计算机维护 COMPUTER MAINTENANCE

在 ObjectARX 程序中动态添加和删除 AutoCAD 菜单命令

刘良华 袁英战

摘 要 本文利用 AutoCAD 2000 提供的类型库，编写了一个 ARX 例程。该例程在加载时将应用程序相关的菜单项添加到指定的菜单中，而在卸载时又将这些定制的菜单项全部删除。本文为 AutoCAD 主程序界面与应用程序功能紧密地结合在一起提供了新的方法和思路。

关键字 AutoCAD 二次开发，定制菜单，COM，ObjectARX

菜单是 Windows 应用程序标准的用户界面元素，其可视化的操作界面让使用者操作起来既简洁又方便。一般而言，AutoCAD 二次开发人员都有这样一种体会，即对于开发完成的大量应用程序模块，往往通过定制相应的菜单命令来执行之。通过菜单命令执行应用程序模块，用户不必牢记烦琐的文本命令。与以前的版本一样，AutoCAD 2000 也提供了定制菜单的功能，即允许用户编写自己的菜单定义文件。如果定制的菜单（或菜单命令）需要挂接在 AutoCAD 本身提供的菜单中，那么只能在原有菜单文件的基础上进行必要的修改和添加。如果应用程序卸载了，其相应的菜单命令尽管无法执行，但依然存留在当前的菜单中，显得很不方便。

通过 AutoCAD 2000 提供的 COM 接口动态添加和删除菜单命令，而无需修改原有的菜单文件。当加载时，该例程为注册的程序模块添加相应的菜单命令项，而当例程卸载时，又将相应的菜单项全部删除。

本文例程使用 ObjectARX 2000 类库，采用 Visual C++ 6.0 开发，在 AutoCAD 2000 上调试通过。

一、建立工程

AutoCAD 2000 提供了引用其 COM 对象的类型库（type library），即 acad.tlb。该文件既可以在 AutoCAD 2000 的安装目录下找到，也可以在 ObjectARX 2000 安装目录的 \Inc\ 子目录中找到，并且两者是一样的。在 Visual C++ 6 中调用 COM 对象有两种途径：一是使用 MFC，从而可以利用 Visual C++ 6 提供的 ClassWizard（类向导）工具输入类型库；二是不使用 MFC，而直接采用 Win32 编程的方法引用类型库。本文以第二种方法为例，即不使用 MFC 的方式调用 AutoCAD 2000 提供的类型库。

按照 AutoCAD2000 ARX 开发的方法和步骤，在 Visual C++ 6 中使用 File | New 菜单命令创建一个空白工程（关于开发的方法和步骤，详见有关书籍）。选择工程类型为 Win32 Dynamic - Link Library（Win32 动态连接库），并且指定工程名称为 CCOMAddMenu。如图 1 所示。

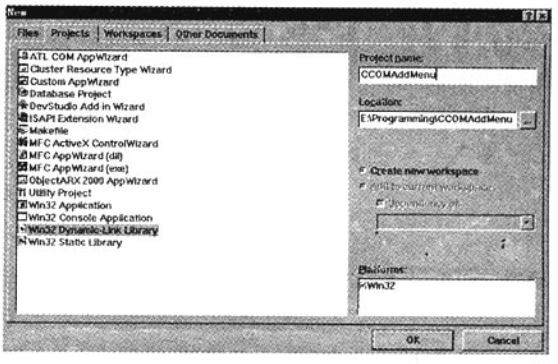

图 1 新建一个 Win32 动态库工程

接下来，使用 Project | Settings 菜单命令设置工程参数如下（其它参数保持默认值）：

(1) C/C++ 参数：选择 Category（分类）为 Code Generation，确保 Use run - time library（运行库）列表项为：Multithreaded DLL。

(2) Link 参数：选择 Category 为 General，将输出 DLL 文件的后缀更改为 .arx；选择引用的库文件为 acad.lib、rxapi.lib、acedapi.lib 和 acrx15.lib，库文件名之间以空格相隔。

最后，确保 Visual C++6 的头文件和库文件引用路径是正确的，否则，使用 Tools | Options | Directories 命令进行设置。

二、添加实现代码

向当前的工程项目添加三个文件，即头文件（CCOMAddMenu.h）、工程实现文件（CCOMAddMenu.cpp）以及模块定义文件（CCOMAddMenu.def）。

1. 打开 CCOMAddMenu.h 头文件，并添加如下代码：

```
/* CCOMAddMenu.h ——头文件 */
```

```
#include <aced.h>          // ARX 头文件
#include <rxregsvc.h>
#include <stdio.h>         // C/C++标准头文件
#include <tchar.h>         //
// 引入 AutoCAD 2000 类型库
#import "acad.tlb" no_implementation raw_interfaces_only named_guids
// 函数原型定义
void addOrDeleteMenuItem(void);          // 该函数动态添加/删除指定的菜单命令
void HelloWorld(void);                   // 该函数定义一个命令模块
// 头文件定义结束
```

注意：输入 COM 类型库是采用#import 预编译指令进行的，它可以附带一些参数(有关参数的含义，详见 Visual C++6 的联机帮助)。类型库文件名可以采用相对路径，也可以采用绝对路径。如果是相对路径，则必须保证其位于 Visual C++6 的库文件引用路径上。

2. 打开 CCOMAddMenu.cpp 文件，并添加如下代码：

```
/* CComAddMenu.cpp ——执行文件 */
#include "CComAddMenu.h"
// 该函数定义一个命令模块
void HelloWorld(void)
{
  acedAlert("Hello World!");
  return;
}
// 通过 AutoCAD 提供的 COM 接口，该函数动态添加 删除指定的菜单命令
void addOrDeleteMenuItem(void)
{
  // 静态变量，指示菜单命令是否已经添加
  static Adesk::Boolean bIsMenuLoaded = Adesk::kFalse;
  AutoCAD::IAcadPopupMenu *pPopUpMenu;
  VARIANT index;          // 临时变量，作索引用
// 通过 QueryInterface()方法获得 AutoCAD 2000 的实例对象
  AutoCAD::IAcadApplication *pAcad;
  HRESULT hr = NOERROR;
      CLSID clsid;
  LPUNKNOWN pUnk = NULL;
  LPDISPATCH pAcadDisp = NULL;
hr = ::CLSIDFromProgID(L"AutoCAD.Application", &clsid);
  if (SUCCEEDED(hr))
  {
   if(::GetActiveObject(clsid, NULL, &pUnk) == S_OK)
     {
if(pUnk->QueryInterface(IID_IDispatch, (LPVOID *) &pAcadDisp) != S_OK)
                return;
            pUnk->Release();
         }
       }
  if( SUCCEEDED(pAcadDisp->QueryInterface(AutoCAD::IID_IAcadApplication, (void **)&pAcad)) )
pAcad->put_Visible(true);      //使 AutoCAD 2000 可见
    else
  {
      acedPrompt("\nQueryInterface 出错!");
      return;
  }
  // 获得当前加载的菜单组
  AutoCAD::IAcadMenuGroups *pMenuGroups;
  pAcad->get_MenuGroups(&pMenuGroups);
pAcad->Release();   // AutoCAD 2000 实例对象已不需要
 // 获得第一个菜单组，通常为 ACAD，即 AutoCAD 2000 的主菜单组
  VariantInit(&index);
  V_VT(&index) = VT_I4;
  V_I4(&index) = 0;
  AutoCAD::IAcadMenuGroup *pMenuGroup;
  pMenuGroups->Item(index, &pMenuGroup);
  pMenuGroups->Release();
  // 获得 ACAD 包含的各下拉子菜单
  AutoCAD::IAcadPopupMenus *pPopUpMenus;
  pMenuGroup->get_Menus(&pPopUpMenus);
  pMenuGroup->Release();
// 获取指定的用于添加菜单项的菜单，此处为 ACAD 的 Window 菜单
  long numbers;
  pPopUpMenus->get_Count(&numbers);
    for( int num = 0; num < numbers; num++ )
  {
    VariantInit(&index);
    V_VT(&index) = VT_I4;
    V_I4(&index) = num;
     pPopUpMenus->Item(index, &pPopUpMenu);
    if(pPopUpMenu == NULL)
    {
      acedPrompt("\n 无法获取指定菜单!");
      return;
    }
BSTR menuName;
      TCHAR tString[256];
      pPopUpMenu->get_Name(&menuName);
     WideCharToMultiByte(CP_ACP, 0, menuName, -1, tString, 256, NULL, NULL);
if( !stricmp(tString, _T("&Window")))// 如果是 Window 菜单
          break;
    }
  pPopUpMenus->Release();
    if( num >= numbers || pPopUpMenu == NULL)
// 如果没有找到合适的菜单
    return;
AutoCAD::IAcadPopupMenuItem *pPopUpMenuItem;
  WCHAR wstrMenuItemName[256], wstrMenuItemMacro[256];
if ( bIsMenuLoaded == Adesk::kFalse)          //如果菜单未添加，则添加
  {
    // 添加菜单项及其宏命令，宏命令中末尾添加一个空格
```

```
        VariantInit(& index);
        V_VT(& index) = VT_I4;
        V_I4(& index) = 0;        // 添加的菜单项为第一项
        MultiByteToWideChar(CP_ACP, 0, "& Hello World",
-1, wstrMenuItemName, 256);
MultiByteToWideChar(CP_ACP, 0, "LLH_COMMANDS_
HELLO", -1, wstrMenuItemMacro, 256);
        pPopUpMenu->AddMenuItem(index, wstrMenuItem-
Name, wstrMenuItemMacro, & pPopUpMenuItem);
MultiByteToWideChar(CP_ACP, 0, "单击该菜单命令执行\"
Hello World!\"例程.", -1, wstrMenuItemName, 256);
pPopUpMenuItem->put_HelpString(wstrMenuItemName);
// 附加帮助提示
            pPopUpMenuItem->Release();
V_I4(& index) = 1;        // 添加的菜单项为第二项
pPopUpMenu->AddSeparator(index, & pPopUpMenuItem);
// 添加分隔符
            pPopUpMenuItem->Release();
        acedPrompt("\n 菜单命令添加在 Window 菜单中.");
            blsMenuLoaded = Adesk::kTrue;
          }
        else    // 如果菜单已经添加,则删除之
        {
// 删除菜单项,先删除第二项,后删除第一项.注意删除的顺序.
        VariantInit(& index);
        V_VT(& index) = VT_I4;
        V_I4(& index) = 1;        // 删除第二个菜单项
        pPopUpMenu->Item(index, & pPopUpMenuItem);
        pPopUpMenuItem->Delete();
        V_I4(& index) = 0;        // 删除第一个菜单项
        pPopUpMenu->Item(index, & pPopUpMenuItem);
        pPopUpMenuItem->Delete();
            acedPrompt("\n 菜单命令已经删除!");
            blsMenuLoaded = Adesk::kFalse;
          }
        pPopUpMenu->Release();
    return;
}    // addOrDeleteMenuItem()函数定义结束
// 加载应用程序的初始化函数
void initApp()
{
addOrDeleteMenuItem(), //第一次执行,添加指定菜单命令
    /* 向 AutoCAD 注册一个定制的命令 */
          acedRegCmds->addCommand(        "ASDK_LLH_COMMANDS",  "ADSK_LLH_COMMANDS_HELLO",
"LLH_COMMANDS_HELLO",  ACRX_CMD_MODAL,  HelloWorld);
    return;
}
// 卸载应用程序的清理函数
void unloadApp()
{
 addOrDeleteMenuItem(); //第二次执行,删除指定菜单命令
acedRegCmds->removeGroup("ASDK_LLH_COMMANDS");
    return;
}
// ARX 程序入口函数
extern "C" AcRx::AppRetCode acrxEntryPoint(AcRx::
AppMsgCode msg, void* pkt)
{
   switch (msg) {
   case AcRx::kInitAppMsg:
      acrxDynamicLinker->unlockApplication(pkt); // 允许
应用程序中途能够卸载
      acrxRegisterAppMDIAware(pkt);
      initApp();        //初始化应用程序
      break;
   case AcRx::kUnloadAppMsg:
      unloadApp();    //应用程序退出前进行必要的清理
      break;
   case AcRx::kLoadDwgMsg:
      break;
   }
   return AcRx::kRetOK;
}
```

函数 addOrDeleteMenuItem()向 AutoCAD 2000 的 Windows 菜单添加（或删除）定制的菜单命令（"Hello World"）和分隔符。函数的主要过程是：首先，通过一系列调用 QueryInterface()函数，获得 AutoCAD 2000 的运行实例；其次，获得 ACAD 菜单组的 Windows 菜单（此处假定当前加载的菜单组是 ACAD，即 AutoCAD 2000 的标准菜单）；最后，根据静态变量 blsMenuLoaded 的值决定，是添加菜单项还是删除菜单项。

事实上，通过 COM 接口访问 AutoCAD 2000，关键是通过一系列 QueryInterface()函数调用获得其实例对象。在获得实例对象之后，余下的工作就是应用程序本身需要实现的功能逻辑了。

不过需要注意的是，在进行字符串的有关操作时（如菜单名的比较、添加指定名称的菜单项等），需要事先进行字符代码的转换，否则，将出现错误结果。例程中字符代码的转换，是使用 WideCharToMultiByte()函数或 MultiByteToWideChar()函数来进行的。

3. 最后，向模块定义文件 CCOMAddMenu.def 添加定义代码如下：

```
;/******* CCOMAddMenu.def ******/
LIBRARY     CCOMAddMenu
DESCRIPTION 'ARX program for AutoCAD 2000'
EXPORTS
   acrxEntryPoint      PRIVATE
   _SetacrxPtp         PRIVATE
   acrxGetApiVersion   PRIVATE
```

注意：英文分号（;）开头的行为模块定义文件的注释行。至此，源代码编辑完毕。

三、编译和运行应用程序

编译整个工程，得到 CCOMAddMenu.arx 文件。如果按照上述步骤进行的话，那么整个编译过程不会出现任何警告或错

误信息。

在 AutoCAD 2000 下加载 CCOMAddMenu.arx，可以看到，在加载的时候，定制的菜单项（即“Hello World”命令和一个分隔符）已经添加完成，如图 2 所示。当光标移到 Hello

图 2 加载程序时添加指定的菜单命令

World 菜单命令时，状态栏显示定制的帮助提示。如果单击 Window | Hello World 菜单命令，则弹出一个“Hello World”信息框。

卸载应用程序时，定制的菜单命令则被删除，如图 3 所示。实际上，通过 COM 接口添加的菜单项不会修改 AutoCAD

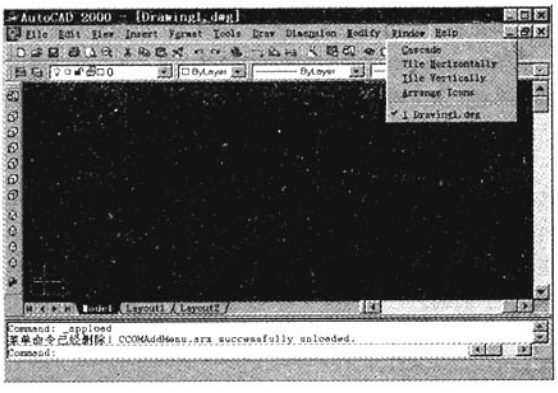

图 3 卸载程序时删除指定的菜单命令

2000 的菜单定义文件，它们只在当前的运行期间有效。当下一次启动 AutoCAD 2000 时，这些菜单项不会出现，除非应用程序再次添加。

四、结束语

本文介绍的例程通过 AutoCAD 2000 提供 COM 接口，在其原有菜单命令的基础上提供定制的菜单命令，并且避免了菜单定义文件的修改。此外，例程可以在程序加载时动态添加必要的菜单命令，而在卸载时又可以删除已不需要的菜单命令，这样既方便又灵活。本文为 AutoCAD 主程序界面与应用程序功能紧密地结合在一起，提供了新的方法和思路。当然，本文目的仅在于抛砖引玉，欢迎广大 AutoCAD 二次开发人员提出更多、更好的应用方法。

参考资料

1. 刘良华、朱东海著．AutoCAD2000 ARX 开发技术．北京：清华大学出版社，2000

2. 杨秀章译．COM 技术内幕——微软组件对象模型．北京：清华大学出版社，1999

（收稿日期：2001 年 1 月 12 日）

注册方式软件加密

海迅达注册控件及注册管理程序

以前软件的加密大多采用软件狗的方式，但该方式存在着一些明显的特点：

- 成本过高，对于千元以下产品不合适；
- 对软硬件环境有较高限制，容易产生硬件冲突；
- 可采用共享器等方式解密；
- 对于盗版狗难以防范；
- 软件免损坏后更换麻烦。

鉴于以上缺点，软件加密的另一种方式—注册方式被越来越广泛地采用。例如，大量的共享软件和 MS Office 等都采用了注册方式加密。

注册方式加密具有保密性强、适用面广，价格低廉、方便管理等众多优点，是软件加密的发展方向。

但注册方式加密也不是轻而易举的事情，开发者需周全考虑用户使用方便、防解密、防盗版注册管理方便、能否适用于各种操作系统和硬件环境等复杂问题。

最近，由北京海迅达科技有限公司推出的“海迅达注册控件及注册管理程序”为广大开发人员提供了一种便利的注册方式加密手段。

“海迅达注册控件及注册管理程序”开发包括一个标准的 ActiveX 控件（用于产品中实现注册加密功能）和一套完整的用户注册管理程序，并提供 VC + + 和 VB 使用示例。

该产品具有以下特点：

- 价格低廉，批量越大，平均每件的价格越低；
- 提取用户微机的软硬件信息加密，无法仿冒；
- 采用 ActiveX 控件方式加密，难以解密；
- 适用于 VC + + 、VB、Delphi 等开发环境；
- 适用于 Windows95 - 2000 的全系统操作系统；
- 对用户的硬件环境没有硬性要求，不存在硬件冲突；
- 用户注册管理程序保留有全面的用户信息，用户管理方便；
- 用户注册管理程序提供产品序列号打印功能；
- 由公司决定允许用户注册次数；
- 开放用户注册数据库，开发商可自行处理及自行开发注册查询、打印软件。

在ObjectARX程序中动态添加和删除AutoCAD菜单命令

作者： 刘良华， 袁英战
作者单位：
刊名： 电脑编程技巧与维护
英文刊名： COMPUTER PROGRAMMING SKILLS & MAINTENANCE
年，卷(期)： 2001(6)

参考文献(2条)

1. 刘良华;朱东海 AutoCAD2000 ARX开发技术 2000

2. 杨秀章 COM技术内幕--微软组件对象模型 1999

本文读者也读过(9条)