New Clip, Export & Zip SDE Feature Classes UPDATED 2/08

Contributed by Bert Granberg
26, Nov. 2007
Last Updated 05, Feb. 2008

```
'Code updated Feb 5th, 2007
'reason: to replace the ESRI Geoprocessor ConvertToShapfile function for better performance
'bgranberg@utah.gov

'These VBA functions are used to clip and export the feature classes on the SGID SDE database server
'into zipped the file geodatabase and shapefile products available from AGRC's FTP site. To use the
'script, make a selection of feature classes or feature datasets in the ArcCatalog Contents view
'(usually the right side of ArcCatalog). Then run the CreateOutputFromArcCatalogSelection() VBA sub
'procedure.


'Notes:

'Be sure to change the value of the variable m_outputLocation to point at a valid folder

'This assumes will likely error if you attampt to write over existing files. So clear the m_outputLocation folder before
running

'The Utah SGID's ftp.agrc.state.ut.us/SGID_Vector folder structure is organized as follows:

'SGID_Vector folder
'. . [Category name folders ]
'. . . . [feature class name folder(s)]
'. . . . . . statewide folder
'. . . . . geodatabase folder
'. . . . . shapefile folder
'. . . . . county folder (clipped extents for each of Utah's 29 counties)
'. . . . . [county name folders]
'. . . . . geodatabase folder
'. . . . . shapefile folder

Option Explicit

Private m_deleteColl As Collection
Private m_pSDEFWS As IFeatureWorkspace
Private m_pSDEWSFactory As IWorkspaceFactory
Private m_GP 'As Object
Private m_CoClipFC As IFeatureClass
Private m_outputLocation As String
Private m_sourceMetadata As IMetadata

Private Sub CreateOutputFromArcCatalogSelection()

    Set m_GP = CreateObject("esriGeoprocessing.GPDispatch.1")
    Set m_deleteColl = New Collection
    m_outputLocation = "c:\temp\"
    m_GP.RefreshCatalog m_outputLocation

    Open m_outputLocation & "sdeexp_logfile_" & Format(Now, "yymmddhhmm") & ".txt" For Output As #1
    Open m_outputLocation & "sdeexp_errfile_" & Format(Now, "yymmddhhmm") & ".txt" For Output As #2

    If Not m_GP.Exists(m_outputLocation & "NAD83\") Then
        m_GP.CreateFolder m_outputLocation, "NAD83"
    End If

    m_outputLocation = m_outputLocation & "NAD83"
    m_GP.ScratchWorkspace = m_outputLocation
```

```
    Call openSDEWS
    Set m_CoClipFC = m_pSDEFWS.OpenFeatureClass("SGID.U024.CountyBoundaries")

    Dim pGxApp As IGxApplication
    Dim pSelGxObject As IGxObject
    Dim pGxSel As IGxSelection
    Dim pEnumGxObjSel As IEnumGxObject
    Set pGxApp = Application
    Set pGxSel = pGxApp.Selection
    Set pEnumGxObjSel = pGxSel.SelectedObjects
    pEnumGxObjSel.Reset
    Set pSelGxObject = pEnumGxObjSel.Next

    Do Until pSelGxObject Is Nothing
        If pSelGxObject.Category = "SDE Feature Class" Then
            CreateOutputFromFeatureClassGxObject pSelGxObject
        ElseIf pSelGxObject.Category = "SDE Feature Dataset" Then
            Dim pGxObjContainer As IGxObjectContainer
            Dim pEnumGxObj As IEnumGxObject
            Dim pCurrGxObject As IGxObject
            Set pGxObjContainer = pSelGxObject

            If pGxObjContainer.HasChildren Then
                Set pEnumGxObj = pGxObjContainer.Children
                If Not pEnumGxObj Is Nothing Then
                    Set pCurrGxObject = pEnumGxObj.Next
                    Do Until pCurrGxObject Is Nothing
                        If pCurrGxObject.Category = "SDE Feature Class" Then
                            Debug.Print pCurrGxObject.FullName
                            Print #1, "----------------------------------"
                            Print #1, pCurrGxObject.FullName
                            CreateOutputFromFeatureClassGxObject pCurrGxObject
                        End If
                        Set pCurrGxObject = pEnumGxObj.Next
                    Loop
                End If
            End If
        End If
        Set pSelGxObject = pEnumGxObjSel.Next
    Loop
    Close #1
    Close #2
End Sub

Private Sub CreateOutputFromFeatureClassGxObject(pInGxObject As IGxObject)

    Dim pGXDataset As IGxDataset
    Dim pDataset As IDataset
    Dim pFC As IFeatureClass
    Dim pCoClipFC As IFeatureClass
    Dim pCofcursor As IFeatureCursor
    Dim pcofeature As IFeature
    Dim pCoShape As IPolygon
    Dim pCoShapeTopOp As ITopologicalOperator
    Dim countyName As String
    Dim pDatasetName As IDatasetName
    Dim pSQLPrivilege As ISQLPrivilege

    Set pGXDataset = pInGxObject
    Set pDatasetName = pGXDataset.datasetname
    Set pSQLPrivilege = pDatasetName 'QI

    If pSQLPrivilege.SQLPrivileges / 2 <> CInt(pSQLPrivilege.SQLPrivileges / 2) And _
       pSQLPrivilege.SQLPrivileges > 0 Then
```

```
        Set pDataset = pGXDataset.Dataset

        Set pFC = pDataset 'QI
        Debug.Print " **** Starting Statewide: " & pDataset.name & " (" & Now & ")"
        Print #1, " **** Starting Statewide: " & pDataset.name & " (" & Now & ")"

        CreateFileBasedOutputFromFC pFC, m_outputLocation, Nothing, "statewide"

        Set pCofcursor = m_CoClipFC.Search(Nothing, True)
        Set pcofeature = pCofcursor.NextFeature

        Do Until pcofeature Is Nothing
            countyName = UCase(Replace(pcofeature.Value(m_CoClipFC.FindField("NAME")), " ", ""))
            Debug.Print " **** Starting " & countyName & " County: " & pDataset.name & " (" & Now & ")"
            Print #1, " **** Starting " & countyName & " County: " & pDataset.name & " (" & Now & ")"

            Set pCoShape = pcofeature.ShapeCopy
            Set pCoShapeTopOp = pCoShape 'QI
            pCoShapeTopOp.Simplify

            CreateFileBasedOutputFromFC pFC, m_outputLocation, pCoShape, countyName
            Set pcofeature = pCofcursor.NextFeature
        Loop

        Dim emptyGDBPath As Variant
        Set m_pSDEFWS = Nothing
        Set m_pSDEWSFactory = Nothing
        For Each emptyGDBPath In m_deleteColl
            Debug.Print "     Delete " & emptyGDBPath
            Print #1, "     Deleting, no features in " & emptyGDBPath
            Kill emptyGDBPath & "\*"
            RmDir emptyGDBPath
        Next
        Set m_deleteColl = New Collection
    Else
        Debug.Print "NO SELECT RIGHT: " & pGXDataset.datasetname.name
    End If
End Sub
Public Sub openSDEWS()
        Dim pCOWS As IWorkspace
        Dim pPropSet As IPropertySet
        Set pPropSet = New PropertySet
        With pPropSet
            .SetProperty "Server", "direct connect"
            .SetProperty "Instance", "sde:sqlserver:168.177.18.211\AGRC"
            .SetProperty "user", "agrc"
            .SetProperty "database", "sgid"
            .SetProperty "password", "agrc"
            .SetProperty "version", "sde.default"
        End With
        Set m_pSDEWSFactory = New SdeWorkspaceFactory
        Set m_pSDEFWS = m_pSDEWSFactory.Open(pPropSet, 0)

End Sub
Public Function createFileGDBWorkspace(location As String, name As String) _
            As IWorkspaceName

    On Error GoTo EH
    Set createFileGDBWorkspace = Nothing

    ' create the FileGDB Workspace factory
    Dim pWorkspaceFactory As IWorkspaceFactory
    'Set pWorkspaceFactory = New AccessWorkspaceFactory
    Set pWorkspaceFactory = New FileGDBWorkspaceFactory
```

```
    Dim pWorkspaceName As IWorkspaceName
    Set pWorkspaceName = pWorkspaceFactory.Create(location, name, Nothing, 0)

    Set createFileGDBWorkspace = pWorkspaceName
    Exit Function

EH:
    MsgBox Err.Number, vbInformation, "createFileGDBWorkspace"
End Function
Private Function pGDBExists(mypath As String) As Boolean
    Dim pGPValue As IGPValue
    Set pGPValue = New DEWorkspace
    pGPValue.SetAsText mypath
    Dim pDEUtil As IDEUtilities
    Set pDEUtil = New DEUtilities
    pGDBExists = pDEUtil.Exists(pGPValue)
End Function
Private Function CreateNew_FC_ForExistingFC(pInFC As IFeatureClass, pFeatureDataset As IFeatureDataset) As
IFeatureClass 'must be sde feature class

    Dim fcname As String
    Dim fDSname As String
    Dim mypath As String

    Dim pInFCDataset As IDataset
    Dim pInFields As IFields
    Dim pClone As IClone
    Dim pNewField As IField
    Dim pNewFieldEdit As IFieldEdit
    Dim pNewFields As IFields
    Dim pNewFieldsEdit As IFieldsEdit
    Dim pUID As IUID

    Set pInFCDataset = pInFC 'qi

    fcname = Replace(pInFCDataset.name, ".", "_")

    'Get editiable, non shape field count
    Dim i As Integer
    Dim fCnt As Integer
    fCnt = 0

    For i = 0 To pInFC.Fields.FieldCount - 1
        If pInFC.Fields.Field(i).Editable And pInFC.Fields.Field(i).Type <> esriFieldTypeGeometry And
UCase(Left(pInFC.Fields.Field(i).name, 8)) <> "OBJECTID" Then
            fCnt = fCnt + 1
        End If
    Next i

    Set pNewFields = New Fields
    Set pNewFieldsEdit = pNewFields
    pNewFieldsEdit.FieldCount = fCnt + 2 'for geometry field and ObjectID

    Set pNewField = New Field
    Set pNewFieldEdit = pNewField
    With pNewFieldEdit
        .name = "OBJECTID"
        .Type = esriFieldTypeOID
    End With

    Set pNewFieldsEdit.Field(0) = pNewField

    Dim shpFieldName As String
    shpFieldName = pInFC.ShapeFieldName
    Dim pOldShapeField As IField
```

```
    Set pOldShapeField = pInFC.Fields.Field(pInFC.FindField(shpFieldName))

    Dim pOldGeometryDef As IGeometryDef
    Dim pNewGeometryDef As IGeometryDef
    Dim pNewGeometryDefEdit As IGeometryDefEdit
    Set pOldGeometryDef = pOldShapeField.GeometryDef
    Set pNewGeometryDef = New GeometryDef
    Set pNewGeometryDefEdit = pNewGeometryDef
    With pNewGeometryDefEdit
        Set .SpatialReference = pOldGeometryDef.SpatialReference
        .GeometryType = pOldGeometryDef.GeometryType
        .GridCount = 2
        .GridSize(0) = calcSpatialIndex(pInFC)
        .GridSize(1) = calcSpatialIndex(pInFC) * 3
        .HasM = pOldGeometryDef.HasM
        .HasZ = pOldGeometryDef.HasZ
    End With

    Set pNewField = New Field
    Set pNewFieldEdit = pNewField
    With pNewFieldEdit
        .name = shpFieldName
        .Type = pOldShapeField.Type
        Set .GeometryDef = pNewGeometryDef
    End With

    Set pNewFieldsEdit.Field(1) = pNewField

    Dim addCnt As Integer
    addCnt = 0
    For i = 0 To pInFC.Fields.FieldCount - 1
        If pInFC.Fields.Field(i).Editable And pInFC.Fields.Field(i).Type <> esriFieldTypeGeometry And
UCase(Left(pInFC.Fields.Field(i).name, 8)) <> "OBJECTID" Then
            'Debug.Print pInFC.Fields.Field(i).name
            Set pClone = pInFC.Fields.Field(i)
            Set pNewField = New Field
            Set pNewField = pClone.Clone
            Set pNewFieldsEdit.Field(addCnt + 2) = pNewField
            addCnt = addCnt + 1
        End If
    Next i


    Set pUID = New UID
    pUID.Value = "esriGeoDatabase.Feature"

    Set CreateNew_FC_ForExistingFC = pFeatureDataset.CreateFeatureClass(fcname, pNewFields, pUID, Nothing,
esriFTSimple, "SHAPE", "")

    Dim targetMetadata As IMetadata
    Dim xmlPropSet As IXmlPropertySet

    Set m_sourceMetadata = pInFC 'QI
    Set targetMetadata = CreateNew_FC_ForExistingFC 'QI

    Set xmlPropSet = m_sourceMetadata.Metadata
    targetMetadata.Metadata = xmlPropSet


End Function



Public Function CreateNew_GDB_ForExistingFC(pInFC As IFeatureClass, ByVal inLocation As String, inFolderName As
String) As IFeatureWorkspace
```

```
    Dim fcname As String
    Dim fDSname As String
    Dim mypath As String
    Dim bWorkspExist As Boolean
    Dim pGDBWSF As IWorkspaceFactory
    Dim pInFCDataset As IDataset
    Set pInFCDataset = pInFC

    fcname = Replace(pInFCDataset.name, ".", "_")
    fDSname = Right(pInFC.FeatureDataset.name, (Len(pInFC.FeatureDataset.name) -
InStrRev(pInFC.FeatureDataset.name, ".")))

    If Not m_GP.Exists(inLocation & "\" & fDSname & "\") Then
        m_GP.CreateFolder inLocation, fDSname
    End If
    If Not m_GP.Exists(inLocation & "\" & fDSname & "\" & fcname & "\") Then
        m_GP.CreateFolder inLocation & "\" & fDSname, fcname
    End If

    If inFolderName = "statewide" Then

        If Not m_GP.Exists(inLocation & "\" & fDSname & "\" & fcname & "\" & "statewide" & "\") Then
            m_GP.CreateFolder inLocation & "\" & fDSname & "\" & fcname, "statewide"
        End If
        If Not m_GP.Exists(inLocation & "\" & fDSname & "\" & fcname & "\" & "statewide" & "\geodatabase\") Then
            m_GP.CreateFolder inLocation & "\" & fDSname & "\" & fcname & "\" & "statewide", "geodatabase"
        End If
        If Not m_GP.Exists(inLocation & "\" & fDSname & "\" & fcname & "\" & "statewide" & "\shapefile\") Then
            'CreateFolder inLocation & "\" & fDSname & "\" & fCname & "\" & "statewide" & "\shapefile\"
            m_GP.CreateFolder inLocation & "\" & fDSname & "\" & fcname & "\" & "statewide", "shapefile"
        End If
        inLocation = inLocation & "\" & fDSname & "\" & fcname & "\" & "statewide" & "\geodatabase\"
        'mypath = inLocation & fCname & ".mdb"
        mypath = inLocation & Replace(fcname, "\", "") & ".gdb"
    Else

        If Not m_GP.Exists(inLocation & "\" & fDSname & "\" & fcname & "\" & "county" & "\") Then
            m_GP.CreateFolder inLocation & "\" & fDSname & "\" & fcname, "county"
        End If
        If Not m_GP.Exists(inLocation & "\" & fDSname & "\" & fcname & "\" & "county" & "\" & inFolderName & "\") Then
            m_GP.CreateFolder inLocation & "\" & fDSname & "\" & fcname & "\" & "county", inFolderName
        End If
        If Not m_GP.Exists(inLocation & "\" & fDSname & "\" & fcname & "\" & "county" & "\" & inFolderName & "\geodatabase\")
Then
            m_GP.CreateFolder inLocation & "\" & fDSname & "\" & fcname & "\" & "county" & "\" & inFolderName, "geodatabase"
        End If
        If Not m_GP.Exists(inLocation & "\" & fDSname & "\" & fcname & "\" & "county" & "\" & inFolderName & "\shapefile\")
Then
            m_GP.CreateFolder inLocation & "\" & fDSname & "\" & fcname & "\" & "county" & "\" & inFolderName, "shapefile"
        End If

        inLocation = inLocation & "\" & fDSname & "\" & fcname & "\" & "county" & "\" & inFolderName & "\geodatabase\"
        'mypath = inLocation & fCname & ".mdb"
        mypath = inLocation & Replace(fcname, "\", "") & ".gdb"

    End If


    bWorkspExist = pGDBExists(mypath)
    If bWorkspExist = False Then
        Call createFileGDBWorkspace(inLocation, fcname)
        'Set pGDBWSF = New AccessWorkspaceFactory
        Set pGDBWSF = New FileGDBWorkspaceFactory
        Set CreateNew_GDB_ForExistingFC = pGDBWSF.OpenFromFile(mypath, 0)
    Else
```

```
        Set CreateNew_GDB_ForExistingFC = Nothing
    End If

End Function
Public Function CreateNew_FDS_ForExistingFC(pInFC As IFeatureClass, pInGDB As IFeatureWorkspace) As
IFeatureDataset
    Dim fDSname As String
    Dim pInGeoDataset As IGeoDataset
    Dim pInFCDataset As IDataset
    Dim pFDS As IFeatureDataset
    Set pInFCDataset = pInFC
    Set pInGeoDataset = pInFCDataset
    Set pFDS = pInFC.FeatureDataset

    fDSname = Mid(pFDS.name, InStrRev(pFDS.name, ".") + 1)

    Set CreateNew_FDS_ForExistingFC = pInGDB.CreateFeatureDataset(fDSname, pInGeoDataset.SpatialReference)

End Function
Public Sub CreateNew_Domains_ForExistingFC(pInFC As IFeatureClass, pEmptyGDB As IFeatureWorkspace)
    Dim pInFields As IFields
    Dim pInField As IField
    Dim pWorkspaceDomains As IWorkspaceDomains
    Dim pNewDomain As IDomain
    Dim pClone As IClone
    Dim f As Integer

    Set pWorkspaceDomains = pEmptyGDB 'qi
    Set pInFields = pInFC.Fields
    For f = 0 To pInFields.FieldCount - 1
        Set pInField = pInFields.Field(f)
        'Debug.Print pInField.name
        If Not pInField.Domain Is Nothing Then
            'Debug.Print "   d- " & pInField.Domain.name & " (" & pInField.Domain.Type & ")"
            If pWorkspaceDomains.DomainByName(pInField.Domain.name) Is Nothing Then
                Set pClone = pInField.Domain
                Set pNewDomain = pClone.Clone
                pWorkspaceDomains.AddDomain pNewDomain
            End If
        End If
    Next f

End Sub
Public Sub CreateFileBasedOutputFromFC(pInFC As IFeatureClass, inLoc As String, pClipItem As Variant,
inFolderName As String)

    Dim pGDB As IFeatureWorkspace
    Dim pFDS As IFeatureDataset
    Dim pNewFC As IFeatureClass
    Dim pNewDataset As IDataset
    Set pGDB = CreateNew_GDB_ForExistingFC(pInFC, inLoc, inFolderName)
    Set pFDS = CreateNew_FDS_ForExistingFC(pInFC, pGDB)
    CreateNew_Domains_ForExistingFC pInFC, pGDB
    Set pNewFC = CreateNew_FC_ForExistingFC(pInFC, pFDS)
    PopulateFeatureClass_ForExistingFeatureClass pInFC, pNewFC, pClipItem
    Set pNewDataset = pNewFC 'qi'

    Dim newFCPathStr As String
    Dim newFDSNameStr As String
    Dim newFCNameStr As String
    Dim newSHPPathStr As String
    Dim featureCount As Long
    featureCount = pNewFC.featureCount(Nothing)
    newFCPathStr = pNewDataset.Workspace.pathName
    newFDSNameStr = pNewFC.FeatureDataset.name
```

```
    newFCNameStr = pNewDataset.name
    Set pNewFC = Nothing
    Set pNewDataset = Nothing
    Set pFDS = Nothing
    Set pGDB = Nothing


    If featureCount > 0 Then
        'Zip Files
        Dim retval
        'Zip GDB

        'gdbFullNameStr = pTarDataset.Workspace.pathName & "\" & pTarFC.FeatureDataset.name & "\" & pTarDataset.name


        newSHPPathStr = Replace(newFCPathStr & "\" & newFDSNameStr & "\", "\geodatabase\", "\shapefile\")
        newSHPPathStr = Left(newSHPPathStr, InStrRev(newSHPPathStr, "shapefile") + 9)

        '-m = move -a = add
        retval = Shell("C:\Program Files\WinZip\winzip32 -min -a -r " & Replace(newFCPathStr, ".gdb", ".zip") & " " &
Replace(newFCPathStr, newFCNameStr & ".gdb", ""))
        Debug.Print "      zipped: " & newFCPathStr
        Print #1, "      zipped: " & newFCPathStr

        'Zip SHP
        retval = Shell("C:\Program Files\WinZip\winzip32 -min -a " & newSHPPathStr & newFCNameStr & ".zip " & _
                    newSHPPathStr & newFCNameStr & ".shp" & " " & _
                    newSHPPathStr & newFCNameStr & ".shx" & " " & _
                    newSHPPathStr & newFCNameStr & ".dbf" & " " & _
                    newSHPPathStr & newFCNameStr & ".prj" & " " & _
                    newSHPPathStr & newFCNameStr & ".shp.xml")
                    'newSHPPathStr & newFCNameStr & ".sbn" & " " &
                    'newSHPPathStr & newFCNameStr & ".sbx" & " " &
        Debug.Print "      zipped: " & newSHPPathStr & newFCNameStr & ".*"
        Print #1, "      zipped: " & newSHPPathStr & newFCNameStr & ".*"
    End If


End Sub
Public Sub PopulateFeatureClass_ForExistingFeatureClass(pSrcFC As IFeatureClass, pTarFC As IFeatureClass,
pClipItem As Variant)
    On Error GoTo ErrorHandler
    Dim pSrcFeatureCursor As IFeatureCursor
    Dim pSrcFeature As IFeature
    Dim pSrcFields As IFields
    Dim pSrcField As IField
    Dim pIntGeometry As IGeometry
    Dim pTopOp As ITopologicalOperator

    'Dim pSrcDataset As IDataset
    Dim pSrcShape As IGeometry
    Dim pTarDataset As IDataset
    Dim gdbFullNameStr As String
    Dim shpfolderstr As String
    Dim pDataset As IDataset
    Dim pWS As IWorkspace
    Dim pathName As String
    Dim pTarFeatureCursor As IFeatureCursor
    Dim pTarFeatureBuffer As IFeatureBuffer
    Dim count As Long
    Dim totalCount As Long
    Dim f As Integer
    Dim tarFieldIndex As Integer

    Dim pSF As ISpatialFilter
```

```
    Dim pLeftOverSelSet As ISelectionSet

    Set pSF = Nothing

    If Not pClipItem Is Nothing Then
        If TypeOf pClipItem Is IPolygon Then
            Set pSF = New SpatialFilter
            Set pSF.Geometry = pClipItem
            Set pTopOp = New Polygon
            pSF.SpatialRel = esriSpatialRelIntersects

            Set pLeftOverSelSet = pSrcFC.Select(pSF, esriSelectionTypeHybrid, esriSelectionOptionNormal, Nothing)

            If pLeftOverSelSet.count > 0 Then
                Print #2, "       - " & pLeftOverSelSet.count & " = sde select"
            End If

            pSF.SpatialRel = esriSpatialRelContains
        End If
    End If

    Set pSrcFeatureCursor = pSrcFC.Search(pSF, True)
    Set pSrcFeature = pSrcFeatureCursor.NextFeature
    Set pSrcFields = pSrcFC.Fields

    Set pTarFeatureCursor = pTarFC.Insert(True)

    count = 0
    totalCount = 0

    Do Until pSrcFeature Is Nothing
        totalCount = totalCount + 1
        Debug.Print pSrcFeature.OID
        Set pTarFeatureBuffer = pTarFC.CreateFeatureBuffer ' makes new feature record

        Set pTarFeatureBuffer.Shape = pSrcFeature.Shape


        If Not pTarFeatureBuffer.Shape.IsEmpty Then
            For f = 0 To pSrcFields.FieldCount - 1
                Set pSrcField = pSrcFields.Field(f)
                If pSrcField.Editable And pSrcField.Type <> esriFieldTypeGeometry And UCase(Left(pSrcField.name, 8)) <>
"OBJECTID" Then
                    tarFieldIndex = pTarFC.FindField(pSrcField.name)
                    If tarFieldIndex < 0 Then
                        tarFieldIndex = pTarFC.FindField(Left(pSrcField.name, 10))
                    End If
                    If tarFieldIndex >= 0 Then
                        pTarFeatureBuffer.Value(pTarFC.FindField(pSrcField.name)) = pSrcFeature.Value(f)
                    End If
                   'Debug.Print "-- " & pSDEFCField.name
                Else
                   'Debug.Print "IGNORED " & pSDEFCField.name
                End If
            Next f
            Dim pArea As IArea
            'Set pArea = pTarFeatureBuffer.Shape
            'Debug.Print pSrcFeature.OID & " " & pArea.Area
            pTarFeatureCursor.InsertFeature pTarFeatureBuffer

            count = count + 1

            If count >= 200 Then
                pTarFeatureCursor.Flush
                count = 0
```

```
            Debug.Print "***Flush " & totalCount & " " & Now
        End If
    End If


    If Not pClipItem Is Nothing Then
        If TypeOf pClipItem Is IPolygon Then
            pLeftOverSelSet.RemoveList 1, pSrcFeature.OID
        End If
    End If

    Set pSrcFeature = pSrcFeatureCursor.NextFeature

Loop

If count > 0 Then
    pTarFeatureCursor.Flush
    count = 0
End If

If Not pLeftOverSelSet Is Nothing Then
    pLeftOverSelSet.Search Nothing, True, pSrcFeatureCursor
    Set pSrcFeature = pSrcFeatureCursor.NextFeature

    Set pTarFeatureCursor = pTarFC.Insert(True)

    Do Until pSrcFeature Is Nothing
        totalCount = totalCount + 1
        Debug.Print pSrcFeature.OID
        Set pTarFeatureBuffer = pTarFC.CreateFeatureBuffer ' makes new feature record
        If pTopOp Is Nothing Then
            Set pTarFeatureBuffer.Shape = pSrcFeature.ShapeCopy
        Else
            Dim starttime As Date
            Set pSrcShape = pSrcFeature.ShapeCopy 'qi
            Set pTopOp = pSrcShape
            'Debug.Print "   " & pSrcFeature.OID
            pTopOp.Simplify
            Set pIntGeometry = pTopOp.Intersect(pClipItem, pSrcShape.Dimension)
            If Not pIntGeometry Is Nothing Then
                If pIntGeometry.GeometryType = pTarFC.ShapeType Then
                    Set pTarFeatureBuffer.Shape = pIntGeometry
                End If
            End If
        End If

        If Not pTarFeatureBuffer.Shape.IsEmpty Then
            For f = 0 To pSrcFields.FieldCount - 1
                Set pSrcField = pSrcFields.Field(f)
                If pSrcField.Editable And pSrcField.Type <> esriFieldTypeGeometry And UCase(Left(pSrcField.name, 8)) <>
"OBJECTID" Then
                    tarFieldIndex = pTarFC.FindField(pSrcField.name)
                    If tarFieldIndex < 0 Then
                        tarFieldIndex = pTarFC.FindField(Left(pSrcField.name, 10))
                    End If
                    If tarFieldIndex >= 0 Then
                        pTarFeatureBuffer.Value(pTarFC.FindField(pSrcField.name)) = pSrcFeature.Value(f)
                    End If
                Else
                   'Debug.Print "IGNORED " & pSDEFCField.name
                End If
            Next f

            pTarFeatureCursor.InsertFeature pTarFeatureBuffer
```

```
            count = count + 1

            If count >= 200 Then
                pTarFeatureCursor.Flush
                count = 0
                'Debug.Print "***Flush " & totalCount & " " & Now
            End If
        End If
        Set pSrcFeature = pSrcFeatureCursor.NextFeature

    Loop

    If count > 0 Then
        pTarFeatureCursor.Flush
        count = 0
    End If

End If

If totalCount = 0 Then
    Set pDataset = pTarFC
    Set pWS = pDataset.Workspace
    pathName = pWS.pathName
    m_deleteColl.Add pathName
Else

    Dim shapeTypeStr As String

    If pTarFC.ShapeType = esriGeometryPolyline Then
        shapeTypeStr = "POLYLINE"
    ElseIf pTarFC.ShapeType = esriGeometryPolygon Then
        shapeTypeStr = "POLYGON"
    ElseIf pTarFC.ShapeType = esriGeometryPoint Then
        shapeTypeStr = "POINT"
    ElseIf pTarFC.ShapeType = esriGeometryMultipoint Then
        shapeTypeStr = "MULTIPOINT"
    End If

    'export to shapefile here
    Set pTarDataset = pTarFC 'qi
    Print #2, "       - " & pTarFC.featureCount(Nothing) & " = fbp"
    Print #2, "       - " & pTarDataset.Workspace.pathName & " " & pSrcFC.featureCount(Nothing)
    Print #2, "   ----------------------------------------"

    gdbFullNameStr = pTarDataset.Workspace.pathName & "\" & pTarFC.FeatureDataset.name & "\" & pTarDataset.name

    shpfolderstr = Replace(pTarDataset.Workspace.pathName & "\" & pTarFC.FeatureDataset.name & "\", "\geodatabase\",
"\shapefile\")
    shpfolderstr = Left(shpfolderstr, InStrRev(shpfolderstr, "shapefile") + 9)
    'Debug.Print shpfolderstr
    m_GP.RefreshCatalog pTarDataset.Workspace.pathName
    'm_GP.FeatureClassToShapefile_conversion gdbFullNameStr, shpfolderstr
    m_GP.CreateFeatureClass_Management shpfolderstr, pTarDataset.name & ".shp", shapeTypeStr, gdbFullNameStr,
"DISABLED", "DISABLED",
"""PROJCS['NAD_1983_UTM_Zone_12N',GEOGCS['GCS_North_American_1983',DATUM['D_North_American_1983',S
PHEROID['GRS_1980',6378137.0,298.257222101]],PRIMEM['Greenwich',0.0],UNIT['Degree',0.0174532925199433]],PR
OJECTION['Transverse_Mercator'],PARAMETER['False_Easting',500000.0],PARAMETER['False_Northing',0.0],PARAM
ETER['Central_Meridian',-
111.0],PARAMETER['Scale_Factor',0.9996],PARAMETER['Latitude_Of_Origin',0.0],UNIT['Meter',1.0]];-5120900 -
9998100 100;0 1;0 1;0.1;0.1;0.1;IsHighPrecision""", "#", "0", "0", "0"

    'Open shapefile where new features will be written to
    'For simplicity, sample does not contain code to create a new shapefile

    Dim newShpFC As IFeatureClass
```

```
        Dim pInsertFeatureCursor As IFeatureCursor
        Dim pInsertFeatureBuffer As IFeatureBuffer
        Dim pFGDBFeatureCursor As IFeatureCursor
        Dim pFGDBFeature As IFeature
        Dim shpFeatureCount As Long
        Dim shpMetadata As IMetadata
        Dim xmlPropSet As IPropertySet

        Set newShpFC = OpenShapefileFeatureClass(shpfolderstr, pTarDataset.name)

        If Not newShpFC Is Nothing Then

            Set shpMetadata = newShpFC
            Set xmlPropSet = m_sourceMetadata.Metadata
            shpMetadata.Metadata = xmlPropSet

            Set pInsertFeatureCursor = newShpFC.Insert(True)
            Set pInsertFeatureBuffer = newShpFC.CreateFeatureBuffer
            'Loop through all the features in InFeatureClass
            Set pFGDBFeatureCursor = pTarFC.Search(Nothing, True)
            Set pFGDBFeature = pFGDBFeatureCursor.NextFeature
            Do While Not pFGDBFeature Is Nothing
                'Add the original feature's geometry to the feature buffer
                Set pInsertFeatureBuffer.Shape = pFGDBFeature.Shape
                'Add all the original feature's fields to the feature buffer
                PopulateShapeFileFeatureAttributes pInsertFeatureBuffer, pFGDBFeature
                'Insert the feature into the cursor
                pInsertFeatureCursor.InsertFeature pInsertFeatureBuffer
                shpFeatureCount = shpFeatureCount + 1

                If shpFeatureCount = 1000 Then
                  pInsertFeatureCursor.Flush
                  Debug.Print "flush " & pFGDBFeature.OID
                  shpFeatureCount = 0
                End If
                Set pFGDBFeature = pFGDBFeatureCursor.NextFeature
            Loop
            pInsertFeatureCursor.Flush 'Flush the cursor one last time

        End If

    End If

    Set pSrcFeatureCursor = Nothing
    Set pSrcFeature = Nothing
    Set pSrcFields = Nothing
    Set pSrcField = Nothing
    Set pIntGeometry = Nothing
    Set pSF = Nothing
    Set pTopOp = Nothing
    Set pSF = Nothing
    Set pSrcShape = Nothing
    'Set pTarDataset = Nothing
    'Set pDataset = Nothing
    Set pWS = Nothing
    Set pTarFeatureCursor = Nothing
    Set pTarFeatureBuffer = Nothing
    Exit Sub
ErrorHandler:
    'Debug.Print "ERROR: Src OID = " & pSrcFeature.OID & " #" & Err.Description
    Resume Next


End Sub
```

```
Public Function calcSpatialIndex(inFC As IFeatureClass) As Long
   Dim pFCursor As IFeatureCursor
   Dim pFeature As IFeature
   Dim dblTotal As Double
   Dim iFCount As Long

   Set pFCursor = inFC.Search(Nothing, False)
   Set pFeature = pFCursor.NextFeature
   dblTotal = 0
   iFCount = 1

   Do Until pFeature Is Nothing
      If Not pFeature.Shape.IsEmpty Then
         dblTotal = dblTotal + ((pFeature.Extent.Height + pFeature.Extent.Width) / 2)
         'Set pFeature = pFCursor.NextFeature
         iFCount = iFCount + 1
      End If
      Set pFeature = pFCursor.NextFeature
   Loop

   calcSpatialIndex = CLng(((dblTotal / iFCount) * 3) / 1000) * 1000


End Function

Public Sub makeDownloadTables()

   'LOAD DATA LOCAL INFILE "C:/Documents and Settings/Dennis/Desktop/menagerie/pet.txt" INTO TABLE PET;

   Open "C:\temp\sgidtest\sgidcategory.txt" For Output As #1
   Open "C:\temp\sgidtest\sgidlayer.txt" For Output As #2
   Open "C:\temp\sgidtest\sgidproduct.txt" For Output As #3
   Open "C:\temp\sgidtest\sgidextent.txt" For Output As #4


   Dim pCofcursor As IFeatureCursor
   Dim pcofeature As IFeature
   Dim countyName As String
   Dim ftpPath As String
   ftpPath = "J:\ftp\pub"
   'ftpPath = "ftp://ftp.agrc.state.ut.us"
   Call openSDEWS
   Set m_CoClipFC = m_pSDEFWS.OpenFeatureClass("SGID.U024.CountyBoundaries")

   Dim extentArray(29) As String
   extentArray(0) = "statewide"

   Dim pTablesort As ITableSort
   Set pTablesort = New TableSort
   Set pTablesort.Table = m_CoClipFC
   pTablesort.Fields = "NAME"

   pTablesort.Sort Nothing


   Set pCofcursor = pTablesort.Rows
   Set pcofeature = pCofcursor.NextFeature
   Dim extentCount As Integer
   extentCount = 1

   Do Until pcofeature Is Nothing
         countyName = UCase(Replace(pcofeature.Value(m_CoClipFC.FindField("NAME")), " ", ""))
         extentArray(extentCount) = countyName
         extentCount = extentCount + 1
         Set pcofeature = pCofcursor.NextFeature
```

```
Loop

Dim catID As Long
Dim layID As Long
Dim prodID As Long
Dim extID As Long
'Print #1, "CATID" & Chr(9) & "CATFDS" & Chr(9) & "CATNAME"
'Print #2, "LAYID" & Chr(9) & "CATID" & Chr(9) & "LAYNAME" & Chr(9) & "LAYSCALE"
'Print #3, "PRODID" & Chr(9) & "LAYID" & Chr(9) & "EXTID" & Chr(9) & "PRODTYPE" & Chr(9) & "PRODSIZE"
'Print #4, "EXTID" & Chr(9) & "EXTNAME"
For extID = 0 To 29
   Print #4, CInt(extID) & Chr(9) & extentArray(extID)
Next extID


Dim pGxApp As IGxApplication
Dim pSelGxObject As IGxObject
Dim pGxSel As IGxSelection
Dim pEnumGxObjSel As IEnumGxObject

Dim catName As String
Dim scaleName As String
Dim layName As String

Set pGxApp = Application
Set pGxSel = pGxApp.Selection
Set pEnumGxObjSel = pGxSel.SelectedObjects
pEnumGxObjSel.Reset
Set pSelGxObject = pEnumGxObjSel.Next
Dim fccnt As Long
Do Until pSelGxObject Is Nothing
   If pSelGxObject.Category = "SDE Feature Class" Then
      'fccnt = fccnt + 1
      'Debug.Print "  .. " & pSelGxObject.BaseName & " " & fccnt
      'Print #2, "  .. " & pSelGxObject.BaseName & " " & fccnt
   ElseIf pSelGxObject.Category = "SDE Feature Dataset" Then
      catName = Replace(Right(pSelGxObject.BaseName, _
            Len(pSelGxObject.BaseName) - InStrRev(pSelGxObject.BaseName, ".")), "Archive", "")
      Debug.Print catName
      Print #1, CStr(catID) & Chr(9) & pSelGxObject.BaseName & Chr(9) & catName '(FDS)
      Dim pGxObjContainer As IGxObjectContainer
      Dim pEnumGxObj As IEnumGxObject
      Dim pCurrGxObject As IGxObject
      Dim fileSize As String
      Set pGxObjContainer = pSelGxObject
      If pGxObjContainer.HasChildren Then
         Set pEnumGxObj = pGxObjContainer.Children
         If Not pEnumGxObj Is Nothing Then
            Set pCurrGxObject = pEnumGxObj.Next
            Do Until pCurrGxObject Is Nothing
               If pCurrGxObject.Category = "SDE Feature Class" Then
                  Debug.Print ".." & pCurrGxObject.BaseName
                  Print #2, CStr(layID) & Chr(9) & CStr(catID) & Chr(9) & Right(pCurrGxObject.BaseName, _
                        Len(pCurrGxObject.BaseName) - InStrRev(pCurrGxObject.BaseName, ".")) & _
                        Chr(9) & pCurrGxObject.BaseName & Chr(9) & Mid(pCurrGxObject.BaseName, 6, 4)
                  For extID = 0 To 29
                     If extID = 0 Then

                        'SHP
                        fileSize = checkFileSize(ftpPath & "/SGID_Vector/" & catName & "/" & Replace(pCurrGxObject.BaseName
                           & "/" & extentArray(extID) & "/shapefile/" & Replace(pCurrGxObject.BaseName, ".", "_") & ".zip")
                        If fileSize <> "" Then
                           Print #3, CStr(prodID) & Chr(9) & CStr(layID) & Chr(9) & CStr(extID) & Chr(9) & "SHP" & Chr(9) & fileSi
                           prodID = prodID + 1
                        End If
```

```
                    'GDB
                    fileSize = checkFileSize(ftpPath & "/SGID_Vector/" & catName & "/" & Replace(pCurrGxObject.BaseName
                         & "/" & extentArray(extID) & "/geodatabase/" & Replace(pCurrGxObject.BaseName, ".", "_") & ".zip")
                    If fileSize <> "" Then
                       Print #3, CStr(prodID) & Chr(9) & CStr(layID) & Chr(9) & CStr(extID) & Chr(9) & "GDB" & Chr(9) & fileS
                       prodID = prodID + 1
                    End If

                 ElseIf extID > 0 And extID < 30 Then

                    'SHP
                    fileSize = checkFileSize(ftpPath & "/SGID_Vector/" & catName & "/" & Replace(pCurrGxObject.BaseName
                      & "/county/" & extentArray(extID) & "/shapefile/" & Replace(pCurrGxObject.BaseName, ".", "_") & ".zip")
                    If fileSize <> "" Then
                       Print #3, CStr(prodID) & Chr(9) & CStr(layID) & Chr(9) & CStr(extID) & Chr(9) & "SHP" & Chr(9) & fileSi
                       prodID = prodID + 1
                    End If

                    'GDB
                    fileSize = checkFileSize(ftpPath & "/SGID_Vector/" & catName & "/" & Replace(pCurrGxObject.BaseName
                      & "/county/" & extentArray(extID) & "/geodatabase/" & Replace(pCurrGxObject.BaseName, ".", "_") & ".zi
                    If fileSize <> "" Then
                       Print #3, CStr(prodID) & Chr(9) & CStr(layID) & Chr(9) & CStr(extID) & Chr(9) & "GDB" & Chr(9) & fileS
                       prodID = prodID + 1
                    End If

                 End If
              Next extID

           End If
           layID = layID + 1
           Set pCurrGxObject = pEnumGxObj.Next
        Loop
     End If
  End If
  catID = catID + 1
End If
Set pSelGxObject = pEnumGxObjSel.Next
Loop
Close #1
Close #2
Close #3
Close #4
End Sub

Public Function checkFileSize(inFilePath As String) As String

    Dim fso
    Dim f
    Dim fileSize As Long

    'Debug.Print checkFileSize

    Set fso = CreateObject("Scripting.FileSystemObject")

    If fso.fileexists(inFilePath) Then
       Set f = fso.GetFile(inFilePath)

       fileSize = f.size / 1000
       If (fileSize < 1) Then
          checkFileSize = "<1K"
       End If
       If (fileSize < 1000) Then
         checkFileSize = CStr(CInt(fileSize)) + "K"
```

```
        End If
        If ((fileSize >= 1000) And (fileSize < 1000000)) Then
          checkFileSize = CStr(CLng(fileSize / 1000)) + "MB"
        End If
        If (fileSize >= 1000000) Then
          checkFileSize = CStr(CLng(fileSize / 1000000)) + "GB"
        End If
      End If

End Function



Public Function OpenShapefileFeatureClass(strWorkspace As String, strFeatureClass As String) As IFeatureClass
  On Error GoTo ErrorHandler
  Dim pShpWorkspaceName As IWorkspaceName
  Dim pDatasetName As IDatasetName
  Dim pName As IName

  'Create the workspace name object
  Set pShpWorkspaceName = New WorkspaceName
  pShpWorkspaceName.pathName = strWorkspace
  pShpWorkspaceName.WorkspaceFactoryProgID = "esriDataSourcesFile.shapefileworkspacefactory.1"

  'Create the feature class name object
  Set pDatasetName = New FeatureClassName
  pDatasetName.name = strFeatureClass
  Set pDatasetName.WorkspaceName = pShpWorkspaceName

  'Open the feature class
  Set pName = pDatasetName
  Set OpenShapefileFeatureClass = pName.Open

  Exit Function

ErrorHandler:
  Set OpenShapefileFeatureClass = Nothing
End Function



Private Sub PopulateShapeFileFeatureAttributes(pFeatureBuffer As IFeatureBuffer, pFeature As IFeature)
  Dim pRowBuffer As IRowBuffer
  Dim pNewFields As IFields 'fields on target feature class
  Dim pNewField As IField
  Dim pFields As IFields 'fields on original feature class
  Dim pField As IField
  Dim FieldCount As Integer
  Dim NewFieldIndex As Long

  'Copy the attributes of the orig feature the new feature
  Set pRowBuffer = pFeatureBuffer
  Set pNewFields = pRowBuffer.Fields

  Set pFields = pFeature.Fields
  For FieldCount = 0 To pFields.FieldCount - 1
    Set pField = pFields.Field(FieldCount)
    If pField.Type = esriFieldTypeGUID Or pField.Type = esriFieldTypeGlobalID Then
        If Not IsNull(pFeature.Value(FieldCount)) Then
            pFeatureBuffer.Value(pFeatureBuffer.Fields.FindField(Left(pField.name, 10))) = pFeature.Value(FieldCount)
        Else
            pFeatureBuffer.Value(pFeatureBuffer.Fields.FindField(Left(pField.name, 10))) = ""
        End If
    ElseIf pField.name = "SHAPE_Length" Then
        Dim pTempCurve As ICurve
```

```
        If Not IsNull(pFeature.Value(FieldCount)) Then
            Set pTempCurve = pFeature.Shape
            pFeatureBuffer.Value(pFeatureBuffer.Fields.FindField("SHAPE_Leng")) = pTempCurve.Length
        Else
            pFeatureBuffer.Value(pFeatureBuffer.Fields.FindField("SHAPE_Leng")) = 0
        End If
    ElseIf pField.name = "SHAPE_Area" Then
        Dim pTempArea As IArea
        If Not IsNull(pFeature.Value(FieldCount)) Then
            Set pTempArea = pFeature.Shape
            pFeatureBuffer.Value(pFeatureBuffer.Fields.FindField("SHAPE_Area")) = pTempArea.Area
        Else
            pFeatureBuffer.Value(pFeatureBuffer.Fields.FindField("SHAPE_Area")) = 0
        End If
    ElseIf Not pField.Type = esriFieldTypeGeometry And Not pField.Type = esriFieldTypeOID _
      And pField.Editable Then
        'NewFieldIndex = pNewFields.Field(FieldCount)
        'If Not NewFieldIndex = -1 Then
        If Not IsNull(pFeature.Value(FieldCount)) Then
            pFeatureBuffer.Value(pFeatureBuffer.Fields.FindField(Left(pField.name, 10))) = CStr(pFeature.Value(FieldCount))
        Else
            If pFeatureBuffer.Fields.Field(FieldCount).Type = esriFieldTypeDate Then
                pFeatureBuffer.Value(pFeatureBuffer.Fields.FindField(Left(pField.name, 10))) = CDate("Jan 1, 100")
            ElseIf pFeatureBuffer.Fields.Field(FieldCount).Type = esriFieldTypeDouble Or _
                  pFeatureBuffer.Fields.Field(FieldCount).Type = esriFieldTypeInteger Or _
                  pFeatureBuffer.Fields.Field(FieldCount).Type = esriFieldTypeSingle Or _
                  pFeatureBuffer.Fields.Field(FieldCount).Type = esriFieldTypeSmallInteger Then
                pFeatureBuffer.Value(pFeatureBuffer.Fields.FindField(Left(pField.name, 10))) = 0
            Else
                pFeatureBuffer.Value(pFeatureBuffer.Fields.FindField(Left(pField.name, 10))) = ""
            End If
        End If
        'Debug.Print FieldCount & pFeature.Fields.Field(FieldCount).name & " " & _
pFeatureBuffer.Value(pfeaturebuffer.fields.findfield(left(pfield.name),10))
        'End If
    End If
  Next FieldCount
End Sub
```