TerraGo Publisher for ArcGIS 6.0.4

Software Developer’s Guide

Introduction

TerraGo Publisher for ArcGIS is a set of ArcObjects® COM components that enable creation and viewing of GeoPDF® files in ArcObjects applications. With version 6.0, advanced GeoPDF layering capabilities have been introduced. Advanced layering gives the user complete control over the GeoPDF layers produced by a GeoPDF export. This document provides the information needed by a developer to support or utilize the TerraGo Publisher for ArcGIS product from a third-party product or to automate the use of the TerraGo Publisher for ArcGIS product. Understanding and use of this document requires knowledge of Environmental Systems Research Institute (ESRI®) ArcObjects and general Microsoft® Windows® software development.

Document Conventions

COM interface specifications are given in abbreviated Visual C++ syntax, as that is the implementation language, but these interfaces are also accessible from VBA, VB.NET and C#, though the semantics and native data types differ. In particular C++ COM methods whose names are prefixed with "get_", "put_" or "putref_" will be implemented as properties and lose that prefix in other languages. In that case the property is typically accessed via the equal operator – '='. This is illustrated in the given code samples, which are intended for use in the ArcMap VBA editor. Note that the "MAP2PDF for ArcGIS Export", "MAP2PDF for ArcGIS View" and "MAP2PDF for GeoMark" type libraries may need to be added from the VBA editor's Tools->Reference command before they will compile and run.

Product

The purpose of the TerraGo Publisher for ArcGIS product is to enable export of GIS data to GeoPDF file format, viewing of GeoPDF files from within ArcMap®, and import of GeoMarks. GeoPDF file format offers georegistration, layering, feature attributes and user markup in the widely supported GeoPDF file format. Freely available TerraGo Technologies plug-ins for Adobe® Acrobat® and Adobe Reader® enable recognition of GeoPDF georegistration. TerraGo Technologies Mobile and Composer products create and export GeoMarks. The TerraGo Publisher for ArcGIS product includes three ArcMap extensions, GeoPDF Export, GeoPDF View and GeoMark Import.

GeoPDF Export Extension

The GeoPDF Export extension provides an ArcMap user interface and the ExportGeoPDF coclass, which implements an ArcObjects exporter. In addition to standard ArcObjects COM interfaces it provides some new COM interfaces, IExportGeoPDF, IExportGeoPDF2, IGeoPDFConfig, IGeoPDFConfig2, IGeoPDFConfig3, IGeoPDFConfig4 ,IGeoPDFConfig5, IJpegConfig, IGeoPDFLayeringConfig,IGeoPDFLayeringConfig2 and IGeoPDFLayeringConfig3. IGeoPDFconfig5 and IGeoPDFLayeringConfig3 are new with the 6.0 release. The The IExportGeoPDF interface offers the same properties and methods as the ESRI IExportPDF interface, but not all operate in the same way as those of ExportPDF. See the next section for an explanation. The IExportGeoPDF2 interface is an enhancement to IExportGeoPDF that provides for a special need when ExportGeoPDF is instantiated by callers other than ArcMap. See the next section for an explanation. The IGeoPDFConfig, IGeoPDFConfig2, IGeoPDFConfig3, IGeoPDFConfig4, IGeoPDFConfig5, IJpegConfig, IGeoPDFLayeringConfig,IGeoPDFLayeringConfig2 and IGeoPDFLayeringConfig3 interfaces are used to configure settings specific to ExportGeoPDF.

GeoPDF Export Is Different

Since the 3.0 release, all rendering is performed within the IExport.StartExporting() call. It is not necessary and not recommended to make the usual call to IActiveView.Output() between a call to StartExporting() and a call to IExport.FinishExporting(). Any drawing performed to the device context returned by IExport.StartExporting() will be discarded when IExport.FinishExporting() is called.

The IExportGeoPDF2.ActiveView property is provided so that applications can supply ExportGeoPDF with the view they wish to export. Map and PageLayout IActiveView interfaces are supported. It is recommended that this property be set as soon as the ExportGeoPDF class has been cocreated. Callers should not call IActiveView.Output() or other drawing methods between the calls to IExport.StartExporting() and IExport.FinishExporting().

GeoPDF Export Usage

In general the standard exporter COM interfaces presented by ExportGeoPDF operate in the same manner as those of ArcObject's ExportPDF coclass; however, there are some differences: the IExportPDF.ImageCompression property has no effect on raster layers as with this product release layers that implement IRasterLayer are tiled and JPEG-compressed in the GeoPDF, and the IExport.StepProgressor and IExport.TrackCancel properties are not implemented. See the IJpegConfig interface for control of JPEG compression.

GeoPDF layering can be globally enabled or disabled. See the IGeoPDFConfig.Layering property. If enabled the GeoPDF layer scheme will be largely the same as the ArcMap layer scheme, except for some additional groups and layers. Map grids and map surround elements are added to the GeoPDF layer scheme. A GeoPDF groups is created for each ArcMap data frame. The IGeoPDFLayeringConfig interfaces enable specifying a custom GeoPDF layer scheme. The IGeoPDFLayeringConfig2 interface supports selecting legacy layering, which overrides the custom GeoPDF layer scheme with an automatically generated legacy layering scheme. The legacy layering scheme emulates the automatically generated layer scheme of the 5.2.1 release. Legacy layering is selected by default. The IGeoPDFLayeringConfig3 interface enables filtering the GeoPDF layers. The 'Ignore Disabled TOC Layers' option in the product graphical user interface is implemented by filtering for ArcMap layers that are off, out of scale, invalid or missing.

To create a custom GeoPDF layer scheme a hierarchy of GeoPDFMenu, GeoPDFGroup and GeoPDFLayer objects is created, ArcObjects objects are assigned to the GeoPDFLayer objects, and the root GeoPDFMenu, GeoPDFGroup and GeoPDFLayer objects passed to the IGeoPDFLayeringConfig interface. Many ArcObjects object types are supported, including map layers, map grids, map annotation groups, map locators and page elements.

GIS feature attributes can be exported to GeoPDF through two mechanisms, traditional GeoPDF annotations which are compatible with Adobe Acrobat / Adobe Reader version 6 and earlier, and object data which requires Acrobat/Reader version 7 or later. Traditional GeoPDF annotation creation is not currently supported by the product. The IGeoPDFConfig.Compatibility property, which selects the annotation mechanism, no longer has any effect. Attributes are turned off or on layer-by-layer. See the IGeoPDFConfig.GetAnnotate() and IGeoPDFConfig.PutAnnotate() methods. Currently only feature attributes of layers that implement IGeoFeatureLayer may be exported; however, presence of the IGeoFeatureLayer interface is not a guarantee of support. The fields to be displayed in the GeoPDF object data and the title displayed for each field are controlled via the ArcObjects IFieldInfo interface. See ESRI ArcObjects documentation to learn about that interface. The total number of features attributed has a significant impact on Acrobat/Reader display performance.

Most common map datums are supported by the TerraGo, Acrobat and Reader plug-ins. Datums that are not directly supported can be supported via a WGS84 shift. A WGS84 shift supplies the x, y, and z translations, in meters, needed to transform the map's datum to the WGS 1984 datum. The IGeoPDFConfig.GetWGS84Shift() and IGeoPDFConfig.PutWGS84Shift() methods retrieve and set a map's WGS4 shift.

Export of feature hyperlinks is supported. Note that the hyperlinks must be created with ArcMap or other software. Both field-based and dynamic hyperlinks are supported by ExportGeoPDF. The IGeoPDFConfig2.GetHyperlink() and IGeoPDFConfig2.PutHyperlink() methods are provided to get and set the hyperlink export status of individual layers.

A zip archive of the output GeoPDF and all documents it is hyperlinked to can be created. All accessible documents are copied into the zip archive and the GeoPDF hyperlink targets redirected to the copy. Any inaccessible documents are logged to a file in the same path as the zip archive. The IGeoPDFConfig2.Archive property controls hyperlinked document archiving. The new IGeoPDFConfig5 interface adds the ability to embed hyperlinked documents into the exported GeoPDF file.

The IGeoPDFConfig2.LayeredLabels property controls if feature labels are placed in their own GeoPDF layer, which is then grouped with the associated GeoPDF feature layer, or merged into the GeoPDF feature layer. This is a global property.

The IGeoPDFConfig2.MapVisibleScale property controls if ExportGeoPDF translates ArcMap layer visible scale ranges to GeoPDF zoom control of layer visibility. If this is disabled, ArcMap layers that are not currently visible due to the map scale being outside of their visible scale range will not be exported to the GeoPDF. If it is enabled, they will be exported, but displayed only at GeoPDF zoom levels that are equivalent to the ArcMap visible scale range. For example, if the map is at 1:100000 scale and the layer visible scale range is set to 1:75000 to 1:50000, it will be visible in the GeoPDF at zoom levels from 133% to 200%. This is a global property.

The IGeoPDFConfig3 interface is provided to allow for exporting attributes from layers that do not implement IGeoFeatureLayer.

The IGeoPDFConfig4.OptimizeFile property controls compression of the GeoPDF. It implements GeoPDF compressed object stream support. Creating compressed object streams can substantially reduce the size of the GeoPDF when object data or hyperlinks are present; however, some applications may not be able to successfully read the resulting GeoPDF, particularly the TerraGo Mobile product.

The IGeoPDFConfig4.VectorResolution property controls the effective resolution for rendering vector graphics. Higher resolutions can enhance the graphic accuracy of the GeoPDF when zoomed in, but can also increase file size. This property does not affect the resolution of GeoPDF imagery. This is a global property.

The IGeoPDFConfig4.ToolbarCompatibility property controls the mapframe styles that are created in the GeoPDF. When `geoPDF_OGC_ONLY` is selected, the mapframes created will conform to the original TerraGo GeoPDF mapframe style, now the OGC Best Practice GeoPDF 2.2 OGC 08-139r1. If the coordinate system cannot be supported with that mapframe style, no mapframe will be generated. If `geoPDF_OGC_PREFERRED` is selected OGC mapframe style is used if possible, otherwise the Adobe mapframe style as described in their ISO 32000 submission is created. Selecting `geoPDF_ISO_ONLY` will always produce ISO mapframe style. This is a global property.

The IJpegConfig interface is used to control export of imagery and web map services to GeoPDF. ExportGeoPDF tiles imagery and when appropriate uses JPEG image compression. When a limited number of colors are present, an image may be converted to palette-color and Flate-compressed. This is a global property.

GeoPDF Export Error Reporting

ExportGeoPDF writes an error log to the users' Application Data folder, in the "TerraGoTechnologies/Map2PDF for ArcGIS" folder. The location of the Application Data folder varies with the version of Windows.

GeoPDF Export COM Interfaces

The ExportGeoPDF coclass implements the following interfaces that can be found in ESRI ArcObjects documentation: IExport, IExportVector, IExportVectorOptions, IExportVectorOptionsEx, IExportColorspaceSettings, IOutputRasterSettings and IClassID.

ExportGeoPDF varies from the behavior of ExportPDF: The value of the

IOutputRasterSettings.ResampleRatio is forced to 1. IExportColorspaceSettings does not affect the colorspace of images, which is controlled througn the IJpegConfig interface.

The ExportGeoPDF coclass implements the following interfaces that are specific to this coclass: IExportGeoPDF, IExportGeoPDF2, IGeoPDFConfig, IGeoPDFConfig2, IGeoPDFConfig3, IGeoPDFConfig4, IGeoPDFConfig5,IJpegConfig, IGeoPDFLayeringConfig, IGeoPDFLayeringConfig2 and IGeoPDFLayeringConfig3. These are given below.

IExportGeoPDF:

The same properties and methods as the IExportPDF interface.

The IExportGeoPDF.ImageCompression property has no effect as all imagery is JPEG-compressed and tiled. See the IJpegConfig interface for control of JPEG compression.

IExportGeoPDF2:

```
HRESULT putref_ActiveView( IActiveView* ActiveView )
```

IGeoPDFConfig:

Globally enable or disable GeoPDF layering.

```
HRESULT get_Layering( VARIANT_BOOL* layering )

HRESULT put_Layering( VARIANT_BOOL layering )
```

Globally get/put the feature attribute export type (`geoPDFAnnotTypeAcrobat6` or `geoPDFAnnotTypeObjectData`). Currently has no effect on output.

```
HRESULT get_Compatibility( geoPDFAnnotType* compatibility )

HRESULT put_Compatibility( geoPDFAnnotType compatibility )
```

For a layer, get/put its feature attribute export status.

```
HRESULT GetAnnotate( IGeoFeatureLayer* layer,
                     VARIANT_BOOL* annotate )

HRESULT PutAnnotate( IGeoFeatureLayer* layer,
                     VARIANT_BOOL annotate )
```

Manage a map datum's WGS84 shift.

```
HRESULT IsWGS84ShiftPresent( IMap* map,
                             VARIANT_BOOL* shifted )

HRESULT GetWGS84Shift( IMap* map,
                       BSTR* name,
                       double* dX,
                       double* dY,
                       double* dZ )

HRESULT PutWGS84Shift( IMap* map,
                       BSTR name,
                       double dX,
                       double dY,
                       double dZ )

HRESULT ClearWGS84Shift( IMap* map )
```

A COM error will be generated if IGeoPDFConfig.GetWGS84Shift() or IGeoPDFConfig.ClearWGS84Shift() is called when no WGS84Shift is present. Call IGeoPDFConfig.IsWGS84Shift() first.

IGeoPDFConfig2:

For a layer, get/put its hyperlink export state.

```
HRESULT GetHyperlink( IGeoFeatureLayer*  layer,
                      VARIANT_BOOL* hyperlink )

HRESULT PutHyperlinks( IGeoFeatureLayer*  layer,
                       VARIANT_BOOL hyperlink )
```

For technical reasons the resolution of a hyperlink target may not be the same in non-ArcMap environments, i.e. standalone applications, Server and Engine. The hyperlink base property used by ArcMap and ExportGeoPDF is not available in other execution environments. In the ArcMap environment relative document hyperlink targets are resolved relative to the ArcMap document. These may not export correctly in other execution environments.

Globally get/put the hyperlinked document archive creation state.

```
HRESULT get_Archive( VARIANT_BOOL* archive )

HRESULT put_Archive( VARIANT_BOOL archive )
```

Globally get/put the label layering state.

```
HRESULT get_LayeredLabels( VARIANT_BOOL* layeredLabels )

HRESULT put_LayeredLabels( VARIANT_BOOL layeredLabels )
```

Globally get/put the visible scale to zoom level mapping state.

```
HRESULT get_MapVisibleScale( VARIANT_BOOL* mapVisibleScale )

HRESULT put_MapVisibleScale( VARIANT_BOOL mapVisibleScale )
```

IGeoPDFConfig3:

Get/put the GeoPDF annotation state for an ArcMap layer.

```
HRESULT GetAnnotate( IFeatureLayer* layer,
                     VARIANT_BOOL* annotate )

HRESULT PutAnnotate( IFeatureLayer* layer,
                     VARIANT_BOOL annotate )
```

Get/put the GeoPDF hyperlink state for an ArcMap layer.

```
HRESULT GetHyperlink( IFeatureLayer* layer,
                      VARIANT_BOOL* hyperlink )

HRESULT PutHyperlink( IFeatureLayer* layer,
                      VARIANT_BOOL hyperlink )
```

IGeoPDFConfig4:

Globally get/put the file size optimization state.

```
HRESULT OptimizeFile( VARIANT_BOOL* OptimizeFile )

HRESULT OptimizeFile( VARIANT_BOOL OptimizeFile )
```

Globally get.put the vector resolution in dots per inch (6400 is the maximum recommended)

```
HRESULT VectorResolution( double* VectorResolution )

HRESULT VectorResolution( double VectorResolution )
```

Globally get/put the GeoPDF Toolbar compatibility state. Supported values are `geoPDF_OGC_ONLY`, `geoPDF_OGC_PREFERRED` and `geoPDF_ISO_ONLY`.

```
HRESULT ToolbarCompatibility( geoPDFToolbarCompatType* compatibility )

HRESULT ToolbarCompatibility( geoPDFToolbarCompatType compatibility )
```

IGeoPDFConfig5:

Options for zip archive creation and document embedding are mutually exclusive. Enabling one automatically disables the other.

Globally get/put the hyperlinked document zip archive creation state.

```
HRESULT get_Archive( VARIANT_BOOL* archive )

HRESULT put_Archive( VARIANT_BOOL archive )
```

Globally get/put the hyperlinked document embedding state.

```
HRESULT get_Embed( VARIANT_BOOL* embed )

HRESULT put_Embed( VARIANT_BOOL embed )
```

Globally get/put the TerraGo enabling state.

```
HRESULT get_Enabled( VARIANT_BOOL* enabled )

HRESULT put_Enabled( VARIANT_BOOL enabled )
```

IJpegConfig:

Globally get/put output JPEG compression image quality (1-100).

```
HRESULT get_JpegQuality( short* Quality )

HRESULT put_JpegQuality( short Quality )
```

Globally get/put output JPEG image type.

```
HRESULT get_JpegImageType( esriExportImageType* ImageType )

HRESULT put_JpegImageType( esriExportImageType ImageType )
```

IGeoPDFLayeringConfig:

Get the number of nodes at the root of the layer tree.

```
HRESULT get_LayerCount( long* count )
```

Get a node at the root of the layer tree.

```
HRESULT get_Layer( long index, IGeoPDFLayerNode**
                   ppNode )
```

Insert a node at the root of the layer tree.

```
HRESULT LayerInsertNode( long index, IGeoPDFLayerNode*
                         pNode )
```

Remove a node from the root of the layer tree.

```
HRESULT LayerRemoveNode( long index )
```

Remove all nodes from the layer tree.

```
HRESULT LayerRemoveAll();
```

The application state methods that follow are not used in this release.

Get the number of states.

```
HRESULT get_StateCount( long* count )
```

Get a state.

```
HRESULT get_State( long index, IGeoPDFApplicationState**
                   state )
```

Insert a state.

```
HRESULT StateAdd( IGeoPDFApplicationState* state )
```

Remove a state.

```
HRESULT StateRemove( IGeoPDFApplicationState* state )
```

Remove all states.

```
HRESULT StateRemoveAll()
```

IGeoPDFLayeringConfig2:

Get the legacy layering state.

```
HRESULT get_LegacyLayering( VARIANT_BOOL* legacy )
```

Set the legacy layering state.

```
HRESULT put_LegacyLayering( VARIANT_BOOL legacy )
```

IGeoPDFLayeringConfig3:

Get the PDF layer filtering state.

```
HRESULT get_FilterLayers( geoPDFLayeringFilterType* layeringFilter )
```

Set the PDF layer filtering state.

```
HRESULT put_FilterLayers( geoPDFLayeringFilterType layeringFilter )
```

Valid values, which may be combined with a bitwise-or operator:

```
geoPDFLayeringFilterNone
geoPDFLayeringFilterLayerOff
geoPDFLayeringFilterLayerOutOfScale
geoPDFLayeringFilterLayerInvalid
geoPDFLayeringFilterLayerMissing
geoPDFLayeringFilterGridOff
geoPDFLayeringFilterGridMissing
geoPDFLayeringFilterAnnoGroupOff
geoPDFLayeringFilterAnnoGroupMissing
geoPDFLayeringFilterMissing
geoPDFLayeringFilterEmptyPdfLayer
geoPDFLayeringFilterEmptyPdfGroup
geoPDFLayeringFilterEmptyPdfMenu
```

GeoPDFMenu, GeoPDFGroup and GeoPDFLayer COM Classes

The GeoPDFMenu, GeoPDFGroup and GeoPDFLayer COM classes are used to construct GeoPDF layer schemes. Each of these classes models a corresponding node type in GeoPDF layer trees: menus, groups and layers. A GeoPDF menu is collapsible and expandable, but does not control visibility of its content. It can contain menus, groups and layers. A GeoPDF group is also collapsible and expandable, but does control visibility of its content. It can contain groups and layers. A GeoPDF layer is not collapsible or expandable and controls visibility of GeoPDF graphic content. A GeoPDF layer does not correspond to an ArcObjects layer. A GeoPDF layer can contain the graphic content of one or more ArcObjects objects of any of the supported types. The graphic content of ArcObjects objects not assigned to a GeoPDF layer is always visible in the GeoPDF.

An ArcObjects object should not be referenced more than one time in a GeoPDF layer scheme. This requirement is not enforced by ExportGeoPDF. The result is undefined and may cause a crash or invalid GeoPDF.
Each of these classes implements the IGeoPDFLayerNode interface required by the IGeoPDFLayeringConfig interface methods. IGeoPDFLayerNode is an indicator interface, having no methods. Below are the primary interfaces for each of these classes.

IGeoPDFMenu:

Get/put the menu name.

```
HRESULT get_Name( BSTR* name )
HRESULT Put_Name( BSTR name )
```

Get the number of items in the menu.

```
HRESULT get_ItemCount( long* count )
```

Get an item.

```
HRESULT get_Item( long index,
                  IGeoPDFLayerNode** node )
```

Add an item.

```
HRESULT ItemInsert( long index,
                    IGeoPDFLayerNode* node )
```

Remove an item.

```
HRESULT ItemRemove( long index )
```

Remove all items.

```
HRESULT ItemRemoveAll()
```

IGeoPDFGroup:

Get/put the group name.

```
HRESULT get_Name( BSTR* name )
HRESULT put_Name( BSTR name )
```

Get the number of items in the group.

```
HRESULT get_ItemCount( long* count )
```

Get an item.

```
HRESULT get_Item( long index,
                              IGeoPDFLayerNode** node )
```

Add an item.

```
HRESULT ItemInsert( long index,
                    IGeoPDFLayerNode* node )
```

Remove an item.

```
HRESULT ItemRemove( long index )
```

Remove all items.

```
HRESULT ItemRemoveAll()
```

Get/put the group initial visibility.

```
HRESULT get_Visible( VARIANT_BOOL* visible )
HRESULT put_Visible( VARIANT_BOOL visible )
```

IGeoPDFLayer:

Get/put the layer name

```
HRESULT get_Name( BSTR* name )
HRESULT put_Name( BSTR name )
```

Get the number of items in the layer.

```
HRESULT get_ItemCount( long* count )
```

Get an item.

```
HRESULT get_Item( long index,
                  IGeoPDFContent** ppContent )
```

Add an item.

```
HRESULT ItemAdd( IGeoPDFContent* pContent )
```

Remove an item.

```
HRESULT ItemRemove( IGeoPDFContent* pContent )
```

Remove all items.

```
HRESULT ItemRemoveAll()
```

Get/put the layer initial visibility.

```
HRESULT get_Visible( VARIANT_BOOL* visible )
HRESULT put_Visible( VARIANT_BOOL visible) ;
```

GeoPDFContentSimple and GeoPDFContentAnnoGroup COM Classes

The GeoPDFContentSimple and GeoPDFContentAnnoGroup COM classes are used for assigning ArcObjects objects to GeoPDFLayer class instances. GeoPDFAnnoGroup class is used for map annotation groups. All other supported object types are handled by the GeoPDFContentSimple class.

GeoPDFContentSimple accepts the IUnknown interface pointer of an object to be rendered in the layer. It supports objects that implement IGroupElement, IMapFrame, ILayer, IElement, IMapGrid and ILocatorRect. For instances of the GroupElement and MapFrame classes the graphic content controlled by the layer consists of the element border, background and drop shadow. For instances of the BasemapSubLayer class it is necessary to pass the wrapped layer returned by IBasemapSubLayer.Layer(). Objects the implement IGroupLayer or IBasemapSubLayer are not accepted.

Annotation groups are handled by GeoPDFContentAnnoGroup as indexes off the CompositeGraphicsLayer object that contains them. An index value of zero indicates the default annotation group.

Both classes implement the IGeoPDFContent interface required by the GeoPDFLayer class. IGeoPDFContent is an indicator interface with no methods. The primary interface of each class follows.

IGeoPDFContentSimple:

Set/put the content.

```
HRESULT Content( IUnknown* pContent )
HRESULT Content( IUnknown** ppContent )
```

IGeoPDFContentAnnoGroup:

Set the content.

```
HRESULT PutAnnoGroup( ICompositeGraphicsLayer* pDefaultAnnoGroup,
                      long index )
```

Get the content.

```
HRESULT GetAnnoGroup( ICompositeGraphicsLayer** ppDefaultAnnoGroup,
                      long* pIndex )
```

GeoPDFLayeringHelper COM Class

The GeoPDFLayeringHelper COM class provides higher-level automation of GeoPDF layering. It currently provides the ability to create only a legacy layering scheme from a PageLayout or Map object. The ExportGeoPDF class uses the GeoPDFLayeringHelper class to dynamically generate its legacy layering scheme at export time when configured to use legacy layering.

IGeoPDFLayeringHelper:

This is an indicator interface.

IGeoPDFLegacyLayering:

Create a legacy layering scheme from a PageLayout or a Map object.

```
HRESULT CreateLegacyLayers( IUnknown* pPageLayoutOrMap,
                            IGeoPDFLayeringConfig* pConfig )
```

GeoPDFHelperCOM Class

The GeoPDFHelper COM class is provided to assist with exporting data-driven pages. It provides the ability to merge single-page PDFs into a multi-page PDF.

IGeoPDFHelper:

This is an indicator interface.

IGeoPDFMerge

Append a GeoPDF to another, optionally saving to a new path.

```
HRESULT Append( BSTR srcPath,
                BSTR appendToPath,
                BSTR saveToPath )
```

Merge a collection of GeoPDFs, saving to a path.

```
HRESULT Merge( IFileNames* pPdfs, BSTR
               saveToPath, ITrackCancel*
               pTrackCancel )
```

GeoPDF Export Code Examples

The standard ESRI developer samples for exporters largely work in the same way with ExportGeoPDF, except for the differences detailed above. It exports the current ArcMap view to a predefined filename.

```
Private Declare Function GetDeviceCaps Lib "gdi32" (ByVal hdc As Long, ByVal
nIndex As Long) As Long
Private Declare Function GetDC Lib "user32" (ByVal hwnd As Long) As Long Private
Declare Function ReleaseDC Lib "user32" (ByVal hwnd As Long, ByVal hdc As Long)
As Long

Sub ExportViewToGeoPDF()

  Dim pMxDocument As IMxDocument
  Set pMxDocument = ThisDocument

  Dim pActiveView As IActiveView
  Set pActiveView = pMxDocument.ActiveView

  ' create a GeoPDF exporter
  Dim pExport As IExport
  Set pExport = New ExportGeoPDF

  Dim pExport2 As IExportGeoPDF2
  Set pExport2 = pExport

  ' let the GeoPDF exporter know what it is that you intend to export
  Set pExport2.ActiveView = pActiveView

  pExport.ExportFileName = "C:\test_geopdf.pdf"
  pExport.Resolution = 300

  Dim tmpDC As Long
  tmpDC = GetDC(0)
  Dim screenResolution As Integer
  screenResolution = GetDeviceCaps(tmpDC, 88)
  ReleaseDC 0, tmpDC

  Dim scaleFactor As Double
  scaleFactor = pExport.Resolution / screenResolution

  Dim exportRECT As tagRECT
  With exportRECT
    .Left = pActiveView.ExportFrame.Left * scaleFactor
    .Top = pActiveView.ExportFrame.Top * scaleFactor
    .Right = pActiveView.ExportFrame.Right * scaleFactor
    .bottom = pActiveView.ExportFrame.bottom * scaleFactor
  End With

  Dim pPixelBoundsEnv As IEnvelope Set
  pPixelBoundsEnv = New Envelope
  pPixelBoundsEnv.PutCoords exportRECT.Left, _
                            exportRECT.Top, _
                            exportRECT.Right, _
                            exportRECT.bottom
  pExport.PixelBounds = pPixelBoundsEnv
```

```
    ' the GeoPDF exporter will perform rendering within this call
    pExport.StartExporting

    ' the usual call to IActiveView.Output at this point, which is required
    ' by other exporters, should not be performed

    ' the GeoPDF exporter will finish writing the GeoPDF within this call
    pExport.FinishExporting  pExport.Cleanup

  End Sub
```

GeoPDF View Extension

The GeoPDF View extension provides an ArcMap user interface and the GeoPDFViewLayer coclass, which implements a custom layer for displaying a GeoPDF file within ArcMap. In addition to standard ArcObjects interfaces it provides the IGeoPDFViewLayer, IGeoPDFViewLayer2, IGeoPDFViewControl and IGeoPDFAnnot interfaces.

GeoPDF View Usage

In this release, the GeoPDFLayer coclass has been renamed to GeoPDFViewLayer; the IGeoPDFLayer interface has been renamed to IGeoPDFViewLayer; and the IGeoPDFLayer2 interface has been renamed to IGeoPDFViewLayer2.

The standard layer COM interfaces presented by GeoPDFLayer coclass operate like any other ArcObjects layer, but as with many other layer types additional layer-specific interfaces are implemented. To view a GeoPDF file in ArcMap, cocreate a GeoPDFViewLayer instance. Call the IGeoPDFViewLayer.LoadMapFrame() method to load a GeoPDF map frame into the layer. Each GeoPDFViewLayer instance can display only a single map frame from a GeoPDF file; however, many GeoPDFViewLayer instances can be created. Insert the GeoPDFViewLayer into an ArcMap data frame with its ILayer interface, as you would any other layer type.

Use the IGeoPDFViewControl interface to control how the GeoPDF map frame is displayed. The visibility state of layers in the GeoPDF file can be set with the IGeoPDFViewControl.LayerStates property. This requires creating an XML string describing the GeoPDF layer structure and the state of each layer in that structure. Retrieve the default state for an example of this string.

Display of annotations can be toggled with the IGeoPDFViewControl.AnnotState property. GeoPDF annotations can be part of a GeoPDF layer, so visibility of a specific annotation may be subject to layer visibility. If the layer is not currently visible, the annotation will not be visible, regardless of the state of this property.

An option to clip to the page is provided because at the time there may be information on the page that is relevant to a map but not contained to the map's neatline. Selecting to clip to the GeoPDF map frame neatline or to the GeoPDF page is accomplished via the IGeoPDFViewControl.ClipToMapFrame property. When the clip is to the page the entire page will be displayed, including other map frames on the page. They will not be georegistered, which can be confusing to the user. Clipping to the map frame is recommended in most cases.

In addition to manual control of layer visibility, GeoPDF allows layer visibility to be controlled by the GeoPDF zoom level, or magnification. Sometimes to improve display performance detailed layers will not display until the user zooms in to some zoom level defined for that layer. For this reason the IGeoPDFViewControl.Zoom property allows controlling the effective GeoPDF zoom level used for display of the GeoPDF layer. This zoom level is expressed as a percentage. The default is 1.0, equivalent to
100% in the Adobe Acrobat user interface. For those accustomed to thinking in terms of map scales, a desired GeoPDF zoom value can be calculated as the ratio between the GeoPDF map scale and the desired map scale. The former is retrieved via the IGeoPDFViewLayer2.MapScale property. If the GeoPDF mapframe scale is 1:100000 and the desired map scale is 1:25000, the needed GeoPDF zoom is
100000/25000, or 4.0, equivalent to 400% in the Adobe Acrobat user interface,
GeoPDFViewLayer supports retrieving information about annotations in the GeoPDF. The IGeoPDFAnnot interface provides this functionality. The methods IGeoPDFAnnot.GetAnnotAtPoint() and IGeoPDFAnnot.GetAnnotsInRange() retrieve indexes into an array of annotations on the page. The former will retrieve the first annotation whose graphic range contains the specified point. The latter will
retrieve all annotations whose ranges intersect or are contained within the specified range. Both methods have an argument that controls whether the retrieval respects the current layer visibility context. If
enabled, the method will not return annotations on layers that are not currently visible. GetAnnotsInRange() also has an argument that controls if annotations must fall completely within the specified graphic range.

The remainder of the methods in the IGeoPDFAnnot interface use the indexes retrieved by GetAnnotAtPoint() and GetAnnotsInRange() to retrieve information about a specific annotation. All but GetAnnotGeometry() return textual information. GetAnnotGeometry() currently returns the graphic range of the annotation as an Envelope in the native spatial reference of the GeoPDF layer.

GeoPDF View COM Interfaces

The GeoPDFViewLayer coclass implements the following interfaces that can be found in ESRI ArcObjects documentation: ILayer, ILayerDrawingProperties, IGeoDataset, ILayerPosition and IPersistStream.

The GeoPDFViewLayer coclass implements the following interfaces that are specific to this coclass: IGeoPDFViewLayer, IGeoPDFViewLayer2, IGeoPDFViewControl and IGeoPDFAnnot. These are given below.

IGeoPDFViewLayer:

Load and retrieve information about a GeoPDF map frame. Indexes are zero-based.

```
HRESULT LoadMapFrame( BSTR
                      pdfName
                      , long
                      pageNum
                      , long
                      frameNu
                      m,
                      IStatusBar *pStatusBar )
```

```
HRESULT GetMapFrame( BSTR*
                     pdfName,
                     long*
                     pageNum,
                     long*
                     frameNum
                     )
```

IGeoPDFViewLayer2:

Get the GeoPDF mapframe map scale. 1:10000 would be returned at 10000.

```
HRESULT get_MapScale( double* mapScale )
```

IGeoPDFViewControl:

Get/put the current GeoPDF layer visibility states as an XML string.

```
HRESULT get_LayerStates( BSTR*

layerStates ) HRESULT put_LayerStates(

BSTR layerStates )
```

Get/put the current GeoPDF annotation display state.

```
HRESULT get_AnnotState( VARIANT_BOOL*

annotState ) HRESULT put_AnnotState(

VARIANT_BOOL annotState )
```

Get/put the current GeoPDF zoom/magnification.

```
HRESULT get_Zoom( float*

zoom ) HRESULT put_Zoom(

float zoom )
```

Get/put the current GeoPDF clip state.

```
HRESULT get_ClipToMapFrame( VARIANT_BOOL*

clip ) HRESULT put_ClipToMapFrame(

VARIANT_BOOL clip )
```

IGeoPDFAnnot:

Get the annotation at a point.

```
HRESULT GetAnnotAtPoint( IPoint* pPoint,
                         VARIANT_BOOL
                         bUseContext, long*
                         pIndex )
```

Get a collection of annotations in a range.

```
HRESULT GetAnnotsInRange( IEnvelope* pRange,
                          VARIANT_BOOL
                          bUseContext,
                          VARIANT_BOOL
                          bContained, long*
                          pCount,
                          long** pIndexes )
```

Get an annotation’s title.

```
HRESULT GetAnnotTitle( long
                       index,
                       BSTR*
                       pTitle )
```

Get an annotation’s date.

```
HRESULT GetAnnotDate( long
                      index,
                      BSTR*
                      pDate )
```

Get an annotation’s content.

```
HRESULT GetAnnotContent( long index,
                            BSTR*
                          pContent )
```

```
HRESULT GetAnnotType( long
                      index,
                      BSTR*
                      pType )
```

Get an annotation’s geometry.

```
HRESULT GetAnnotGeometry( long index,
                          IGeometry** pGeometry )
```

GeoPDF View Code Sample

The given example creates a new layer that displays a GeoPDF and inserts it into the currently selected map in ArcMap.

```
Sub AddGeoPDFLayer()

  Dim pGeoPDFLayer As IGeoPDFViewLayer
  Set pGeoPDFLayer = New GeoPDFViewLayer

  pGeoPDFLayer.LoadMapFrame "C:\GeoPDF.pdf", 0, 0,

  Nothing

  Dim pLayer As ILayer
  Set pLayer =

  pGeoPDFLayer

  pLayer.Name =

  "GeoPDF"

  Dim pMxDocument As IMxDocument
  Set pMxDocument = ThisDocument

  Dim pMap As IMap
  Set pMap = pMxDocument.FocusMap

  pMap.AddLayer pLayer

End Sub
```

GeoMark Import Extension

The GeoMark Import extension provides an ArcMap user interface and the GeoMarkHelper coclass. GeoMarkHelper can translate a GeoMark store into an ESRI File Geodatabase (GDB) or shapefiles. The extension translates the TWX store and creates a group layer that is returned to the caller. The links and attachments become layer hyperlinks, and in the GDB case TWX styles translate into attribute-based symbology.

For File Geodatabase import, the contents of the TWX are stored as a series of GDB feature layers and tables. The attachments are stored in the "_Resources" subdirectory of the GDB directory. Export will completely overwrite a database that already exists.

For shapefile import, the contents are stored as a series of shapefiles for feature layers and dBase files for tables. The attachments are stored in the same directory as the rest of the output. The shape directory must already exist. The existing files will be overwritten.

GeoMark Import Usage

GeoMarkHelper implements three interfaces: IGeoMarkHelper, IGeoMarkImport2 and IGeoMarkImport. GeoMark import is accomplished by instantiating the GeoMarkHelper coclass and using the IGeoMarkImport2 or the IGeoMarkImport interface. To create a file geodatabase, use IGeoMarkImport2.CreateFgdbFromTwx(). IGeoMarkImport2.AddFGDBGeoMarks adds layers from the file geodatabase to a map. To create shapefiles, use IGeoMarkImport.CreateLayer().

GeoMark Import COM Interfaces

The GeoMarkHelper coclass implements the following interfaces that are specific to this coclass: IGeoMarkHelper, IGeoMarkImport2, and IGeoMarkImport. These are given below.

IGeoMarkHelper

This is an identifier interface; it has no properties or methods.

IGeoMarkImport

Create a layer from a GeoMark store. Data will be written to `shpPath` with `base` filename prefixes.

```
HRESULT   CreateLayer(   BSTR
                      twzPath
                      ,   BSTR
                      shpPath
                      ,   BSTR
                      base,
                           ILayer**
                        geoMarkLayer )
```

IGeoMarkImport Code Sample

This sample creates a new layer that displays Geomarks imported into a series of Shapefiles and dBase files and inserts it into the currently selected map in ArcMap.

```
Sub AddGeoMarkLayer()

    Dim pMxDocument As IMxDocument
    Set pMxDocument = ThisDocument

    Dim pMap As IMap
    Set pMap = pMxDocument.FocusMap

    Dim pGeoMarkImport As IGeoMarkImport
    Set pGeoMarkImport = New GeoMarkHelper

    Dim pLayer As ILayer
    Set                     pLayer                        =
                                            pGeoMarkImp
                                            ort.CreateL
                                            ayer("c:\ge
                                            omark.zip",
                                            "c:\my_geom
                                            arks",
                                            "batch01")

    pMap.AddLayer pLayer

End Sub
```

IGeoMarkImport2

Import the GeoMarks into a file geodatabase. Data will be written to fgdbPath.

```
      HRESULT CreateFGDBFromTwx( BSTR
                                 twzPath,
                                 BSTR
                                 fgdbPath
                                 )
```

```
Add layers to a map from the file geodatabase imported from GeoMarks.
This method will also apply symbology based on GeoMark styles and
create relations between GeoMarks and form data.
```

```
      HRESULT AddFGDBGeoMarks( BSTR
                               fgdbPath
                               , IMap*
                               pMap )
```

IGeoMark Import2 Code Sample

This sample imports GeoMarks into a file geodatabase and displays them on the selected map in ArcMap.

```
Sub AddGeoMarkLayer()

    Dim pMxDocument As IMxDocument
    Set pMxDocument = ThisDocument

    Dim pMap As IMap
    Set pMap = pMxDocument.FocusMap

    Dim pGeoMarkImport As IGeoMarkImport2
    Set pGeoMarkImport = New GeoMarkHelper
   pGeoMarkImport.CreateFGDBFromTwx(“c:\geomar
   k.twx”,
                                 “c:\my_geomarks\batch01.gdb”)

   pGeoMarkImport.AddFGDBGeoMarks(“c:\my_geomarks\batch0
                               1.gdb”, pMap)

    pMap.AddLayer pLayer

End Sub
```

IGeoMarkImport3

Import multiple GeoMark stores into a file geodatabase.

```
      HRESULT CreateFGDBFromTwxMulti( IStringArray* twzPaths,
                                      BSTR fgdbPath,
                                      VARIANT_BOOL keepDuplicates )
```

Test if a file geodatabase is valid.

```
HRESULT IsValidGeoMarksFGDB( BSTR fgdbPath,
                             VARIANT_BOOL* isValid )
```

GeoMark Export Extension

The GeoMark Export extension provides an ArcMap user interface and the GeoMarkExporter coclass. GeoMarkExporter can translate a feature layer into a GeoMark store. Hyperlinked documents can be exported as attachments in the GeoMark store. Export can be limited to selected features.

GeoMark Export Usage

GeoMark export is accomplished by instantiating the GeoMarkExporter coclass and using IGeoMarkExport.ExportLayer() to create a store from a layer or add a layer to an existing store.

GeoMark Export COM Interfaces

The GeoMarkExporter coclass implements the following interfaces that are specific to this coclass: IGeoMarkExporter and IGeoMarkExport. These are given below

IGeoMarkExporter

This is an identifier interface; it has no properties or methods.

IGeoMarkExport

Test a layer for compatibility with GeoMark export.

```
HRESULT CanExportLayer( IFeatureLayer*
                        pExportLayer, IEnvelope*
                        pExtent,
                        esriSpatialRelEnum
                        spatialRel, VARIANT_BOOL
                        selectedOnly,
                        VARIANT_BOOL* canExport )
```

Export a layer to a GeoMark store. Create a new store or add to an existing store. Filter by the envelope and the spatial relationship. Reproject into the envelope's spatial reference.

```
HRESULT ExportLayer( IFeatureLayer* pExportLayer,
                     IEnvelope* pExtent,
                     esriSpatialRelEnum spatialRel,
                     VARIANT_BOOL selectedOnly,
                     VARIANT_BOOL includeDocs,
                     VARIANT_BOOL includeKML,
                    BSTR twzPath,
                   BSTR twzSchemaName )
```

IGeoMarkExport2

Get the field used for the GeoMark 'name' field.

```
HRESULT get_NameField( BSTR* nameField )
```

Set the field used for the GeoMark ‘name’ field.

```
HRESULT put_NameField( BSTR nameField )
```

Get the field used for the GeoMark ‘subject field.

```
HRESULT get_SubjectField( BSTR* subjectField )
```

Set the field used for the GeoMark ‘subject field.

```
HRESULT put_SubjectField( BSTR subjectField )
```

IGeoMarkUpdate

Insert a file into a GeoMark store.

```
HRESULT InsertFile ( BSTR* twzPath,
                     BSTR insertPath )
```

GeoMark Export Code Sample

This sample exports a map layer to a GeoMark store.

```
    Sub ExportGeoMarks()

        Dim pMxDocument As IMxDocument
        Set pMxDocument = ThisDocument

        Dim pMap As IMap
        Set pMap = pMxDocument.FocusMap

        Dim pLayer As ILayer
        Set pLayer = pMap.Layer(0)

        Dim pFeatureLayer As IFeatureLayer
        Set pFeatureLayer = pLayer

        Dim pActiveView As IActiveView
        Set pActiveView = pMap

        Dim pExtent As IEnvelope
        Set pExtent = pActiveView.Extent

        Dim pGeoMarkExporter As IGeoMarkExporter
        Set pGeoMarkExporter = New GeoMarkExporter

        Dim pGeoMarkExport As IGeoMarkExport
        Set pGeoMarkExport = pGeoMarkExporter

        pGeoMarkExport.ExportLayer pFeatureLayer, _
                                   pExtent, _
                                   esriSpatialRelIntersect
                                   s, _
                                False,
                                True,
                                False, _
                                "C:\export.twz"

End Sub
```