

博文

[转载]MFC使用ADO读写Access数据库

已有 1697 次阅读 2014-4-24 18:36 | 个人分类:程序设计 | 系统分类:生活其它 | 关键词:C++ 数据库 | 文章来源:转载

MFC使用ADO读写Access数据库

ADO(ActiveX Data Object)是Microsoft数据库应用程序开发的新接口，是建立在OLE DB之上的高层数据库访问技术，即使你对OLE DB，COM不了解也能轻松对付ADO，因为它非常简单易用，甚至比你以往所接触的ODBC API、DAO、RDO都要容易使用，并不失灵活性。本文详细地介绍在Visual C++开发环境下如何使用ADO来进行数据库应用程序开发，并给出示例代码。为了使读者朋友都能测试本例提供的代码，我们采用Access数据库，您可以直接在我们提供的示例代码中找到这个test.mdb

一、实现方法

万事开头难，任何一种新技术对于初学者来说最重要的还是"入门"，掌握其要点。让我们来看看ADO数据库开发的基本流程吧！它的基本步骤如下：

- （1）初始化COM库，引入ADO库定义文件
- （2）用Connection对象连接数据库
- （3）利用建立好的连接，通过Connection、Command对象执行SQL命令，或利用Recordset对象取得结果记录集进行查询、处理。
- （4）使用完毕后关闭连接释放对象。

下面我们将详细介绍上述步骤并给出相关代码。

1、COM库的初始化

我们可以使用AfxOleInit()来初始化COM库，这项工作通常在CWinApp::InitInstance()的重载函数中完成，请看如下代码：

```
BOOL CADOTest1App::InitInstance()
{
AfxOleInit();
.....
}
```

2、用#import指令引入ADO类型库

为了引入ADO类型库，需要在项目的stdafx.h文件中加入如下语句：

```
#import "c:\program files\common files\system\ado\msado15.dll" no_namespace rename("EOF","adoEOF")
```

这一语句有何作用呢？其最终作用同我们已经十分熟悉的#include类似，编译的时候系统会为我们生成msado15.tlh,ado15.tli两个C++头文件来定义ADO库。

需要读者朋友注意的是：如果你的开发环境中msado15.dll不在这个目录下，请按实际情况修改；在编译的时候可能会出现如下警告，对此微软在MSDN中作了说明，并建议我们不要理会这个警告：msado15.tlh(405) : warning C4146: unary minus operatorapplied to unsigned type, result still unsigned。

3、创建Connection对象并连接数据库

为了首先我们需要添加一个指向Connection对象的智能指针_ConnectionPtr m_pConnection，下面的代码演示了如何创建Connection对象实例及如何连接数据库并进行异常捕捉：

```
BOOL CADOTest1Dlg::OnInitDialog()
{
    CDialog::OnInitDialog();
    HRESULT hr;
    try
    {
        hr = m_pConnection.CreateInstance("ADODB.Connection"); //创建Connection对象
        if(SUCCEEDED(hr))
        {
            hr = m_pConnection->Open("Provider=Microsoft.Jet.OLEDB.4.0;
            Data Source=test.mdb","", "",adModeUnknown); //连接数据库
            //上面一句中连接字符串中的Provider是针对ACCESS2000环境的，对于ACCESS97，
            //需要改为： Provider=Microsoft.Jet.OLEDB.3.51;
        }
    }
    catch(_com_error e)///捕捉异常
    {
        CString errorMessage;
        errorMessage.Format("连接数据库失败!\n\n错误信息:%s",e.ErrorMessage());
        AfxMessageBox(errorMessage);///显示错误信息
    }
}
```

在这段代码中我们是通过Connection对象的Open方法来进行连接数据库的，下面是该方法的原型：

```
HRESULT Connection15::Open ( _bstr_t ConnectionString, _bstr_t UserID, _bstr_t Password, long Options );
```

上述函数中参数ConnectionString为连接字符串；参数UserID是用户名；参数Password是登陆密码；参数Options是连接选项，用于指定Connection对象对数据的更新许可权，一般情况下Options可以是如下几个常量：

adModeUnknown:缺省。当前的许可权未设置

adModeRead:只读

adModeWrite:只写

adModeReadWrite:可以读写

adModeShareDenyRead:阻止其它Connection对象以读权限打开连接

adModeShareDenyWrite:阻止其它Connection对象以写权限打开连接

adModeShareExclusive:阻止其它Connection对象以读写权限打开连接

adModeShareDenyNone:阻止其它Connection对象以任何权限打开连接

我们给出一些常用的连接方式供大家参考：

（1）通过JET数据库引擎对ACCESS2000数据库的连接：

```
m_pConnection->Open("Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:\test.mdb","", "",adModeUnknown);
```

（2）通过DSN数据源对任何支持ODBC的数据库进行连接：

```
m_pConnection->Open("Data Source=adotest;UID=sa;PWD=;", "", "",adModeUnknown);
```

（3）不通过DSN对SQL SERVER数据库进行连接：

```
m_pConnection->Open("driver={SQL Server};Server=127.0.0.1;DATABASE=vckbase;
UID=sa;PWD=139","", "",adModeUnknown);
```

其中Server是SQL服务器的名称，DATABASE是库的名称。

Connection对象除Open（）方法外还有许多方法，我们先介绍Connection对象中两个有用的属性ConnectionTimeOut与State。ConnectionTimeOut用来设置连接的超时时间，需要在Open之前调用，例如：

```
m_pConnection->ConnectionTimeout = 5; //设置超时时间为5秒
```

```
m_pConnection->Open("Data Source=adotest","","",adModeUnknown);
```

State属性指明当前**Connection**对象的状态，0表示关闭，1表示已经打开，我们可以通过读取这个属性来进行相应的处理，例如：

```
if(m_pConnection->State)
m_pConnection->Close(); //如果已经打开了连接则关闭它
```

4、执行SQL命令并取得结果记录集

为了取得结果记录集，我们定义一个指向**Recordset**对象的指针：**_RecordsetPtr m_pRecordset**；并为其创建**Recordset**对象的实例：**m_pRecordset.CreateInstance("ADODB.Recordset")**，SQL命令的执行可以采用多种形式，下面我们一一进行阐述。

（1）利用**Connection**对象的**Execute**方法执行SQL命令

Execute（）方法的原型如下所示：

```
_RecordsetPtr Connection15::Execute ( _bstr_t CommandText, VARIANT * RecordsAffected, long Options )
```

其中**CommandText**是命令字符串，通常是SQL命令。参数**RecordsAffected**是操作完成后所影响的行数, 参数**Options**表示**CommandText**中内容的类型，**Options**可以取如下值之一：**adCmdText**表明**CommandText**是文本命令；**adCmdTable**表明**CommandText**是一个表名；**adCmdProc**表明**CommandText**是一个存储过程；**adCmdUnknown**表明**CommandText**内容未知。**Execute**（）函数执行完后返回一个指向记录集的指针，下面我们给出具体代码并作说明：

```
_variant_t RecordsAffected;
///执行SQL命令：CREATE TABLE创建表格users,users包含四个字段:整形ID,字符串username,整形old,日期型birthday
m_pConnection->Execute("CREATE TABLE users(ID INTEGER,username
TEXT,old INTEGER,birthday DATETIME)",&RecordsAffected,adCmdText);

///往表格里面添加记录
m_pConnection->Execute("INSERT INTO users(ID,username,old,birthday)
VALUES (1, 'Washington',25,'1970/1/1')",&RecordsAffected,adCmdText);

///将所有记录old字段的值加一
m_pConnection->Execute("UPDATE users SET old = old+1",&RecordsAffected,adCmdText);

///执行SQL统计命令得到包含记录条数的记录集
m_pRecordset = m_pConnection->Execute("SELECT COUNT(*) FROM
users",&RecordsAffected,adCmdText);

_variant_t vIndex = (long)0;
_variant_t vCount = m_pRecordset->GetCollect(vIndex);///取得第一个字段的值放入vCount变量
m_pRecordset->Close();///关闭记录集

CString message;
message.Format("共有%d条记录",vCount.IVal);
AfxMessageBox(message);///显示当前记录条数
```

（2）利用**Command**对象来执行SQL命令

```
_CommandPtr m_pCommand;
m_pCommand.CreateInstance("ADODB.Command");

_variant_t vNULL;
vNULL.vt = VT_ERROR;
vNULL.scode = DISP_E_PARAMNOTFOUND;///定义为无参数
```

```
m_pCommand->ActiveConnection = m_pConnection;///非常关键的一句，将建立的连接赋值给它
m_pCommand->CommandText = "SELECT * FROM users";///命令字符串
m_pRecordset = m_pCommand->Execute(&vNULL,&vNULL,adCmdText);
//执行命令取得记录集
```

在这段代码中我们只是用**Command**对象来执行了**SELECT**查询语句，**Command**对象在进行存储过程的调用中能真正体现它的作用。下次我们将详细介绍。

（3）直接用**Recordset**对象进行查询取得记录集，例如：

```
m_pRecordset->Open("SELECT * FROM users",_variant_t((IDispatch *)m_pConnection,true),
adOpenStatic,adLockOptimistic,adCmdText);
```

Open（）方法的原型如下：

```
HRESULT Recordset15::Open ( const _variant_t & Source, const _variant_t & ActiveConnection, enum CursorTypeEnum
CursorType, enum LockTypeEnum LockType, long Options )
```

上述函数中参数**Source**是数据查询字符串；参数**ActiveConnection**是已经建立好的连接（我们需要用**Connection**对象指针来构造一个**_variant_t**对象）；参数**CursorType**光标类型，它可以是以下值之一；请看这个枚举结构：

```
enum CursorTypeEnum
{
adOpenUnspecified = -1,///不作特别指定
adOpenForwardOnly = 0,///前滚静态光标。这种光标只能向前浏览记录集，比如用MoveNext向前滚动,这种方式可以提高浏览速度。但诸如Bookmark,RecordCount,AbsolutePosition,AbsolutePage都不能使用
adOpenKeyset = 1,///采用这种光标的记录集看不到其它用户的新增、删除操作，但对于更新原有记录的操作对你可见的。
adOpenDynamic = 2,///动态光标。所有数据库的操作都会立即在各用户记录集上反应出来。
adOpenStatic = 3,///静态光标。它为你的记录集产生一个静态备份，但其它用户的新增、删除、更新操作对你的记录集来说是不可见的。
};
```

参数**LockType**表示数据库的锁定类型，它可以是以下值之一，请看如下枚举结构：

```
enum LockTypeEnum
{
adLockUnspecified = -1,///未指定
adLockReadOnly = 1,///只读记录集
adLockPessimistic = 2,悲观锁定方式。数据在更新时锁定其它所有动作，这是最安全的锁定机制
adLockOptimistic = 3,乐观锁定方式。只有在你调用Update方法时才锁定记录。在此之前仍然可以做数据的更新、插入、删除等动作
adLockBatchOptimistic = 4，乐观分批更新。编辑时记录不会锁定，更改、插入及删除是在批处理模式下完成。
};
```

参数**Options**的含义请参考本文中对**Connection**对象的**Execute（）**方法的介绍。

5、记录集的遍历、更新

根据我们刚才通过执行**SQL**命令建立好的**users**表，它包含四个字段:**ID,username,old,birthday**

以下的代码实现：打开记录集，遍历所有记录，删除第一条记录，添加三条记录，移动光标到第二条记录，更改其年龄数据，保存到数据库。

```
_variant_t vUsername, vBirthday, vID, vOld;
_RecordsetPtr m_pRecordset;
m_pRecordset.CreateInstance("ADODB.Recordset");
m_pRecordset->Open("SELECT * FROM users",_variant_t((IDispatch*)m_pConnection,true),
adOpenStatic,adLockOptimistic,adCmdText);
while(!m_pRecordset->adoEOF)
```

```
///这里为什么是adoEOF而不是EOF呢?还记得rename("EOF","adoEOF")这一句吗?
{
vID = m_pRecordset->GetCollect(_variant_t((long)0));///取得第1列的值,从0开始计数,你也可以直接给出列的名称;
vUsername = m_pRecordset->GetCollect("username");///取得username字段的值
vOld = m_pRecordset->GetCollect("old");
vBirthday = m_pRecordset->GetCollect("birthday");
///在DEBUG方式下的OUTPUT窗口输出记录集中的记录
if(vID.vt != VT_NULL && vUsername.vt != VT_NULL && vOld.vt != VT_NULL &&
vBirthday.vt != VT_NULL)
TRACE("id:%d,姓名:%s,年龄:%d,生日:%s\r\n",vID.IVal,(LPCTSTR)(_bstr_t)vUsername,vOld.IVal,(LPCTSTR)
(_bstr_t)vBirthday);
m_pRecordset->MoveNext();///移到下一条记录
}
m_pRecordset->MoveFirst();///移到首条记录
m_pRecordset->Delete(adAffectCurrent);///删除当前记录


///添加三条新记录并赋值
for(int i=0;i<3;i++)
{
m_pRecordset->AddNew();///添加新记录
m_pRecordset->PutCollect("ID",_variant_t((long)(i+10)));
m_pRecordset->PutCollect("username",_variant_t("叶利钦"));
m_pRecordset->PutCollect("old",_variant_t((long)71));
m_pRecordset->PutCollect("birthday",_variant_t("1930-3-15"));
}


m_pRecordset->Move(1,_variant_t((long)adBookmarkFirst));///从第一条记录往下移动一条记录,即移动到第二条记录处
m_pRecordset->PutCollect(_variant_t("old"),_variant_t((long)45));///修改其年龄
m_pRecordset->Update();///保存到库中
```

6、关闭记录集与连接

记录集或连接都可以用**Close**（）方法来关闭：

```
m_pRecordset->Close();///关闭记录集
m_pConnection->Close();///关闭连接
```

至此，我想读者朋友已经熟悉了**ADO**操作数据库的大致流程，也许您已经胸有成竹，也许您还有点糊涂，不要紧！建议你尝试写几个例子，这样会更好地熟悉**ADO**，最后我给大家写了一个小例子，例子实现的功能是读出所有记录并放到列表控件中，同时可以添加、删除、修改记录。

二、编程步骤

- 1、启动**Visual C++6.0**，生成一个基于对话框的应用程序，将该程序命名为**ADOTest1**；
- 2、在对话框界面上放置显示记录列表控件和添加、删除记录用的编辑、按钮控件，具体设置参加代码中对话框资源部分；
- 3、使用**Class Wizard**为添加、修改数据库记录的按钮添加消息响应函数；
- 4、添加成程序代码，编译运行程序。

三、程序代码

```
////////////////////////////////////// ADOTest1Dlg.h : header file

#ifndef __AFX_ADOTEST1DLG_H__29B385C0_02C0_4588_A8B4_D0EFBB4F578D__INCLUDED_
#define __AFX_ADOTEST1DLG_H__29B385C0_02C0_4588_A8B4_D0EFBB4F578D__INCLUDED_
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

class CADOTest1Dlg : public CDialog
{
// Construction
public:
    BOOL m_bAutoSave;
```



```
void SaveData();
void LoadData();
_variant_t vUserID, vUsername, vOld, vBirthday;
BOOL m_bSuccess;
int m_nCurrentSel;
_RecordsetPtr m_pRecordset;

CADOTest1Dlg(CWnd* pParent = NULL); // standard constructor
// Dialog Data
//{{AFX_DATA(CADOTest1Dlg)
enum { IDD = IDD_ADOTEST1_DIALOG };
CButton m_cDelItem;
CButton m_cAddItem;
CListCtrl m_userlist;
UINT m_nUserID;
UINT m_nOld;
CString m_sUsername;
COleDateTime m_tBirthday;
//}}AFX_DATA
// ClassWizard generated virtual function overrides
//{{AFX_VIRTUAL(CADOTest1Dlg)
protected:
virtual void DoDataExchange(CDataExchange* pDX); // DDX/DDV support
//}}AFX_VIRTUAL
// Implementation
protected:
HICON m_hIcon;
// Generated message map functions
//{{AFX_MSG(CADOTest1Dlg)
virtual BOOL OnInitDialog();
afx_msg void OnSysCommand(UINT nID, LPARAM lParam);
afx_msg void OnPaint();
afx_msg HCURSOR OnQueryDragIcon();
virtual void OnOK();
afx_msg void OnAdditem();
afx_msg void OnDelitem();
afx_msg void OnItemchangedUserlist(NMHDR* pNMHDR, LRESULT* pResult);
//}}AFX_MSG
DECLARE_MESSAGE_MAP()
};
#endif
```

////////////////////////////////////// ADOTest1Dlg.cpp : implementation file

```
#include "stdafx.h"
#include "ADOTest1.h"
#include "ADOTest1Dlg.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
extern CADOTest1App theApp;
class CAboutDlg : public CDialog
{
public:
CAboutDlg();
// Dialog Data
//{{AFX_DATA(CAboutDlg)
enum { IDD = IDD_ABOUTBOX };
//}}AFX_DATA
// ClassWizard generated virtual function overrides
//{{AFX_VIRTUAL(CAboutDlg)
protected:
virtual void DoDataExchange(CDataExchange* pDX); // DDX/DDV support
```

```
//}}AFX_VIRTUAL
// Implementation
protected:
//{{AFX_MSG(CAboutDlg)
//}}AFX_MSG
DECLARE_MESSAGE_MAP()

};

CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
//{{AFX_DATA_INIT(CAboutDlg)
//}}AFX_DATA_INIT
}

void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
CDialog::DoDataExchange(pDX);
//{{AFX_DATA_MAP(CAboutDlg)
//}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
//{{AFX_MSG_MAP(CAboutDlg)
// No message handlers
//}}AFX_MSG_MAP
END_MESSAGE_MAP()

CADOTest1Dlg::CADOTest1Dlg(CWnd* pParent /*=NULL*/)
: CDialog(CADOTest1Dlg::IDD, pParent)
{
//{{AFX_DATA_INIT(CADOTest1Dlg)
m_nUserID = 0;
m_nOld = 0;
m_sUsername = _T("");
m_tBirthday = COleDateTime::GetCurrentTime();
//}}AFX_DATA_INIT
// Note that LoadIcon does not require a subsequent DestroyIcon in Win32
m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
m_nCurrentSel = -1;
m_bSuccess = FALSE;
m_bAutoSave = TRUE;
}

void CADOTest1Dlg::DoDataExchange(CDataExchange* pDX)
{
CDialog::DoDataExchange(pDX);
//{{AFX_DATA_MAP(CADOTest1Dlg)
DDX_Control(pDX, IDC_DELITEM, m_cDelItem);
DDX_Control(pDX, IDC_ADDITEM, m_cAddItem);
DDX_Control(pDX, IDC_USERLIST, m_userlist);
DDX_Text(pDX, IDC_USERID, m_nUserID);
DDX_Text(pDX, IDC_OLD, m_nOld);
DDX_Text(pDX, IDC_USERNAME, m_sUsername);
DDX_DateTimeCtrl(pDX, IDC_DATETIMEPICKER1, m_tBirthday);
//}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CADOTest1Dlg, CDialog)
//{{AFX_MSG_MAP(CADOTest1Dlg)
ON_WM_SYSCOMMAND()
ON_WM_PAINT()
ON_WM_QUERYDRAGICON()
ON_BN_CLICKED(IDC_ADDITEM, OnAdditem)
ON_BN_CLICKED(IDC_DELITEM, OnDelitem)
ON_NOTIFY(LVN_ITEMCHANGED, IDC_USERLIST, OnItemchangedUserlist)
//}}AFX_MSG_MAP
END_MESSAGE_MAP()
```

```

    BOOL CADOTest1Dlg::OnInitDialog()
    {
        CDialog::OnInitDialog();

        m_cDelItem.EnableWindow(FALSE);

        ::SendMessage(m_userlist.m_hWnd, LVM_SETEXTENDEDLISTVIEWSTYLE,
        LVS_EX_FULLROWSELECT, LVS_EX_FULLROWSELECT);

        //////////为列表控件添加列//////////
        m_userlist.InsertColumn(0,"用户ID",LVCFMT_LEFT,60);
        m_userlist.InsertColumn(1,"用户名",LVCFMT_LEFT,100);
        m_userlist.InsertColumn(2,"年龄",LVCFMT_LEFT,60);
        m_userlist.InsertColumn(3,"生日",LVCFMT_LEFT,100);

        //////////读取数据库中的信息添加到列表控件//////////
        int nItem;
        _variant_t vUsername,vBirthday,vID,vOld;
        try
        {
            m_pRecordset.CreateInstance("ADODB.Recordset");
            m_pRecordset->Open("SELECT*FROM users",
            _variant_t((IDispatch*)theApp.m_pConnection,true),
            adOpenStatic,adLockOptimistic,adCmdText);
            m_bSuccess = TRUE;
            while(!m_pRecordset->adoEOF)
            {
                vID = m_pRecordset->GetCollect("ID");
                vUsername = m_pRecordset->GetCollect("username");
                vOld = m_pRecordset->GetCollect("old");
                vBirthday = m_pRecordset->GetCollect("birthday");
                nItem=m_userlist.InsertItem(0xffff,(_bstr_t)vID);
                m_userlist.SetItem(nItem,1,1,(_bstr_t)vUsername,NULL,0,0,0);
                m_userlist.SetItem(nItem,2,1,(_bstr_t)vOld,NULL,0,0,0);
                m_userlist.SetItem(nItem,3,1,(_bstr_t)vBirthday,NULL,0,0,0);
                m_pRecordset->MoveNext();
            }
        }
        catch(_com_error e)///捕捉异常
        {
            AfxMessageBox("读取数据库失败!");///显示错误信息
        }

        ASSERT((IDM_ABOUTBOX & 0xFFFF0) == IDM_ABOUTBOX);
        ASSERT(IDM_ABOUTBOX < 0xF000);
        CMenu* pSysMenu = GetSystemMenu(FALSE);
        if (pSysMenu != NULL)
        {
            CString strAboutMenu;
            strAboutMenu.LoadString(IDS_ABOUTBOX);
            if (!strAboutMenu.IsEmpty())
            {
                pSysMenu->AppendMenu(MF_SEPARATOR);
                pSysMenu->AppendMenu(MF_STRING, IDM_ABOUTBOX, strAboutMenu);
            }
        }
        SetIcon(m_hIcon, TRUE); // Set big icon
        SetIcon(m_hIcon, FALSE); // Set small icon
        return TRUE; // return TRUE unless you set the focus to a control
    }

    void CADOTest1Dlg::OnSysCommand(UINT nID, LPARAM lParam)
    {
        if ((nID & 0xFFFF0) == IDM_ABOUTBOX)
        {

```



```
CAboutDlg dlgAbout;
dlgAbout.DoModal();
}
else
{
CDialog::OnSysCommand(nID, IParam);
}
}

void CADOTest1Dlg::OnPaint()
{
if (IsIconic())
{
CPaintDC dc(this); // device context for painting
SendMessage(WM_ICONERASEBKGND, (WPARAM) dc.GetSafeHdc(), 0);
// Center icon in client rectangle
int cxIcon = GetSystemMetrics(SM_CXICON);
int cyIcon = GetSystemMetrics(SM_CYICON);
CRect rect;
GetClientRect(&rect);
int x = (rect.Width() - cxIcon + 1) / 2;
int y = (rect.Height() - cyIcon + 1) / 2;
// Draw the icon
dc.DrawIcon(x, y, m_hIcon);
}
else
{
CDialog::OnPaint();
}
}

HCURSOR CADOTest1Dlg::OnQueryDragIcon()
{
return (HCURSOR) m_hIcon;
}

void CADOTest1Dlg::OnOK()
{
if(m_bSuccess)
{
m_pRecordset->Update();
m_pRecordset->Close();
}
CDialog::OnOK();
}

void CADOTest1Dlg::OnAdditem()
{
if(UpdateData())
if(m_sUsername.GetLength()>0)
{
m_pRecordset->AddNew();
m_nCurrentSel = m_userlist.InsertItem(0xffff,"");
SaveData();///保存数据
m_userlist.SetItemState(m_nCurrentSel,LVIS_SELECTED|LVIS_FOCUSED,LVIS_SELECTED|LVIS_FOCUSED);
m_userlist.SetHotItem(m_nCurrentSel);
m_userlist.SetFocus();
}
else
AfxMessageBox("请输入用户名");
}

void CADOTest1Dlg::OnDelitem()
{
m_bAutoSave = FALSE;
if(m_nCurrentSel >= 0)
{
```

```
m_userlist.DeleteItem(m_nCurrentSel);
int count = m_userlist.GetItemCount();
if(count <= m_nCurrentSel)
m_nCurrentSel = count-1;
m_pRecordset->Delete(adAffectCurrent);
m_pRecordset->MoveNext();
LoadData();
m_userlist.SetItemState(m_nCurrentSel,LVIS_SELECTED|LVIS_FOCUSED,LVIS_SELECTED|LVIS_FOCUSED);
m_userlist.SetFocus();
}
m_bAutoSave = TRUE;
}

////////在选择列表框的时候调用//////////

void CADOTest1Dlg::OnItemchangedUserlist(NMHDR* pNMHDR, LRESULT* pResult)
{
NM_LISTVIEW* pNMListView = (NM_LISTVIEW*)pNMHDR;
if(pNMListView->uNewState&LVIS_SELECTED)
{
UpdateData();
SaveData();///保存旧数据
m_nCurrentSel = pNMListView->iItem;
LoadData();///加载新数据
m_cDelItem.EnableWindow();
}
*pResult = 0;
}

/////将记录集中的数据加载到编辑框/////
void CADOTest1Dlg::LoadData()
{
m_pRecordset->Move(m_nCurrentSel,_variant_t((long)adBookmarkFirst));
vUserID = m_pRecordset->GetCollect("ID");
vUsername = m_pRecordset->GetCollect("username");
vOld = m_pRecordset->GetCollect("old");
vBirthday = m_pRecordset->GetCollect("birthday");
m_nUserID = vUserID.IVal;
m_sUsername = (LPCTSTR)(_bstr_t)vUsername;
m_nOld = vOld.IVal;
m_tBirthday = vBirthday;
UpdateData(FALSE);
}

/////将编辑框的数据保存到记录集与列表框
void CADOTest1Dlg::SaveData()
{
if(!m_pRecordset->adoEOF && m_nCurrentSel >= 0 && m_bAutoSave)
{
vUserID = (long)m_nUserID;
vUsername = m_sUsername;
vOld = (long)m_nOld;
vBirthday = m_tBirthday;
m_pRecordset->PutCollect("ID",vUserID);
m_pRecordset->PutCollect("username",vUsername);
m_pRecordset->PutCollect("old",vOld);
m_pRecordset->PutCollect("birthday",vBirthday);
m_userlist.SetItem(m_nCurrentSel,0,LVIF_TEXT,(_bstr_t)vUserID,NULL,0,0,0);
m_userlist.SetItem(m_nCurrentSel,1,LVIF_TEXT,(_bstr_t)vUsername,NULL,0,0,0);
m_userlist.SetItem(m_nCurrentSel,2,LVIF_TEXT,(_bstr_t)vOld,NULL,0,0,0);
m_userlist.SetItem(m_nCurrentSel,3,LVIF_TEXT,(_bstr_t)vBirthday,NULL,0,0,0);
}
}

////////////////////////////////////

BOOL CADOTest1App::InitInstance()
{
AfxEnableControlContainer();
AfxOleInit();///初始化COM库
```

```
HRESULT hr; //////////连接数据库//////////
try
{
hr = m_pConnection.CreateInstance("ADODB.Connection");//创建Connection对象
if(SUCCEEDED(hr))
{
hr = m_pConnection->Open("Provider=Microsoft.Jet.OLEDB.4.0;
Data Source=test.mdb","", "",adModeUnknown);//连接数据库
//上面一句中连接字符串中的Provider是针对ACCESS2000环境的,
//////对于ACCESS97,需要改为:Provider=Microsoft.Jet.OLEDB.3.51; }
}
}
catch(_com_error e)//捕捉异常
{
CString errormessage;
errormessage.Format("连接数据库失败!\n\n错误信息:%s",e.ErrorMessage());
AfxMessageBox(errormessage);//显示错误信息
return FALSE;
}
#ifdef _AFXDLL
Enable3dControls(); // Call this when using MFC in a shared DLL
#else
Enable3dControlsStatic(); // Call this when linking to MFC statically
#endif
CADOTest1Dlg dlg;
m_pMainWnd = &dlg;
int nResponse = dlg.DoModal();
if (nResponse == IDOK)
{
}
else if (nResponse == IDCANCEL)
{
}
return FALSE;
}

//////////////////////////
int CADOTest1App::ExitInstance()
{
if(m_pConnection->State)
m_pConnection->Close(); //如果已经打开了连接则关闭它
return CWinApp::ExitInstance();
}
```

四、小结

限于篇幅ADO中的许多内容还没有介绍，如绑定方式处理记录集数据、存储过程的调用、事务处理、图像在数据库中的保存与读取、与表格控件的配合使用等。如果读者对上述内容感性认识的话，可以自行参考相关编程资料。

本文引用地址：<http://blog.sciencenet.cn/blog-51026-788237.html> 此文来自科学网郑晖博客，转载请注明出处。

上一篇：[\[转载\]【GAMIT-BLOCK】GAMIT 解算各步骤-详细](#)