打算为 12 版开发 C++库的 12 版。然而，Autodesk 花费相当的时间对 AutoCAD 及 AutoCAD 开发进行 C++测试。相信将来的 AutoCAD 会在更多方面对C++提供支持。

1.7 选择编译器

实模式编译器和保护模式编译器均有各自的利弊。实模式编译器相对便宜、简单、支持广泛，然而，它由于对内存敏感而限制应用程序规模。此外，无 AME 及 ASE API 库，这些都使得实模式编译器只适合于低档的选择。

由于高速及采用 32 位保护模式，无须附加 Phar-Lap 连接器，使得 WATCOM C/386 是编写 AutoCAD ADS 应用程序的理想选择。

注释： AutoCAD 并不提供 Zortech 编译器的 ASI 函数。

1.8 有关C语言

本书主要介绍如何使用 AuotCAD 开发系统将 C 函数集成到 AutoCAD 环境中，并非一本 C 语言专著。有些 C 编译器支持 ANSI 标准，有些支持传统的 K&R C。各种编译器又都应用自己的标准来选择函数。如果想保持对 AutoCAD 平台的可移植性，在编程中应尽量遵守 ANSI 标准。

有关 ADS 更详细的信息请查阅《AutoCAD Development System Programmer's Reference Mannual》。关于 ANSI C，请查阅第三方出版物，Autodesk 公司向您推荐《C Programming Language》(Prince Hall 1988)，第二版。该书讲述了 ANSI C 和 &R C。

1.9 小 结

通过本章的学习，我们了解了使用 ADS 的方法，也了解了使用 ADS 所需的各种条件，并且对不同的编译器有了明确的认识，知道如何选择适合自己的 AutoCAD 所支持的编译器及连接器。

下一章将介绍 ADS 应用程序各构件的结构，以及如何用它来组成一个 ADS 应用程序。

第二章 ADS 编 程

尽管 ADS 应用程序结构复杂，但 ADS 编程是有规律可循的。本章以下的内容对 ADS 应用程序的各组成部分进行剖析，并介绍如何用它们来构造一个完整的 ADS 应用程序。本章所构造的 ADS 应用程序形成了 ADS 应用程序的核心。有了这个可运转的核心，编制实用 ADS 应用程序实际上就简化为往这个核心外增加函数。

提示： AutoCAD 在 ADS 子目录中提供名为 TEMPLATE.C 的样板 ADS 程序，可做为 ADS 应用程序的内核。在 PT 盘中，CORE.C 是另一个 ADS 应用程序内核。

2.1 ADS 的 结 构

本节阐述组成 ADS 应用程序的各个部分。这些部分是组成 ADS 应用程序的基本构件。在提供的练习中将介绍如何利用它们来构造一个 ADS 应用程序。

2.1.1 准备

ADS 编程并不难，但这并不取决于用户自己的算法结构。由于对大多数 ADS 动作来说，许多事件必须正确地起作用并按正确的顺序发生，因此单个函数中须使用多个 RETURN。一般而言，C 程序在函数中不宜使用一个以上的 RETURN 命令。因为函数若使用了多个 RETURN 不便于进行调试，且影响函数流的延续，但在 ADS 中却必须如此。

用户可使用嵌套循环及 IF 判断来减少函数中 RETURN 的数量，但不幸的是 AutoCAD 操作的复杂性决定了用户代码的结构。用户可以避免退出函数，但无法避免在函数中使用多个 RETURN——哪怕是最简单的函数。注意到上面所述，用户会发现开发 ADS 变得相对容易了。

2.1.2 文头件和库文件

头文件和库文件在第一章已讨论过。所有的 ADS 应用程序必须包含 ADSLIB.H，而 ADSLIB.H 包含了 ADS.H、ADSCODES.H 及 STDLIB.H 几个头文件。另外，C 函数也许需要其他头文件，这取决于用户使用的 C 函数及具体的 C 编译器。API 应用程序要求 APILIB.H 头文件。使用对话框的应用程序须包含 ADSDLG.H。ASI 应用程序需包含头文件 ASI.H。

所有的应用程序必须与 ADS 库连接，而 API 及 ASI 应用程序还需要别的库文件。标准 C 的库文件取决于用户所使用的编译器。\ADS\DOCS 中有介绍库文件使用的文档。

2.1.3 ADS_INIT 和 ADS_LINK

第一章介绍过 ADS 函数 ADS_INIT 和 ADS_LINK。ADS_INIT 用来初始化与 Au-

toLISP 之间的通信。在 ADS_INIT 成功地执行后，ADS 应用程序每次与 AutoLISP 通信均使用 ADS_LINK。下面两小节将对每个函数的语法进行阐述。

2.1.3.1 ADS_INIT

ADS_INIT 初始化 AutoLISP 与 ADS 应用程序之间的通信。下面是 ADS_INIT 的语法：

```
int resbuf ads_init (argc,argv);
int argc;
int * argv[];
```

ADS_INIT 必须是 MAIN 函数中第一个被调用的 ADS 函数。形参 ARGC 和 ARGV 与 MAIN 函数的参数 ARGC 和 ARGV 相同。如果初始化成功，则 ADS_INIT 返回一个整数，任何值均表示成功。如果 ADS_INIT 不返回值，则 ADS 应用程序与 AutoLISP 之间的联系不能建立，应用程序中止执行。

下面的例子说明了如何使用 ADS_INIT：

```
void main(argc,argv)
int argc;
char *argv[];
{
  ads_init(argc,argv); /* Initialize the interface */
}
```

2.1.3.2 ADS_LINK

ADS_LINK 用于激活由 ADS_INIT 建立起的 ADS 应用程序与 AutoLISP 之间的通信。语法如下：

```
int ads_link (cbc);
int cbc;
```

ADS_LINK 向 AutoLISP 传递 ADS 应用程序的结果码，并返回 AutoLISP 向应用程序的请求码，应用程序请求码和结果码在后面详细讨论。参数 CBC 必须为表 2.2 列出的值之一。如果成功，ADS_LINK 从 AutoLISP 返回一个应用程序请求码；如果失败，则返回一个小于零的值。每次对 ADS_LINK 的调用，其目的在于响应上一次应用程序的请求码并得到下一次请求码。

下面的例子说明如何使用 ADS_LINK：

```
void main(argc,argv)
int argc;
char *argv[];
{
  int stat;   /* stat hold the status of the ADS to
                 AutoLISP link */
  int val;    /* val holds the function code */
  short scode=RSRSLT;   /* This is the default result code */
  ads_init(argc,argv);   /* Initialize the interface */
  /* repeat a while loop while the link from ADS to
```