

Programming AutoCAD 2000 Using ObjectARX

AutoCAD 2000 ObjectARX 编程指南

```
ObjectARX_EntryPoint  
extern "C" AcRx::AppRetCode  
crxEntryPoint(AcRx::AppMsgCode msg, void* pkt)  
{  
    switch (msg) {  
        case AcRx::kInitAppMsg:  
            // Comment out the following line if your  
            // application should be locked into memory  
            acrxDynamicLinker->unlockApplication(pkt);  
            acrxDynamicLinker->registerApp(Aware(pkt));  
            InitApplication();  
            break;  
        case AcRx::kUnloadAppMsg:  
            UnloadApplication();  
            break;  
    }  
    return AcRx::kRetOk;  
}
```

附 赠

(爱尔兰) Charles McAuley 著
李世国 潘建忠 平雪良 译

Autodesk设计技术丛书

AutoCAD 2000 ObjectARX

编程指南

(爱尔兰) Charles McAuley 著
李世国 潘建忠 平雪良 译



机械工业出版社
China Machine Press



ObjectARX 是 AutoCAD 2000 最强大的应用程序开发环境, 本书循序渐进地介绍了 ObjectARX 技术, 详细介绍了如何利用 Visual C++6.0 和 ObjectARX 2000 结合开发 AutoCAD 2000 实用程序, 满足 AutoCAD 定制要求。书中有大量的应用程序实例, 这些实例可有效地帮助读者掌握编程技巧。

本书英文版是 Autodesk 公司技术开发小组与 Autodesk 出版社联合制作的结晶, 它的出版受到 Autodesk 公司、出版社各方的高度评价, 一致认为是一本值得程序设计人员珍藏的好书。

Charles McAuley: Programming AutoCAD 2000 Using ObjectARX (ISBN 0-7668-0643-X).
Original copyright © 2000 by Thomson Learning. All rights reserved.

First published by Autodesk Press, an imprint of Thomson Learning, United States of America.

Authorized Chinese language (Simplified Chinese Characters) translation of the edition by Thomson Learning. No part of this book may be reproduced in any form without the express written permission of Thomson Learning and China Machine Press.

本书中文简体字版由汤姆森公司授权机械工业出版社独家出版。未经出版者书面许可, 不得以任何形式复制或抄袭本书内容。

版权所有, 侵权必究。

本书版权登记号: 图字: 01-2000-2059

图书在版编目(CIP)数据

AutoCAD 2000 ObjectARX 编程指南 / (爱尔兰) 麦克奥理 (McAuley, C.) 著; 李世国等译.
-北京: 机械工业出版社, 2000.7

(Autodesk 设计技术丛书)

书名原文: Programming AutoCAD 2000 Using ObjectARX

ISBN 7-111-08074-2

I. A... II. ① 麦... ② 李... III. 计算机辅助设计-应用程序, AutoCAD 2000 IV. TP391.72

中国版本图书馆 CIP 数据核字 (2000) 第 29728 号。

机械工业出版社 (北京市西城区百万庄大街 22 号 邮政编码 100037)

责任编辑: 刘立卿

北京市昌平第二印刷厂印刷·新华书店北京发行所发行

2000 年 7 月第 1 版第 1 次印刷

787mm × 1092mm 1/16 · 27.25 印张

印数: 0 001 - 5 000 册

定价: 58.00 元 (附光盘)

凡购本书, 如有倒页、脱页、缺页, 由本社发行部调换



译者的话

本书英文版是由Autodesk出版社出版发行的专门论述用Microsoft Visual C++开发AutoCAD应用程序的书籍。书中详尽介绍了用Visual C++ 6.0和ObjectARX 2000 SDK(软件开发工具包)结合开发在AutoCAD 2000环境中运行的ARX应用程序的基本知识和设计技术。

本书不同于ObjectARX 2000的联机帮助,其体系结构经过作者精心设计,内容选择合理,层次清楚,示例丰富。循序渐进地介绍了ObjectARX 技术,不仅非常适于自学,也可作为一本必备的参考手册。

在翻译过程中,我们查阅了大量的参考文献,对所涉及的术语尽量采用合适和易于理解的译法,以便于读者掌握本书所涉及的有关技术。下面对本书主要用到的有关术语说明如下:

| 原文 | 现有的中文译法 | 本书采用的译法 |
|--------------------------|--------------------|---------|
| Iterator | 浏览器、迭代器 | 浏览器 |
| Transaction | 事务、事件 | 事务 |
| Reactor | 反应器、响应器 | 反应器 |
| Named objects dictionary | 有名对象字(词)典、命名对象字典 | 有名对象字典 |
| MDI-Aware | MDI-Aware型、MDI-感知型 | MDI-感知型 |
| Transient reactor | 瞬时反应器、瞬态反应器 | 临时反应器 |
| Custom class | 定制类、自定义类 | 定制类 |
| Custom entity | 定制实体、自定义实体 | 定制实体 |

另外, ObjectARX 2000中提供了大量的虚拟函数(virtual function),在应用程序中使用时一般需要重定义(override)。有的书将override译为重载,实际上在C++中函数重载与函数重定义是两个不同的概念,前者为overload而不是override,因而,在本书中加以区分,以避免理解错误。

本书由无锡轻工大学CAD/CAE/CAM研究所组织翻译。参加本书翻译的有:平雪良(序、前言、第1章和第2章)、潘建忠(第3~5章)、李世国(第6~9章)。全书由李世国审校。

非常感谢机械工业出版社华章公司对我们的信任和本书责任编辑所付出的辛勤劳动。

由于译者水平有限,书中难免会有错译和疏漏之处,欢迎读者批评指正。

无锡轻工大学CAD/CAE/CAM研究所

李世国

2000年4月

DJS 04/10



序

Thomson Learning™的致词

Autodesk出版社是出版Autodesk公司学习软件及Autodesk系列产品的学习和培训材料的专业出版社。AutoCAD的功能如此强大，每一个使用它的人都能发挥他的潜能，这正是出版这套编程丛书设想的前提。这套丛书的书目涉及了AutoCAD最高级的主题，可以帮助读者最大限度地掌握AutoCAD，同时也向读者介绍了AutoCAD领域中最好的、最著名的作者。也许读者已经阅读过他们在CAD杂志上的专栏，或者在Autodesk公司组织的活动上听过他们的演讲，也许是第一次知道这些作者。不管什么情况，我们相信读者都会喜欢他们的作品，并将应用所学到的知识。感谢您选择这本书，并希望您在编程旅途中一切顺利。

Autodesk 出版社主编 Sandy Clark

Autodesk公司的致词

自从AutoCAD诞生以来，关于如何使用这种软件的出版物已经出版了很多，而如何定制这种软件的资料则相对较少。涉及AutoLISP的书籍极少，大多数AutoCAD的书籍只有一两个章节涉及到定制的内容。总之，有关开放性技术、应用程序编程接口(API)以及AutoCAD设计系统编程等，这些用于对AutoCAD进行开发方面的书籍的数量基本上不能满足用户的要求。

四年前，当我作为一个AutoCAD的API产品经理与开发者一起工作时，很快发现对ObjectARX™、AutoLISP、VBA以及AutoCADOEM书籍的需要量极大。出版编程系列丛书的初衷是因为AutoCAD是市场上最可编程的、最可定制及最可扩展的设计系统。据估计，有70%以上的用户立足于使用Visual LISP、VBA编程或定制菜单以提高效率。随着对提高设计者编程能力的要求的提高，对于提高自身编程和定制过程能力的新读物之需求也在随之提高。

AutoCAD的Autodesk标准文档提供了有关AutoCAD API技术的原始资料，然而，这种资料往往是在软件开发实验室中开发软件时用来描述所开发的软件的。真正实用的、具有一定深度和广度的文档则是通过成千上万软件开发者、消费者和用户根据特定的需要对系统加以理解和应用而完成的。

这样一来，对如何使用AutoCAD API的有关说明文档的要求就更高。这种开发技术指南需要改进，它可以提高编程者接口编程的能力，并能提供只有通过实际编程经验才能获得的接口编程用法上的细微差别。AutoCAD用户和开发者正在寻求学习的捷径和不同的参考资料。用户和开发者都想要提高他们的编程经验，使自己能尽快成为编程的行家并能更快更好地把精力集中于自己特定的定制开发项目上。更早、更快、更好地完成一个定制开发项目，意味着开发效率的提高，并能更快地将成果用于实际。

作为一个CAD管理者，通过加快学习过程来提高生产率意味着将会提高他(她)所在CAD

部门的生产率。对于一个专业开发者来说，这意味着将更快地把成果应用于市场，并保持在市场上的竞争力。

最初设想

1997年秋在洛杉矶Autodesk大学举办的一次活动引发了对AutoCAD API技术资料的需求。当时Bill Kramer正准备举办一个有关AutoCAD ObjectARX的会议，Autodesk大学计划有20~30人出席这次会议，并为此安排了相应的场地，大约在会议开始的前三周，我接到电话说来登记参加会议的人数已经达到了预定会议场地的容量，必须更换会议地点。由于前来参加会议的用户、CAD管理者、设计人员、公司设计管理人员甚至开发人员在不断增加，我们三次更改会议场所。参加会议的所有人员都想知道AutoCAD的API——ObjectARX如何提高他们自己的以及部门的设计生产效率。会议开始时，我走到会场后面，看到会场上至少有250人以上，大家只能站在那里。也许这正是我决定开始着手的时刻。

听众迫切地想知道Bill Kramer将如何评说ObjectARX的功能，接连不断的、意想不到的提问和解答持续了大约有一个小时。Bill成功地向参加会议的技术人员和非技术人员介绍了ObjectARX这一专题。这对于我来说是一个启示，我开始感兴趣于开辟与更多人进行沟通的新途径，将AutoCAD和API的技术问题、功能以及它们所带来的利益介绍给他们。

于是，我问Bill是否能写一本专门针对这次演讲专题的书籍。我很高兴地说，Bill的这本书成为了Autodesk出版社出版的关于AutoCAD编程丛书的第一批书籍中的一本。

关于编程丛书的主题

AutoCAD这一开放的可编程设计系统的程度使它早就需要有这样一套丛书，在Cynde Hargrave(AutoCAD市场高级管理人员)的领导下，我们一个小组开始在Autodesk出版社为开发这一丛书而工作。

这是一个令人兴奋的计划，致力于这一丛书的作者有很多，他们所从事的领域覆盖了从关于ObjectARX的AutoCAD开放内核技术到关于VBA应用程序开发技术的Windows标准。其结果就形成了一个完善Autodesk编程系列资料库，它们覆盖了所有以下技术，包括ObjectARX、Visual LISP、AutoLISP，利用ActiveX Automation和微软的VBA来定制AutoCAD、AutoCAD数据库的关联性以及AutoCAD通用定制方法。

谁阅读这套编程丛书

每一个AutoCAD用户都能从这套系列丛书中找到适用于自己的、感兴趣的、有关AutoCAD定制和开发的书。我们这个小组和Autodesk出版社合作出版这一丛书，确定每一本书的书名以及合适的作者，目的是为不同兴趣和不同经历的读者提供更广泛的有关AutoCAD定制的内容。

Autodesk出版社和Autodesk公司共同开发了一套编程丛书，这套丛书覆盖了所有重要的API和AutoCAD定制的主题。另外，丛书还为不同层次的读者提供更多的信息，无论是刚开始学习定制AutoCAD的初学者，还是那些开发更深层次的AutoCAD应用软件及为增加自己的编程经验而寻求参考资料的专业程序编制人员，都可以从中得到有用的信息。

Bill Kramer的致词

开发一种新技术需要时间，而使这种新技术真正对他人有用，又需要一些非常特殊的人才。本书作者Charles就是这种天才人物之一，他具备专业技术能力和向别人传递信息的能力，这些是将最新技术转化成让人容易接受的读物所不可缺少的。

我得到了在著作完成之前对著作进行审阅的唯一机会。我的任务是给本书增加一些附录，并在AutoCAD R14和AutoCAD 2000两个版本的文本和例子之间帮助找出一些差异。在阅读这著作的初稿时，我一直在想，如果我是初次接触ObjectARX，这本书对我将是多么有用。

这本书给人的第一个印象是，它是一本值得任何严肃认真的程序设计人员收藏的好书，包括我自己。书中的例子和文本反映了许多应用程序开发的真实事例，而不是那些大多数计算机科学家们在不知读者兴趣所在时描述的理论性内容。

ObjectARX是一门综合性的学科，要充分地利用ObjectARX需要有编程和计算机辅助设计的坚实基础。更进一步讲，编程者必须对AutoCAD有相当的了解，要知道如何操作AutoCAD这一软件产品。ObjectARX给编程者一个能对AutoCAD进行深层次开发的控制权，因此刚开始给编程者的印象是，要掌握并应用ObjectARX是一件比较困难的事。但实际上并非如此，Charles在说明怎样使用ObjectARX以解决AutoCAD的实际应用时做了非常出色的工作。

本书的实例很棒，对于我来讲，这里任何有关编程语言或编程环境的内容以及许多很好的实例都是很重要的。本书有许多好的实例，有些特别的实例对ObjectARX编程初学者，甚至于对系统深层次的开发都有很大的帮助。

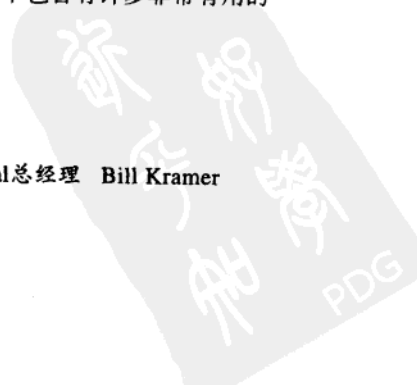
为什么这本书能有这样的效果？这是因为Charles作为一个教师，他是用自己的经验编写了本书。他把复杂的概念降低到了能让人理解的程度来进行解释。如使用MFC、调试器以及AutoCAD实体对象和反应器等内容，他通过能让C++编程者更清楚地理解的方式来解释和论证。

请注意，读者应该对编程，特别是用C和C++语言编程比较熟悉。虽然有坚实的AutoLISP和Visual Basic基础对学习本书是一个良好的开端，但对有关C/C++的基本概念应该在阅读本书之前就掌握。你不必一定是个C/C++的编程大师，书中的例子对所有的对学习C、C++有足够兴趣的编程者是有用的。为了更好地掌握本书，读者所需要的是有一定的应用程序开发经历和掌握AutoCAD的使用方法。

在阅读本书的时候，应该多花一点时间在理解本书的实例上。这些例子是掌握如何编写应用程序的好老师，但千万不要跳过枯燥的叙述而浏览例子，叙述中包含有许多非常有用的见识。

坚持编程是很重要的一点。

Auto-Code Mechanical 总经理 Bill Kramer



前 言

欢迎使用用于AutoCAD 2000的ObjectARX，ObjectARX是AutoCAD 2000最强大的应用程序开发环境。本书的目的是让读者全面掌握ObjectARX的基础，而不是ObjectARX的全部。我所要传授给读者的是ObjectARX是如何工作的，读完本书后你能够领略到ObjectARX的深奥之处。希望能帮助你渡过学习开发AutoCAD 2000应用程序最艰难的阶段。

本书阅读对象

一开始我就指出本书不是为初学者而写的，希望你有一定的开发经验并且达到了一定的水平。首先应该懂得C++：需要知道什么是类，什么是虚拟和重载函数，以及如何从一个基类中派生一个类，本书不是一本关于C++的指南，如果遇到陌生的概念，你或许需要进修课程。还需要熟悉Visual C++ 6.0，并且懂得如何使用应用程序向导和类向导，需要知道如何使用调试器。如果要利用MFC(微软基础类)的对话框以及AutoCAD/ObjectARX 2000 UI扩展，你应该知道MFC以及消息映射。当然不一定要掌握MFC的所有内容，对本书而言，能掌握MFC的对话框、控件以及消息映射就够了。

至于AutoCAD，显然应该懂得如何使用。你不一定对AutoCAD掌握得很深，基本的要求是知道如何创建实体，要知道层、线型、块、标注尺寸、文本以及文本样式等，当然知道得越多越好。如果你以前有一定的AutoCAD编程经验最好，但这不是必须的。

ObjectARX是一个综合的、广泛的API。确认你对AutoCAD 2000应用程序开发的需求：如果不打算去创建一个定制对象或实体，那么Visual LISP或者VBA也许对你比较合适，学习这些开发环境相对要简单一些，当然功能没有ObjectARX强大。首先对自己的需求有个评价，然后决定什么时候、什么样的API能符合需要。

本书没有涉及到的内容及其原因

正如在以前所提及的，本书不可能包括ObjectARX 2000的所有内容，有一些问题将不包括在本书中：

因为本书是关于C++和ObjectARX的，不包括COM或ATL的内容，因此本书没有关于“对象特性管理器(Object Properties Manager)”的内容，介绍这一主题的文档可以从ADN(Autodesk Developer Network)网页上得到，网页上有相关的详细材料。

- 本书不包括“设计中心”，可以从如下的文件夹中得到一些有关设计中心的例子：

```
<drive>:\ObjectARX 2000\docsamps  
<drive>:\ObjectARX 2000\samples
```

- ObjectARX 2000的UI扩展的有些内容也不包括在本书中，可以从以上的文件夹以及下面这个文件夹中找到实例：

```
<drive>:\ObjectARX 2000\samples\mfcsamps
```

- 所有关系主题和“深层克隆”(“Deep Cloning”)在“AutoCAD 2000 ObjectARX开发人

员指南”中谈到,有关如“AutoCAD 2000的克隆”的文档可以从ADN网页上获得:

<http://www.autodesk.com/adn>

- 本书没有专门讨论MDE(Multiple Document Environment, 多文档环境)内容的章节,但第6、7、8、9章中涉及到了有关MDE/MDI的实例。

阅读本书需要准备些什么

首先,必须要有微软的Visual C++ 6.0 SP2版,并且在计算机上要装有ObjectARX 2000。还需要装有Windows NT 4.0以便调试你的应用程序, ObjectARX应用程序可在Windows 95 或Windows 98上运行,但不可以在这种环境中调试它们。在编写本书的同时,我们正在等待即将发布的Windows 2000,目前Autodesk正在这一环境中测试ObjectARX。

光盘上的内容

可从光盘上的Code文件夹下找到有关AutoCAD/ObjectARX 2000的代码,以下是组成本书各个章节的代码的编排,有关组成本书的应用程序示例参照2.8节“本书内容概述”。

第1章

| | |
|--------|-------------------|
| Hello1 | acrxEntryPoint() |
| Hello2 | 从AcRxObject中派生一个类 |

第3章

| | |
|-------|--------------|
| Ch3_1 | 数据类型、用户输入和命令 |
| Ch3_2 | 选择集 |
| Ch3_3 | 选择集 |

第4章

| | |
|-------|--------------|
| Ch4_1 | 层表记录和浏览器 |
| Ch4_2 | 创建实体 |
| Ch4_3 | 交点和偏移 |
| Ch4_4 | 对选择集改变层 |
| Ch4_5 | 创建LWPOLYLINE |
| Ch4_6 | 插入块 |
| Ch4_7 | 插入带属性的块 |
| Ch4_8 | 创建带属性的块 |

第5章

| | |
|--------|-----|
| Ch5_1 | 几何类 |
| Ch5_1a | 几何类 |

第6章

| | |
|-------|----------|
| Ch6_1 | 加载DCL对话框 |
| Ch6_2 | DCL对话框 |
| Ch6_3 | 隐藏DCL对话框 |
| Ch6_4 | 嵌套DCL对话框 |

第7章



| | |
|------------|------------------------|
| Ch7_1 | 模式对话框 |
| Ch7_2 | 隐藏模式对话框 |
| Ch7_3 | 无模式对话框 |
| Ch7_4 | 标签式对话框 |
| Ch7_4MFC | MFC标签式对话框 |
| Ch7_5 | 向导式对话框 |
| Ch7_6 | MFC工具条 |
| Ch7_7 | 单一资源DLL模式对话框 |
| Ch7_7Res | 单一资源DLL |
| 第8章 | |
| Ch8_1 | 扩展实体数据 |
| Ch8_2 | 扩展字典 |
| Ch8_3 | 有名对象字典 |
| Ch8_4 | 定制对象 |
| Ch8_5 | 定制实体——ObjectDBX 应用程序 |
| Ch8_5UI | 用于Ch8_5的定制实体用户接口 |
| 第9章 | |
| Ch9_1 | |
| Ch9_1UI | 事务处理程序 |
| DebugDBX | 调试版CH9_1UI的ObjectDBX文件 |
| ReleaseDBX | 发行版CH9_1UI的ObjectDBX文件 |
| Ch9_2 | |
| Ch9_2UI | 临时反应器 |
| DebugDBX | 调试版CH9_2UI的ObjectDBX文件 |
| ReleaseDBX | 发行版CH9_2UI的ObjectDBX文件 |
| Ch9_3 | 永久反应器ObjectDBX 应用程序 |
| Ch9_3UI | 用于Ch9_3的永久反应器用户接口 |

对于正在使用AutoCAD R14和ObjectARX 2.02的用户，在一个名为R14 Code的文件夹中有一组与之相对应的应用程序，其格式和指令与AutoCAD/ObjectARX 2000 的相类似。AutoCAD R14的代码是由Visual C++ 5.0创建的。



目 录

| | |
|---|----|
| 译者的话 | |
| 序 | |
| 前言 | |
| 第1章 ObjectARX入门 | 1 |
| 1.1 引言 | 1 |
| 1.2 如何选择Visual LISP、VBA或 ObjectARX 2000进行AutoCAD编程 | 1 |
| 1.2.1 AutoLISP/Visual LISP | 1 |
| 1.2.2 用于应用程序的Visual Basic | 2 |
| 1.2.3 ADS | 2 |
| 1.2.4 Object ARX | 2 |
| 1.3 选用哪一种Visual C++版本 | 2 |
| 1.4 为什么要学MFC | 3 |
| 1.5 动态链接库 | 3 |
| 1.6 ARX入口点函数: AcrxEntryPoint() | 4 |
| 1.7 创建第一个ObjectARX应用程序 | 6 |
| 1.7.1 创建DLL | 6 |
| 1.7.2 在Visual C++工程中插入文件 | 7 |
| 1.7.3 修改工程设置 | 8 |
| 1.8 运行Hello1 ARX程序 | 10 |
| 1.9 ARX应用程序HELLO1的成员 | 13 |
| 1.10 用定制类创建一个ARX应用程序 | 14 |
| 1.10.1 用从AcRxObject中派生出来的类创 建一个应用程序 | 16 |
| 1.10.2 运行Hello2 ObjectARX应用程序 | 19 |
| 1.10.3 ARX应用程序HELLO2的成员 | 20 |
| 1.11 ObjectARX 2000向导 | 24 |
| 1.12 辅助资源 | 24 |
| 第2章 ObjectARX环境与本书综述 | 26 |
| 2.1 ObjectARX库 | 26 |
| 2.2 AcRx库 | 26 |
| 2.3 AcEd库 | 27 |
| 2.4 AcDb库 | 27 |
| 2.5 AcGi库 | 28 |
| 2.6 AcGe库 | 28 |
| 2.7 ADSRX库(以前的ADS) | 28 |
| 2.8 本书内容概述 | 28 |
| 第3章 ADSRX(ADS)基础 | 32 |
| 3.1 ADS的历史 | 32 |
| 3.2 ADS中定义的变量、类型和常数值 | 32 |
| 3.3 ADSRX中的AutoCAD命令 | 36 |
| 3.4 向用户发送信息 | 38 |
| 3.5 应用程序实例CH3_1 | 39 |
| 3.6 应用程序实例CH3_1要点分析 | 45 |
| 3.7 获取用户信息 | 46 |
| 3.8 在AutoCAD中选择实体 | 55 |
| 3.9 应用程序实例CH3_1要点的进一步 分析 | 57 |
| 3.10 选择集 | 58 |
| 3.10.1 选择集的过滤 | 61 |
| 3.10.2 选择集的关系过滤 | 62 |
| 3.10.3 选择集的条件过滤 | 63 |
| 3.10.4 选择集的扩展实体数据过滤 | 64 |
| 3.10.5 转换矩阵和选择集 | 64 |
| 3.10.6 选择集的操作 | 66 |
| 3.11 数据类型转换函数 | 68 |
| 3.12 应用程序实例CH3_2 | 70 |
| 3.13 应用程序实例CH3_2要点分析 | 75 |
| 3.14 应用程序实例CH3_3 | 75 |
| 3.15 应用程序实例CH3_3要点分析 | 79 |
| 第4章 AutoCAD的数据库和实体结构 | 80 |
| 4.1 符号表 | 81 |
| 4.2 有名对象字典 | 81 |
| 4.3 数据库对象的初始状态 | 82 |
| 4.4 实体句柄、对象ID号和多重数据库 | 82 |
| 4.5 创建对象 | 83 |
| 4.6 用ObjectARX创建对象 | 84 |
| 4.7 对象的打开和关闭 | 87 |

| | | | |
|--|-----|---|-----|
| 4.8 在组字典中加进一个组 | 87 | 第6章 DCL对话框 | 200 |
| 4.9 AutoCAD的数据库结构 | 88 | 6.1 对话框的组成 | 200 |
| 4.10 数据库常驻对象 | 89 | 6.1.1 预定义的活动控件 | 203 |
| 4.11 常用返回码 | 92 | 6.1.2 预定义控件的属性 | 212 |
| 4.12 符号表函数 | 94 | 6.1.3 布局和尺寸属性 | 214 |
| 4.13 应用程序实例CH4_1 | 98 | 6.1.4 功能属性 | 214 |
| 4.14 应用程序实例CH4_1要点分析 | 105 | 6.1.5 关键字属性 | 215 |
| 4.15 AutoCAD的实体类 | 107 | 6.2 DCL语法 | 215 |
| 4.15.1 AcDbEntity类查询函数 | 107 | 6.3 应用程序实例CH6_1 | 217 |
| 4.15.2 AcDbEntity类编辑函数 | 109 | 6.4 常用的对话框函数 | 222 |
| 4.15.3 AcDbEntity类杂函数 | 111 | 6.5 单文档界面和部分注意事项 | 224 |
| 4.15.4 AcDbCurve类 | 111 | 6.6 应用程序实例CH6_2 | 226 |
| 4.15.5 AcDbLine类 | 114 | 6.7 隐藏对话框 | 235 |
| 4.15.6 AcDbCircle类 | 115 | 6.8 应用程序实例CH6_3 | 235 |
| 4.16 应用程序实例CH4_2 | 116 | 6.9 嵌套对话框 | 249 |
| 4.17 应用程序实例CH4_2要点分析 | 121 | 6.10 应用程序实例CH6_4 | 253 |
| 4.18 应用程序实例CH4_3 | 122 | 第7章 MFC对话框和ObjectARX的用户 界面扩展 | 255 |
| 4.19 应用程序实例CH4_3要点分析 | 129 | 7.1 工程设置 | 255 |
| 4.20 ObjectARX的集合类 | 131 | 7.2 资源 | 256 |
| 4.21 应用程序实例CH4_4 | 132 | 7.3 模式对话框函数和应用程序实例 CH7_1 | 257 |
| 4.22 AutoCAD的复杂实体 | 135 | 7.4 模式对话框函数和应用程序实例 CH7_2 | 268 |
| 4.23 应用程序实例CH4_5: AcDbPolyline | 136 | 7.5 无模式对话框函数和应用程序实例 CH7_3 | 285 |
| 4.24 应用程序实例CH4_6: AcDbBlockReference | 140 | 7.6 标签式对话框和应用程序实例CH7_4 | 288 |
| 4.25 应用程序实例CH4_7: AcDbBlockReference 和AcDbAttribute | 143 | 7.7 向导式对话框和应用程序实例CH7_5 | 296 |
| 4.26 应用程序实例CH4_8: AcDbBlockReference 和AcDbAttribute | 147 | 7.8 通用控件和应用程序实例CH7_6 | 303 |
| 第5章 ObjectARX的几何类 | 152 | 7.9 单一资源动态链接库和应用程序实例 CH7_7 | 319 |
| 5.1 几何基本类概述 | 152 | 7.10 其他示例资料 | 322 |
| 5.1.1 AcGePoint2d | 152 | 第8章 定制类、实体和ObjectDBX | 324 |
| 5.1.2 矩阵操作 | 154 | 8.1 扩展实体数据和应用程序实例 CH8_1 | 324 |
| 5.2 AcGe二维实体类 | 156 | 8.2 扩展字典和应用程序实例CH8_2 | 334 |
| 5.2.1 AcGeEntity2d | 156 | 8.3 有名对象字典和应用程序实例CH8_3 | 339 |
| 5.2.2 AcGeCurve2d | 157 | 8.4 派生于AcDbObject的定制类和应用 程序实例CH8_4 | 342 |
| 5.2.3 AcGeLinearEnt2d | 161 | | |
| 5.2.4 AcGeCircArc2d | 163 | | |
| 5.2.5 AcGe三维实体类层次 | 165 | | |
| 5.3 应用程序实例CH5_1 | 166 | | |

| | | | |
|--|-----|--------------------------------------|-----|
| 8.5 ObjectDBX | 355 | 9.3 反应器和通知 | 397 |
| 8.6 派生于AcDbEntity的定制实体和应用 程序实例CH8_5 | 356 | 9.4 临时反应器、通知和应用程序实例 CH9_2UI | 397 |
| 8.7 其他示例资料 | 382 | 9.5 永久反应器、通知和应用程序实例 CH9_3UI | 412 |
| 第9章 事务、反应器和通知 | 383 | 9.6 其他示例资料 | 422 |
| 9.1 事务 | 383 | 9.7 结束语 | 423 |
| 9.2 事务和应用程序实例CH9_1UI | 384 | | |



第1章 ObjectARX 入门

1.1 引言

本章将全面介绍在设计 and 实现基于AutoCAD 2000的应用程序时对开发者十分有用的技术。就计算机技术而言, Autodesk(AutoCAD的开发者)是一家老牌公司, 从80年代初AutoCAD R1.0发布以来, 到现在发生了许许多多的变化。就易用性和API功能性来讲, AutoCAD 2000显然是目前最好的版本。DOS的时代已经过去, AutoCAD现在已经只适用于Win32平台(Windows 95/98和Windows NT)。AutoCAD现在是一个面向对象的CAD应用软件, 它具有令人难以置信的强大功能, 它能向用户和开发者提供各种应用程序的开发方式选择。本书将向读者介绍各种定制和开发AutoCAD应用程序的方法, 介绍Visual C++、微软的基础类库(MFC)、动态链接库(DLL)以及动态链接库的AutoCAD版本——ARX。还要创建一些应用程序, 并介绍它们的原理。最后, 将向读者介绍如何对AutoCAD应用程序使用调试器。

1.2 如何选择Visual LISP、VBA或ObjectARX 2000进行AutoCAD编程

AutoCAD很早就是一种开放体系结构的应用程序, 是一种通用的CAD应用软件, 用户和开发者可以利用各种方法定制AutoCAD以满足他们各自的需要。改变AutoCAD的工作方式可以分为两大类, 即定制和编程。很多AutoCAD用户通过创建新的宏代码、新的工具条或者用DIESEL(可直接解释求值的字符串表达式语言)来编制状态栏、创建智能菜单和菜单组来定制AutoCAD的菜单。他们也可以创建一些预置了各种层、文字样式、尺寸样式、线型等的样图来使操作简化和更有效率。以上的每一种都是定制的例子。有许多人则在定制AutoCAD和学习如何进行AutoCAD编程之间徘徊(我曾经就是这样的)。那么, 为什么要进行AutoCAD编程? 也许其主要原因是要求灵活性和扩充功能。AutoCAD在定制上相当灵活。可以告诉AutoCAD需要做什么, 要它做到怎么样, 然后它就可以按照要求忠实地一遍又一遍地做你所希望做的事。然而如何进行AutoCAD编程? 这里有多种选择。

1.2.1 AutoLISP/Visual LISP

如果你像我和大多数人一样, 是从AutoLISP开始的, AutoLISP是早期的AutoCAD版本引入的。当我开始用AutoCAD R9版本开始进行编程时, AutoLISP已经很时兴了。什么是AutoLISP? 它是LISP编程语言的一个子集, LISP是指表指令集处理。其中很重要的一点是: 括号必须是成对使用, 有一个括号“(”, 就必须要有一个括号“)”与之相对应, 当时有一句很有感触的话就是“沉浸在乏味的圆括号中”。AutoLISP是一种编程语言, 包含有许多像C和C++那样的基本元素, 但并不会那么令人感到十分头痛。AutoLISP是一种解释编程语言, 所以不必对它进行编译。所有的程序代码均是用文本编辑器编写的, 并且都有一个相同的后缀.LSP。(大家一定还记得行编辑命令EDLIN吧!)因为AutoLISP是一个解释程序, 其运行速度要比C++(ObjectARX)来得慢, 但要画一大张图时要比手工画图快得多, 这是我在许多场合

下向我的学生这么说的。AutoLISP的优点之一是因为它不需要编译，可以运行于任何平台。但是，因为现在的AutoCAD版本只考虑支持一个平台，所以这已不再成为一个优点。AutoLISP的另外一个优点是已经有很长的历史，有许多例程是用AutoLISP来写成的，不管需要什么，都会有机会找到已经有人写过这类程序。那么，如何去找到这类程序呢？可以到Autodesk的网站上去：www.autodesk.com，有两本杂志《Cadalyst》和《Cadence》也提供LISP的程序集，可以从它们的网站上获得。如果以前从未编过程，AutoLISP友好的编程环境一定很适合你，并可帮助你熟悉编程结构。

1.2.2 用于应用程序的Visual Basic

AutoCAD 2000包含VBA(Visual Basic for Applications)形式的Visual Basic，我对Visual Basic知道得不多，但我知道在AutoCAD 2000中已实现了。如果已经熟悉了Visual Basic或VBA，这种途径还是值得去探索的，尤其是当你不熟悉C++的情况下。据我所知，VBA唯一不能做到的是不能用它来定制AutoCAD对象。

1.2.3 ADS

AutoCAD开发系统ADS(AutoCAD Development System)是在OS/2操作系统下的AutoCAD R10版本引入的。ADS编程由C语言和提供给开发者用来创建应用程序的C程序库组成。ADS是随着DOS版AutoCAD R11介绍给大众的。ADS现在已经过时，且不再延续。这也不完全正确——它是以ADSRX的形式被集成到ARX中。如果准备编写AutoCAD的应用程序，我建议使使用ObjectARX。由于在ObjectARX中现在还保留有ADS的成分，如果在使用ARX过程中遇到有关ads_XXX的名字也不要感到奇怪，它们现在已被完全集成到ObjectARX中。我在学习ObjectARX、C++和MFC时，也曾学习过C和ADS，根据经验，我可以告诉你C++要比C好得多，ObjectARX要比ADS好得多，MFC/ObjectARX的UI扩展可以说是锦上添花。

1.2.4 ObjectARX

ObjectARX中的ARX表示AutoCAD运行时扩展(AutoCAD Runtime eXtension)，在AutoCAD R13中称为ARX。AutoCAD R14发行后，Autodesk公司发布了ObjectARX 2.0，它用于AutoCAD R14的进一步改善和提高。后来的ObjectARX 2000则是针对AutoCAD 2000版本的。

ObjectARX 2000是一个综合的API(Application Programming Interface)，它包含有220个类和3000多个独一无二的成员函数。尽管有很多内容，但在创建应用程序时不一定都要掌握它们。有许多类和函数都是按类型分组，有一些子集将经常使用。现在让我们来看看Visual C++、MFC和AutoCAD。

1.3 选用哪一种Visual C++版本

对AutoCAD 2000的ObjectARX 2000，需要Visual C++ 6.0以及SP2(软件补丁2)，也需安装Windows NT 4.0。请注意，创建的所有应用程序都能在各种版本的Windows环境下运行，但是，为了调试应用程序，需要用Windows NT 4.0。如果已经安装了Windows 95/98，也可以在同一机器上再安装Windows NT 4.0。

1.4 为什么要学MFC

为什么应该学MFC? 什么是MFC? MFC表示微软基础类库(Microsoft Foundation Classes), 是与Windows SDK和API一起打包的。在过去的Windows编程中, 应用程序是用C和Windows SDK一起开发的。到了1992年, 微软发布了Microsoft C/C++ 7.0版, 在该版本中包含了MFC 1.0版本。在此版本以后, 微软发布了Visual C++ 1.0, 现在发展到Visual C++ 6.0版, MFC也随着Visual C++的版本一起推向市场。过去用C和SDK编制的Windows应用程序会使人难以理解。我开始从事C++编程是从Visual C++1.0开始的, 因此我几乎是一开始就学MFC。MFC提供类、函数(方法)和数据成员, 这些都是与Windows API一起打包的。当使用MFC时, 它调用API来实现所有需要Windows来完成的工作。但这并不是完全不需要API, 我对API也有了一点了解, 有时也直接用它来处理某些特殊问题, 而这些往往是MFC所没有提供的, 当然这种情况很少发生。

那么, 为什么AutoCAD开发人员应该使用MFC呢? 对我而言, 也许最好的理由是因为MFC有一个用于创建对话框的类——CDialog。毫无疑问, Windows提供的对话框大大优于DCL(Dialog Control Language)对话框, DCL可以让你创建一个可以在任何平台下运行的对话框, 但我们现在只涉及到一个平台(Win32), 我想DCL最终将会被淘汰。在后面的章节中将较详细地讨论DCL的内容。CDialog类不但提供各种类型的对话框, 还可让你在对话框中使用任何Windows控件, 包括ActiveX控件。MFC也提供DDX(Dialog Data eXchange)和DDV(Dialog Data Validation)来控制对话框中的各种成员。DDX用于在对话框数据和成员变量中传递数据, DDV则是用来进行数据的校验, 包括数据的范围检查。DCL还将继续使用。如果懂得MFC, 也就能充分发挥ObjectARX 2000的UI扩展类的长处。另外, 在本书中还会看到许多引人注目的MFC属性。

1.5 动态链接库

如果浏览Windows的各个目录, 就会发现有許多文件的扩展名为.DLL, 这就是动态链接库文件。常用函数以及某些对话框(尤其是Windows通用控件)可以用一种库的形式即DLL文件来保存。DLL是当应用程序在调用它时在运行时加载的, 与调用程序的代码是相互独立的。如果在DLL中增加函数或改变代码, 只需编译DLL文件, 而不必对调用它的应用程序重新编译。用一个新的DLL文件来代替现有的DLL文件后, 应用程序还是和以前一样调用它。所以DLL是通用函数或对话框的仓库, 可以被多个应用程序调用。例如, 如果两个Windows应用程序调用一个保存文件对话框, 在两个应用程序中出现的对话框是完全一样的, 因为它们调用的是同一个DLL文件中的对话框。对于应用程序来说, 当从一个DLL中调用一个函数时, 这个函数就相当于应用程序中的一部分。然而, 当创建或改变这个DLL时, 没有必要因为应用程序使用DLL而改变应用程序的源代码。DLL独立于调用它的应用程序。这就是用ObjectARX创建应用程序时AutoCAD所采用的方法: 不必改变AutoCAD的内部代码。AutoCAD不用.DLL这个后缀, 而是用.ARX作为后缀, 但AutoCAD所涉及的仍然是DLL, 只是把它们当作AutoCAD的一部分而已。

在Visual C++中使用三种类型的动态链接库: Regular DLL(常规的动态链接库)、Regular DLL dynamically linked to MFC(与MFC动态链接的常规动态链接库)、Extension DLL(扩展的

动态链接库)。按ObjectARX 2000向导(有关向导的更多内容将在以后详述)的惯例,本书的大多数应用程序都是使用扩展的动态链接库。在创建MFC应用程序时,采用扩展的动态链接库。注意,不一定非要使用扩展的动态链接库,但为什么要有扩展的动态链接库,它的关键在哪里?在C++中创建类,这些类具有方法(函数)、属性和数据,然后将类添加到你的工程中,重新连编应用程序,应用程序又可以使用新的类。使用扩展的动态链接库,可将所有的类生成一个扩展的动态链接库,将它加入到你的工程中,使用它就像是应用程序中的一部分一样。这样做非常方便,我曾经几次采用这种方法。另外两种动态链接库,常规的动态链接库和与MFC动态链接的常规动态链接库,是自身解释的。我提到过ARX应用程序是动态链接库,ARX(DLL)应用程序也称为其他动态链接库的服务程序。

在ARX应用程序中使用外部DLL的服务程序的原因是国际化软件开发或在调用时本地化。可以创建一个资源文件DLL,将所有的位图文件、对话框和字符串放到DLL文件中。如果要为你的应用程序创建一个外文版本,只要将合适的DLL文件放到DLL目录下就可以了。我们将在本书的后面讨论有关本地化的问题。

关于DLL还有一点需要说明,DLL没有ADS应用程序那样的main()函数。从某种意义上来说,DLL不是应用程序,它不可以单独运行,确切地讲,它是一个服务程序。在AutoCAD中,应用程序调用由DLL提供的服务程序,而应用程序必须知道该服务程序的入口点。ARX应用程序的入口点是acrxEntryPoint函数,我们将在下面讨论该入口点函数。

1.6 ARX入口点函数: AcrxEntryPoint()

让我们来看看在Hello1.cpp文件清单中的acrxEntryPoint函数(以后我们将在这个文件中增加更多的内容):

```
#include <aced.h>
#include <rxregsvc.h>

void initApp();
void unloadApp();

void helloWorld();

void initApp()
{
    // register a command with the AutoCAD
    command mechanism
    acedRegCmds->addCommand("HELLOWORLD_COMMANDS",
                           "Hello",
                           "Bonjour",
                           ACRX_CMD_TRANSPARENT,
                           helloWorld);
}

void unloadApp()
{
    acedRegCmds->removeGroup("HELLOWORLD_COMMANDS");
}

void helloWorld()
{
    acutPrintf("\nHello World!");
}

extern "C" AcRx::AppRetCode
```