

1.点上生成面的代码

```
if (m_pFeatureLayer.FeatureClass.ShapeType == esriGeometryType.esriGeometryPolygon)
{
    IPointCollection m_pPointCollection = new PolygonClass();
    object missing = Type.Missing;
    int icount = newFeature.XLIST.Count;
    if (icount < 3)
        return;
    for (int i = 0; i < icount; i++)
    {
        IPoint point = new PointClass();
        point.PutCoords(newFeature.XLIST[i], newFeature.YLIST[i]);
        m_pPointCollection.AddPoint(point, ref missing, ref missing);
    }

    IPolygon m_pPolygon = m_pPointCollection as IPolygon;
    if (m_pPolygon == null)
    {
        System.Windows.Forms.MessageBox.Show("null");
        return;
    }
    else
    {
        ITopologicalOperator pTopo = m_pPolygon as ITopologicalOperator;
        if (pTopo != null)
        {
            pTopo.Simplify();
        }
    }
    IWorkspaceEdit m_pWorkspaceEdit = m_EngineEditor.EditWorkspace as
IWorkspaceEdit;
    m_pWorkspaceEdit.StartEditOperation();
    IFeature m_pFeature = m_pFeatureLayer.FeatureClass.CreateFeature();
    m_pFeature.Shape = m_pPolygon as IGeometry;
    m_pFeature.Store();
    m_pWorkspaceEdit.StopEditOperation();
}
```

2.文件的打开 保存 另存的代码

```
using System;
```

```

using System.Windows.Forms;

using ESRI.ArcGIS.esriSystem;

using ESRI.ArcGIS.SystemUI;

using ESRI.ArcGIS.Carto;


namespace SaveMapDocument
{
    /// <summary>
    /// Summary description for Form1.
    /// </summary>

    public class SaveMapDocument : System.Windows.Forms.Form
    {
        public System.Windows.Forms.TextBox txtMapDocument;

        public System.Windows.Forms.Button cmdOpen;

        public System.Windows.Forms.Button cmdSave;

        public System.Windows.Forms.Button cmdSaveAs;

        private System.Windows.Forms.OpenFileDialog openFileDialog1;

        private System.Windows.Forms.SaveFileDialog saveFileDialog1;

        private IMapDocument m_MapDocument;

        private ESRI.ArcGIS.Controls.AxToolbarControl axToolbarControl1;

        private ESRI.ArcGIS.Controls.AxPageLayoutControl
axPageLayoutControl1;

        private ESRI.ArcGIS.Controls.AxLicenseControl axLicenseControl1;

        private ESRI.ArcGIS.Controls.AxTOCControl axTOCControl1;
    }
}

```

```

    /// <summary>

    /// Required designer variable.

    /// </summary>

    private System.ComponentModel.Container components = null;


    public SaveMapDocument()
    {
        //

        // Required for Windows Form Designer support

        //

        InitializeComponent();


        //

        // TODO: Add any constructor code after InitializeComponent call

        //
    }


    /// <summary>

    /// Clean up any resources being used.

    /// </summary>

    protected override void Dispose( bool disposing )
    {
        //Release COM objects

        ESRI.ArcGIS.ADF.COMSupport.AOUninitialize.Shutdown();
    }

```

```
        if( disposing )
        {
            if (components != null)
            {
                components.Dispose();
            }
        }
        base.Dispose( disposing );
    }
}
```

```
#region Windows Form Designer generated code
```

```
/// <summary>
/// The main entry point for the application.
/// </summary>
[STAThread]
static void Main()
{
    Application.Run(new SaveMapDocument());
}
```

```
private void Form1_Load(object sender, EventArgs e)
{
    //Add toolbar definitions to the ToolbarControl
```

```
axToolBarControl1.AddToolBarDef("esriControls.ControlsPageLayoutToolb  
ar", -1, false, 0, esriCommandStyles.esriCommandStyleIconOnly);
```

```
axToolBarControl1.AddToolBarDef("esriControls.ControlsGraphicElementT  
oolbar", -1, true, 0, esriCommandStyles.esriCommandStyleIconOnly);
```

```
//Set buddy control
```

```
axToolBarControl1.SetBuddyControl(axPageLayoutControl1);
```

```
axTOCControl1.SetBuddyControl(axPageLayoutControl1);
```

```
cmdSave.Enabled = false;
```

```
cmdSaveAs.Enabled = false;
```

```
}
```

```
private void cmdOpen_Click(object sender, System.EventArgs e)
```

```
{
```

```
//Open a file dialog for opening map documents
```

```
openFileDialog1.Title = "Open Map Document";
```

```
openFileDialog1.Filter = "Map Documents (*.mxd)|*.mxd";
```

```
openFileDialog1.ShowDialog();
```

```
// Exit if no map document is selected
```

```
string sFilePath = openFileDialog1.FileName;
```

```
if (sFilePath == "")
```

```
{  
    return;  
}  
  
//Open document  
OpenDocument((sFilePath));  
  
if (cmdSave.Enabled == false)  
{  
    cmdSave.Enabled = true;  
}  
  
if (cmdSaveAs.Enabled == false)  
{  
    cmdSaveAs.Enabled = true;  
}  
}  
  
private void cmdSave_Click(object sender, System.EventArgs e)  
{  
    //Save changes to the current document  
    SaveDocument();  
}  
  
private void cmdSaveAs_Click(object sender, System.EventArgs e)  
{
```

```

//Open a file dialog for saving map documents

saveFileDialog1.Title = "Save Map Document As";

saveFileDialog1.Filter = "Map Documents (*.mxd)|*.mxd";

saveFileDialog1.ShowDialog();


//Exit if no map document is selected

string sFilePath = saveFileDialog1.FileName;

if (sFilePath == "")
{
    return;
}

if (sFilePath == m_MapDocument.DocumentFilename)
{
    //Save changes to the current document

    SaveDocument();
}

else
{
    //SaveAs a new document with relative paths

    m_MapDocument.SaveAs(sFilePath, true, true);

    //Open document

    OpenDocument((sFilePath));

    MessageBox.Show("Document saved successfully!");
}

```

```

    }
}

private void OpenDocument(string sFilePath)
{
    if (m_MapDocument != null) m_MapDocument.Close();

    //Create a new map document
    m_MapDocument = new MapDocumentClass();

    //Open the map document selected
    m_MapDocument.Open(sFilePath, "");

    //Set the PageLayoutControl page layout to the map document page
    layout
    axPageLayoutControl1.PageLayout = m_MapDocument.PageLayout;

    txtMapDocument.Text = m_MapDocument.DocumentFilename;
}

private void SaveDocument()
{
    //Check that the document is not read only

    if (m_MapDocument.get_IsReadOnly(m_MapDocument.DocumentFilename)
== true)
    {
        MessageBox.Show("This map document is read only!");

        return;
    }
}

```



```

        //Save with the current relative path setting
        m_MapDocument.Save(m_MapDocument.UsesRelativePaths, true);

        MessageBox.Show("Changes saved successfully!");
    }

}

}

```

3.访问一个地图

```

static void OpenMXDViaMapDocument(string path)
{
    IMapDocument pMapDocument = new MapDocumentClass();
    if (pMapDocument.get_IsMapDocument(path))
    {
        pMapDocument.Open(path, null);

        IMap pMap;

        for (int i = 0; i <= pMapDocument.MapCount - 1; i++)
        {
            pMap = pMapDocument.get_Map(i);

            Console.WriteLine(pMap.Name);

            IEnumLayer pEnumLayer = pMap.get_Layers(null, true);

            pEnumLayer.Reset();

            ILayer pLayer = pEnumLayer.Next();

            while (pLayer != null)

```

```

        {
            Console.WriteLine(pLayer.Name);

            pLayer = pEnumLayer.Next();
        }
    }
}

```

4、地图坐标

```

private void axMapControl1_OnMouseMove(object sender,
IMapControlEvents2_OnMouseMoveEvent e)
{
    string outx = null;

    string outy = null;

    IActiveView pActiveView = axMapControl1.ActiveView;

    BJAEIFunction.bj54tojingweiduAo(pActiveView, e.mapX, e.mapY, ref
outx, ref outy);

    labelItem1.Text = "地理坐标:   经度: " + outx + "   纬度: " + outy;

    //IFeatureLayer pFeatLyr;

    //pFeatLyr = axMapControl1.Map.get_Layer(2) as IFeatureLayer;

    //pFeatLyr.DisplayField = "面积";

    //pFeatLyr.ShowTips = true;

```

```

        //string pTips;

        //pTips = pFeatLyr.get_TipText(e.mapX, e.mapY,
pActiveView.FullExtent.Width / 100);

        //toolTip1.SetToolTip(axMapControll, pTips);

    }

```

5、大地转北京 54

```

public static void bj54tojingweiduAo(IActiveView pActiveView, double inx, double iny, ref
string outx, ref string outy)

```

```

    {

        try

        {

            IMap pMap = pActiveView.FocusMap;

            SpatialReferenceEnvironment pSpRE = new
SpatialReferenceEnvironment();

            IGeographicCoordinateSystem pGeoCS =
pSpRE.CreateGeographicCoordinateSystem((int)esriSRGeoCSType.esriSRGeoCS_Beijing1954)
;

            ISpatialReference pSpr = pGeoCS;

            IPoint pPoint = new ESRI.ArcGIS.Geometry.Point();

            pPoint.X = inx;

            pPoint.Y = iny;

            IGeometry pGeo = pPoint;

            pGeo.SpatialReference = pMap.SpatialReference;

```

坐标

pGeo.Project(pSpr); //坐标转换，由当前地图坐标转为北京 54 经纬度

```
double jwd_jd = pPoint.X;
```

```
double jwd_wd = pPoint.Y;
```

```
//转化成度、分、秒
```

```
//经度
```

```
int Jd, Wd, Jf, Wf;
```

```
double temp;
```

```
Single Jm, Wm;
```

```
Jd = (int)jwd_jd; //度
```

```
temp = (jwd_jd - Jd) * 60;
```

```
Jf = (int)temp; //分
```

```
temp = (temp - Jf) * 60;
```

```
Jm = Convert.ToInt32(temp); //秒
```

```
//纬度
```

```
Wd = (int)jwd_wd; //度
```

```
temp = (jwd_wd - Wd) * 60;
```

```
Wf = (int)temp; //分
```

```
temp = (temp - Wf) * 60;
```

```
Wm = Convert.ToInt32(temp); //秒
```

```
outx = Jd + "度" + Jf + "分" + Jm + "秒";
```

```

        outy = Wd + "度" + Wf + "分" + Wm + "秒";
    }

    catch (Exception ex)
    {

        MessageBox.Show(ex.Message);
    }

```

6.拖动代码

```

private void btnClearSelction_Click(object sender, EventArgs e)
{
    ICommand pCommand = new ControlsMapIdentifyToolClass();

    ITool pTool = pCommand as ITool;

    switch (this.tabControl.SelectedTabIndex)
    {
        case 0:
            pCommand.OnCreate(this.mainMapControl.Object);

            this.mainMapControl.CurrentTool = pTool;

            break;

        case 1:
            pCommand.OnCreate(this.axPageLayoutControl.Object);

            this.axPageLayoutControl.CurrentTool = pTool;

            break;
    }
}

```

```
    }  
  
}
```

7.axMapControl 和 axPagelayoutControl 数据同步显示的程序

```
private void axMapControl1_OnMouseDown(object sender,  
IMapControlEvents2_OnMouseDownEvent e)  
  
    {  
  
        axMapControl1.MousePointer =  
esriControlsMousePointer.esriPointerCrosshair;  
  
        IGeometry pGeom = axMapControl1.TrackRectangle();  
  
        DrawMapShape(pGeom);  
  
        axMapControl1.CtlRefresh(esriViewDrawPhase.esriViewGeography,  
null, null);  
    }  
  
private void DrawMapShape(IGeometry pGeom)  
  
    {  
  
        IRgbColor pColor;  
  
        pColor = new RgbColorClass();  
  
        pColor.Red =100;  
  
        pColor.Green =100;  
  
        pColor.Blue =100;  
  
        ISimpleFillSymbol pFillstyl;  
  
        pFillstyl = new SimpleFillSymbolClass();
```

```

        pFillsyl.Color = pColor;

        object oFillsyl = pFillsyl;

        axMapControl1.DrawShape(pGeom, ref oFillsyl);
    }

```

```

private void CopyAndOverwriteMap()
{
    IObjectCopy objectCopy = new ObjectCopyClass();

    object toCopyMap = axMapControl1.Map;

    object copiedMap = objectCopy.Copy(toCopyMap);

    object toOverwriteMap = axPageLayoutControl1.ActiveView.FocusMap;

    objectCopy.Overwrite(copiedMap, ref toOverwriteMap);
}

```

```

private void axMapControl1_OnAfterScreenDraw(object sender,
IMapControlEvents2_OnAfterScreenDrawEvent e)
{
    IActiveView activeView =
    (IActiveView)axPageLayoutControl1.ActiveView.FocusMap;

    IDisplayTransformation displayTransformation =
    activeView.ScreenDisplay.DisplayTransformation;

    displayTransformation.VisibleBounds = axMapControl1.Extent;
}

```

```

        axPageLayoutControl1.ActiveView.Refresh();

        CopyAndOverwriteMap();
    }

    private void axPageLayoutControl1_OnViewRefreshed(object sender,
IPageLayoutControlEvents_OnViewRefreshedEvent e)
    {
        axTOCCControl1.CtlUpdate();

        CopyAndOverwriteMap();
    }

```

8.放大 缩小

放大

```

ICommand pCommand = new ControlsMapZoomInToolClass();

ITool pTool = pCommand as ITool;

pCommand.OnCreate(this.axMapControl1.Object);

this.axMapControl1.CurrentTool = pTool;

```

缩小


```

ICommand pCommand = new ControlsMapZoomOutToolClass();

ITool pTool = pCommand as ITool;

pCommand.OnCreate(this.axMapControl1.Object);

this.axMapControl1.CurrentTool = pTool;

```

9. 在 arcsence 中的各个控件的应用

类似的arcmap也一样。

```

case "ZoomIn":

    {

        ICommand command = new
ControlsSceneZoomInTool();//ControlsSceneZoomInToolClass();

        command.OnCreate(this.axSceneControl1.Object);

        this.axSceneControl1.CurrentTool = command as
ESRI.ArcGIS.SystemUI.ITool;

    }

    break;

case "toolFly":

    {

        ICommand command = new
ControlsSceneFlyToolClass();//ControlsSceneZoomInToolClass();

```

```

        command.OnCreate(this.axSceneControl1.Object);

        this.axSceneControl1.CurrentTool = command as
ESRI.ArcGIS.SystemUI.ITool;

    }

    break;

case "toolSelectFeatures":

    {

        ICommand command = new
ControlsSceneSelectFeaturesToolClass();//ControlsSceneZoomInToolClass();

        command.OnCreate(this.axSceneControl1.Object);

        this.axSceneControl1.CurrentTool = command as
ESRI.ArcGIS.SystemUI.ITool;

    }

    break;

case "toolTargetZoom":

    {

        ICommand command = new
ControlsSceneTargetZoomToolClass();//ControlsSceneZoomInToolClass();

        command.OnCreate(this.axSceneControl1.Object);

        this.axSceneControl1.CurrentTool = command as
ESRI.ArcGIS.SystemUI.ITool;

    }

```

```

        break;

        case "toolFullExtent":

            {

                ICommand command = new
ControlsSceneFullExtentCommandClass();//ControlsSceneZoomInToolClass();

command.OnCreate(this.axSceneControl1.Object);

                this.axSceneControl1.CurrentTool = command as
ESRI.ArcGIS.SystemUI.ITool;

            }

            break;

        case "ZoomOut":

            {

                ICommand command = new
ControlsSceneZoomOutTool();

command.OnCreate(this.axSceneControl1.Object);

                this.axSceneControl1.CurrentTool = command as
ESRI.ArcGIS.SystemUI.ITool;

            }

            break;

        case "Pan":

            {

```

```

        ICommand command = new ControlsScenePanTool();

command.OnCreate(this.axSceneControl1.Object);

        this.axSceneControl1.CurrentTool = command as
ESRI.ArcGIS.SystemUI.ITool;

    }

    break;

    case "Navigate":

    {

        ICommand command = new
ControlsSceneNavigateTool();

command.OnCreate(this.axSceneControl1.Object);

        this.axSceneControl1.CurrentTool = command as
ESRI.ArcGIS.SystemUI.ITool;

```

10.在 C#中如何连接 ACCESS 数据库

```

using System.Data.OleDb

public string myConnstring="Provider=Microsoft.Jet.OLEDB.4.0;

Data Source="+HttpContext.Current.Server.MapPath("data.mdb"); //data.mdb
是你的数据库名称

OleDbConnection MyConnection;

MyConnection = new OleDbConnection(myConnstring);

strInsert=""; //strinsert 是你的 sql 语句

OleDbCommand MyCommand = new OleDbCommand(strInsert,MyConnection);

MyConnection.Open();

MyCommand.ExecuteNonQuery();

MyConnection.Close();

```

11.创建文件（word）

//创建文件夹

```

System.Object Nothing = System.Reflection.Missing.Value;
Directory.CreateDirectory("c:/CNSI"); //创建文件所在目录
string name = "CNSI_" + DateTime.Now.ToLongDateString() +
".doc";
object filename = "c://CNSI//" + name; //文件保存路径

```

```

//创建Word文档
Microsoft.Office.Interop.Word.Application WordApp = new
Microsoft.Office.Interop.Word.ApplicationClass();
Microsoft.Office.Interop.Word.Document WordDoc =
WordApp.Documents.Add(ref Nothing, ref Nothing, ref Nothing, ref
Nothing);
IQueryFilter pQueryFilter;
pQueryFilter = pFeatureWorkspace;

```

保存文档：

```

string name = "y00";
object filename = "c://CNSI//" + name;//保存文件路径。
object Nothing = System.Reflection.Missing.Value;
//Microsoft.Office.Interop.Word.Application WordApp =
new Microsoft.Office.Interop.Word.ApplicationClass();
Microsoft.Office.Interop.Word.Document WordDocs =
WordApp.Documents.Add(ref Nothing, ref Nothing, ref Nothing, ref
Nothing);

WordDocs.Close(ref Nothing, ref Nothing, ref Nothing);
WordApp.Quit(ref Nothing, ref Nothing, ref Nothing);
WordDocs.SaveAs(ref filename, ref Nothing, ref Nothing,
ref Nothing, ref Nothing, ref Nothing, ref Nothing, ref
Nothing, ref Nothing, ref Nothing, ref Nothing, ref Nothing,
ref Nothing, ref Nothing);

//WordApp.Quit(ref Nothing, ref Nothing, ref Nothing);
message = name + "文档生成成功，以保存到C:CNSI下";

```

hy

```

catch
{
    message = "文件导出异常! ";
}
string path = "E:\\yxl\\tianjia\\database2.mdb";
IWorkspace iW = AccessWorkspaceFromPropertySet(path);
IFeatureWorkspace pFeatureWorkspace =
(IFeatureWorkspace)iW;
ITable pTable = pFeatureWorkspace.OpenTable("database2");

```

12. C#操作 Access 数据库的方法

//取得连接

```
public OleDbConnection getConn()
{
    ConnectDatabase connstr=new ConnectDatabase();
    string connStr=connstr.GetConnectionString();
    OleDbConnection oledb=new OleDbConnection(connStr);
    return oledb;
}
```

(1) 采用 OleDbCommand,OleDbDataReader 访问数据库

1. 查询

```
public User getUserFromName(string Searchname)
{
    User tempUser=new User();
    try
    {
        OleDbConnection oleconn=getConn();//数据库连接
        string strSel="select * from MyUser where
        UserName='"+Searchname+"'";//查询语句
        OleDbCommand myCommand=new OleDbCommand(strSel,oleconn);//查询命令
        oleconn.Open();//打开数据库连接
        OleDbDataReader reader;
        reader=myCommand.ExecuteReader();//执行查询命令，返回记录集
        if(reader.Read())
        {
            tempUser.ID=(int)reader["UserID"];
            tempUser.Name=reader["UserName"].ToString();
            tempUser.Salary=(float)reader["UserSalary"];
            tempUser.Password=reader["UserPassword"].ToString();
            tempUser.Memo=reader["UserMemo"].ToString();
            tempUser.Birthday=(DateTime)reader["UserBirthday"];
            tempUser.Address=reader["UserAddress"].ToString();
        }
        else
    }
```

```

    {
        throw new Exception("没有记录");
    }
    reader.Close(); //关闭记录集
    oleconn.Close(); //关闭连接

}
catch(Exception e)
{
    throw new Exception("打开数据库出错"+e.Message);
}
return tempUser;
}

```

2. 插入记录

```

public void InsertUser(User insertuser)
{
    try
    {
        OleDbConnection oleconn=getConn(); //数据库连接
        oleconn.Open(); //打开数据库连接
        string strSel="insert into [MyUser]([UserName],[UserPassword],
[UserSalary],[UserAddress],[UserBirthday],[UserMemo])"; //插入语句
        strSel+=" values

('"+insertuser.Name+"', '"+insertuser.Password+"', '"+insertuser.Salary.
ToSt

ring());
        strSel+=", '"+insertuser.Address+"', '"+insertuser.Birthday.ToStrin
g()

+"#, '"+insertuser.Memo+"')";
        OleDbCommand myCommand=new OleDbCommand(strSel, oleconn); //查询命令

        myCommand.ExecuteNonQuery();
        oleconn.Close(); //关闭连接
    }
}

```



```

    }
    catch(Exception e)
    {
        throw new Exception("打开数据库出错"+e.Message);
    }
}

```

3. 删除记录

```

public void DeleteUser(int m_id)
{
    try
    {
        OleDbConnection oleconn=getConn();
        oleconn.Open();
        string strSel="Delete From [Myuser] where UserID="+m_id.ToString();
        OleDbCommand myCommand=new OleDbCommand(strSel, oleconn);
        myCommand.ExecuteNonQuery();
        oleconn.Close();
    }
    catch(Exception e)
    {
        throw new Exception("删除记录出错"+e.Message);
    }
}

```

(2) 采用

OleDbDataAdapter, OleDbCommandBuilder, DataSet, DataTable, DataRow 访问数据库

问数据库

添加记录如下

```

public void InsertUserA(User insertUser)
{
    using(OleDbConnection conn=getConn())
    {
        OleDbDataAdapter adapter = new OleDbDataAdapter();
        string queryString="Select * from MyUser order by UserID";
        adapter.SelectCommand = new OleDbCommand(queryString, conn);
    }
}

```

```

OleDbCommandBuilder builder = new OleDbCommandBuilder(adapter);
        // builder.QuotePrefix="[";
        // builder.QuoteSuffix="]";
conn.Open();

DataSet users = new DataSet();
adapter.Fill(users, "MyUser");
DataTable dt=new DataTable();
dt=users.Tables["MyUser"];
DataRow r=dt.NewRow();
r["UserName"]=insertUser.Name;
r["UserPassword"]=insertUser.Password;
r["UserAddress"]=insertUser.Address;

r["UserSalary"]=insertUser.Salary;
r["UserBirthday"]=insertUser.Birthday;
r["UserMemo"]=insertUser.Memo;
dt.Rows.Add(r);
adapter.Update(users, "MyUser");

    }
}

```

需要注意字段不能和关键字相同，否则会出现 Insert into 出错的提示。解决办法在

前一篇

(3)采用参数化查询的方式

```

public class AccessUtil
{
public AccessUtil()
{
}

    private string connString;

    public string ConnString
    {

```

```

        get { return connString; }
        set { connString = value; }
    }
    public AccessUtil(string connstr)
    {
        this.connString = connstr;
    }
    //带参数的插入语句, 返回值为 id 关键字的值, 单条插入语句
    public int ExecuteInsert(string SQL, OleDbParameter[]
parameters)
    {
        using(OleDbConnection conn=new OleDbConnection(connString))
        {
            OleDbCommand cmd = new OleDbCommand(SQL, conn);
            try
            {
                conn.Open();
                if (parameters!=null)
                {
                    cmd.Parameters.AddRange(parameters);
                }
                cmd.ExecuteNonQuery();
                cmd.CommandText = @"Select @@identity";
                int value =
Int32.Parse(cmd.ExecuteScalar().ToString
());

                return value;
            }
            catch (System.Exception e)
            {
                throw e;
            }
        }
    }
    //不带参数的插入语句, 返回值为关键字的值
    public int ExecuteInsert(string SQL)

```

```

    {
        return ExecuteInsert(SQL, null);
    }
    //带参数的插入、删除、更新语句，返回受影响的记录的个数
    public int ExecuteNonQuery(string SQL, OleDbParameter[]
parameters)
    {
        using(OleDbConnection conn=new OleDbConnection(connString))
        {
            conn.Open();
            OleDbCommand cmd = new OleDbCommand(SQL, conn);
            try
            {
                if (parameters!=null)
                {
                    cmd.Parameters.AddRange(parameters);
                }
                int rows=cmd.ExecuteNonQuery();
                return rows;
            }
            catch (System.Exception e)
            {
                throw e;
            }
        }
    }
    //不带参数的插入、删除、更新语句，返回受影响的记录的个数
    public int ExecuteNonQuery(string SQL)
    {
        return ExecuteNonQuery(SQL, null);
    }
    //带参数的查询语句，返回所查询到的记录集
    public DataSet ExecuteQuery(string SQL, OleDbParameter[]
parameters)
    {
        using(OleDbConnection conn=new OleDbConnection(connString))
        {

```

```

        DataSet ds = new DataSet();
        try
        {
            conn.Open();
            OleDbDataAdapter da = new OleDbDataAdapter(SQL,
conn);

            if (parameters != null)
            {
                da.SelectCommand.Parameters.AddRange(paramete
rs);
            }
            da.Fill(ds, "ds");
        }
        catch(System.Exception e)
        {
            throw e;
        }
        return ds;
    }
}
//不带参数的查询，返回所查询到的记录集
public DataSet ExecuteQuery(string SQL)
{
    return ExecuteQuery(SQL, null);
}
}

```

```

class ManageUser
{
    //Access 数据库工具对象
    AccessUtil accessutil = new AccessUtil

    (ConnectDatabase.GetConnectionString());
    public ArrayList GetAllUserArr()//获得 User 表中的所有记录，存
储进
ArrayList。
    {

```

```

        string SQL = "select * from MyUser order by ID";
        DataSet ds=accessutil.ExecuteQuery(SQL); //返回的临时表的
名称
为 “ds”

        /*
            ArrayList arr = new ArrayList();
            for (int i = 0; i < ds.Tables
["ds"].Rows.Count;i++ )
            {
                arr.Add(DataRow2User(ds.Tables["ds"].Rows
[i]));
            }*/
        ArrayList arr = DataTable2ArrayList(ds.Tables["ds"]);
        return arr;

    }
    public DataSet GetAllUserDataSet() //存储成 DataSet
    {
        string SQL = "select * from MyUser order by ID";
        DataSet ds = accessutil.ExecuteQuery(SQL);
        return ds;
    }
    private User DataRow2User(DataRow dr) //将数据表中的一条记录转换
为
一个 User 类的实例
    {
        User user = new User();
        user.ID = Int32.Parse(dr["ID"].ToString());
        user.Name = dr["Name"].ToString();
        user.Address = dr["Address"].ToString();
        user.Birthday = Convert.ToDateTime(dr
["Birthday"].ToString());
        user.Memo = dr["Memo"].ToString();
        user.Salary =(float) Convert.ToDouble(dr

```

```

["Salary"].ToString());
        user.Password = dr["Password"].ToString();
        return user;
    }
    private ArrayList DataTable2ArrayList(DataTable dt)//将一个表中的
    的

```

记录转化为 ArrayList 对象

```

    {
        ArrayList tempArr = new ArrayList();
        DataTableReader dr = new DataTableReader(dt);
        while(dr.Read())
        {
            User user = new User();
            user.ID = Int32.Parse(dr["ID"].ToString());
            user.Name = dr["Name"].ToString();
            user.Address = dr["Address"].ToString();
            user.Birthday =
Convert.ToDateTime(dr["Birthday"].ToString());
            user.Memo = dr["Memo"].ToString();
            user.Salary = (float)Convert.ToDouble(dr["Salary"].ToString
());
            user.Password = dr["Password"].ToString();
            tempArr.Add(user);
        }
        return tempArr;
    }
    public DataSet GetUserByName(string name)
    {
        String SQL = "Select * from MyUser where Name=?";
        OleDbParameter[] parameter = new OleDbParameter[1];
        parameter[0] = new OleDbParameter("@Name",
OleDbType.VarChar);
        parameter[0].Value = name;
        DataSet dt= accessutil.ExecuteQuery(SQL, parameter);
        return dt;
    }

```

```

public int InsertUser(User inUser)
{
    String SQL = "insert into [MyUser] ([Name], [Password],
[Salary], [Address], [Birthday], [Memo]) values(?, ?, ?, ?, ?, ?)";
    OleDbParameter[] parameters = new OleDbParameter[6];
    parameters[0] = new OleDbParameter("@Name",
OleDbType.VarChar);
    parameters[0].Value = inUser.Name;
    parameters[1] = new OleDbParameter("@Password",
OleDbType.VarChar);
    parameters[1].Value = inUser.Password;
    parameters[2] = new OleDbParameter("@Salary",
OleDbType.Single);
    parameters[2].Value = inUser.Salary;
    parameters[3] = new OleDbParameter("@Address",
OleDbType.VarChar);
    parameters[3].Value = inUser.Address;
    parameters[4] = new OleDbParameter("@Birthday",
OleDbType.Date);
    parameters[4].Value = inUser.Birthday;
    parameters[5] = new OleDbParameter("@Memo",
OleDbType.VarChar);
    parameters[5].Value = inUser.Memo;
    return accessutil.ExecuteInsert(SQL, parameters);
}

public void DelUserById(int id)
{
    String SQL = "DELETE FROM [MyUser] where ID=?";
    OleDbParameter[] parameters = new OleDbParameter[1];
    parameters[0] = new OleDbParameter("@ID",
OleDbType.Integer);
    parameters[0].Value = id;

```



```

        accessutil.ExecuteNoQuery(SQL, parameters);
    }
    public void UpdateUser(User userupdate)
    {
        String SQL = "update [MyUser] Set [Name]=?, [Password]=?,
[Salary]=?, [Address]=?, [Birthday]=?, [Memo]=? where [ID]=?";
        OleDbParameter[] parameters = new OleDbParameter[7];
        parameters[0] = new OleDbParameter("@Name",
OleDbType.VarChar);
        parameters[0].Value = userupdate.Name;
        parameters[1] = new OleDbParameter("@Password",
OleDbType.VarChar);
        parameters[1].Value = userupdate.Password;
        parameters[2] = new OleDbParameter("@Salary",
OleDbType.Single);
        parameters[2].Value = userupdate.Salary;
        parameters[3] = new OleDbParameter("@Address",
OleDbType.VarChar);
        parameters[3].Value = userupdate.Address;
        parameters[4] = new OleDbParameter("@Birthday",
OleDbType.Date);
        parameters[4].Value = userupdate.Birthday;
        parameters[5] = new OleDbParameter("@Memo",
OleDbType.VarChar);
        parameters[5].Value = userupdate.Memo;
        parameters[6] = new OleDbParameter("@ID",
OleDbType.Integer);
        parameters[6].Value = userupdate.ID;
        accessutil.ExecuteNoQuery(SQL, parameters);
    }
}

```

13.加载 cad 代码

```
private void AddCADRasterLayer(string sPath, string sName)

{

    IWorkspaceFactory pCadWorkspaceFactory= new
    CadWorkspaceFactoryClass();

    IWorkspace pWorkspace = pCadWorkspaceFactory.OpenFromFile(sPath,0);

    ICadDrawingWorkspace pCadDrawingWorkspace = pWorkspace as
    ICadDrawingWorkspace;

    ICadDrawingDataset pCadDataset; //"定义并获得 CAD 文件的数据集

    pCadDataset = pCadDrawingWorkspace.OpenCadDrawingDataset(sName);

    ICadLayer pCadLayer; //"加入图层到 Map 对象中去

    pCadLayer = new CadLayerClass();

    pCadLayer.CadDrawingDataset = pCadDataset;

    axMapControl1.AddLayer(pCadLayer, 0);

    axMapControl1.Refresh();

}

private void button1_Click(object sender, EventArgs e)

{
```

```
OpenFileDialog openFileDialog1 = new OpenFileDialog();

openFileDialog1.Filter = "所有文件|*.*";


if (openFileDialog1.ShowDialog() == DialogResult.OK)
{

    AddCADRasterLayer(openFileDialog1.FileName,"11.dwg");//

}

}
```