Oracle® Outside In Viewer for ActiveX

Developer's Guide

Release 8.5.2

E12487-09

April 2015

ORACLE

Oracle Outside In Viewer for ActiveX Developer's Guide, Release 8.5.2

E12487-09

Primary Author: Mike Manier

Contents

Preface

This document describes the installation and usage of the Outside In Viewer for ActiveX Software Developer's Kit (SDK).

Audience

This document is intended for developers who are integrating Outside In Viewer for ActiveX into Original Equipment Manufacturer (OEM) applications.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Related Documents

For more information, go to:

`http://www.oracle.com/technetwork/indexes/documentation/index.html#middleware`

and click on Outside In Technology.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Introduction

Welcome to Outside In Viewer for ActiveX, the fastest, easiest way to add powerful file access capabilities to your Visual Basic/C++ applications. Whether you're an experienced programmer or just learning, this control will allow you to view, print, cut, copy and paste more than 600 document types with just a few keystrokes.

This powerful control is based on the core viewing technology found in mainstream applications. You will not find another file access system on the market with the variety of formats, up-to-date filters and fidelity as found in Outside In.

There may be references to other Oracle Outside In Technology SDKs within this manual. To obtain complete documentation for any other Oracle Outside In product, see:

http://www.oracle.com/technetwork/indexes/documentation/index.html#middleware

and click on Outside In Technology.

This chapter includes the following sections:

What's New in this Release

- The updated list of supported formats is linked from the page http://www.outsideinsdk.com/. Look for the data sheet with the latest supported formats.
- If you are installing the ActiveX control into the Visual C++ or .NET developer's environment, the following caution has been added. If you are building 64-bit applications using a 32-bit IDE, you need to install the 32-bit control. 32-bit IDEs look for the 32-bit ActiveX control at compile time to create the callable wrapper for the component. Alternatively, you could create interop assembly for the control by running the tool TlbImp. More instructions on this tool and using the generated interop assembly can be found on MSDN.
- A new option, RenderEmbeddedFonts, has been added to disable the use of embedded fonts in PDF input files.
- A new option, StrokeText, has been added to display text as graphical primitives in an AutoCAD file.

- A new method, GetPageCount, counts the number of pages in the currently viewed file.
- A new event, InformationMessage, is generated when the Viewer encounters a file that it cannot properly render.

What Does the Outside In Viewer for ActiveX Do?

The ActiveX control may be used for simple applications that need to view a document as well as complex custom applications that require searching, e-mail attachment viewing, or custom publishing techniques. Outside In can also add value to Intranet technologies with its support for documents embedded within HTML and XML files.

You'll quickly discover the ease of integration and the power behind the Outside In technology.

This ActiveX control is based on the industry's leading file-access technology and integrated into mainstream applications from companies such as Microsoft, Lotus, Xerox and DEC. Outside In provides file viewing capabilities which offer software developers the highest fidelity, best performing and most stable support for the broadest number of file types across multiple operating systems.

There are several benefits to using Outside In over other solutions:

- Industry-standard filters: The Outside In viewing filters have been tested by hundreds of thousands of users while embedded within the industry's top software programs. This ensures high-quality filters that are submitted to constant quality control procedures.
- Up-to-date file formats: One of the most difficult aspects of creating file filters is keeping them up-to-date with evolving formats. Because of relationships with leading vendors, Oracle's filters handle the latest versions of the various document files.
- Speed of development: The control is extremely easy to use. Just place the control on a form, set a few properties, and a complete application for viewing documents is accessible.

Definition of Terms

Term	Definition
Developer	Someone integrating this technology into another technology or application. Most likely this is you, the reader.
Source File	The file the developer wishes to view.

Directory Structure

Each Outside In product has an sdk directory, under which there is a subdirectory for the platform on which the product ships. For the Viewer for ActiveX, the structure is ax/sdk/ax_win-x86-32_sdk or ax/sdk/ax_win-x86-64_sdk. Under this directory are the following four subdirectories:

- **install** - Contains files needed to set up the control for demo purposes.

- **redist** - Contains only the files that the customer is allowed to redistribute. These include all the compiled modules, filter support files, cmmap000.bin, and third-party libraries.
- **sdk** - Contains the other subdirectories that used to be at the root-level of an sdk (common and resource).

In the root platform directory (for example, ax/sdk/ax_win-x86-32_sdk), there is one file: README. This explains the contents of the sdk.

2

Getting Started

This chapter describes getting started with the Viewer for ActiveX. The Outside In Viewer for ActiveX control provides developers of container-aware applications the ability to view and print documents of varying types from within their own application.

This is the same technology that is provided through the Outside In Viewer Technology API. Developers using Microsoft Visual Basic/C++/C# and Borland's Delphi will all find this control easy to use and powerful enough to handle document viewing needs.

This chapter includes the following sections:

Installation and System Requirements

In order for applications to use this control, the control must be installed on each workstation on which the application is to run.

Outside In requires the msvcr80.dll redistributable module.

C++.NET sample applications require Visual Studio 2005 SP1 C-runtime libraries.

MFC applications require MFC 4.2 runtime libraries.

The system requirements for this product are as follows:

Requirement	Value
Disk Space	18 Mb
CPU	Intel 486 or higher
Operating System	Microsoft Windows 2000 or newer
Memory	16Mb
Compiler	Visual C++ and Visual Basic Reference 6.0

NSF Support

Notes Storage Format (NSF) files are produced by the Lotus Notes Client or the Lotus Domino server. The NSF filter is the only Outside In filter that requires the native application to be present to filter the input documents. Due to integration with an outside application, NSF support will not work with redirected I/O nor will it work

when an NSF file is embedded in another file. Either Lotus Notes version 8 or Lotus Domino version 8 must be installed on the same machine as OIT. A 32-bit version of the Lotus software must be used if you are using a 32-bit version of OIT. A 64-bit version of the Lotus software must be used if you are using a 64-bit version of OIT. SysLotusNotesPath should be set to the directory containing the nnotes.dll. NSF support is only available on the Win32 and Win x86-64 platforms.

Options and Information Storage

The technology creates two sets of information: the default options, and a list of available display engines. In the Windows implementations, these lists are built by the technology as needed, usually the first time ActiveX is run. You do not need to ship these lists with your application. The lists are automatically regenerated if corrupted or deleted.

The files used to store this information are stored in a .oit subdirectory in the following location:

\documents and settings*user name**Application Data*

If an .oit directory does not exist in the user's directory, the directory is created automatically by the technology. The files are automatically regenerated if corrupted or deleted.

The files are:

- Display engine lists: *.d
- Persistent options:*.opt

These file names are intended to be unique enough to avoid conflict for any combination of machine name and install directory. This allows the user to run products in separate directories without having to reload the files above. The file names are built from an 11-character string derived from the directory the Outside In technology resides in and the name of the machine it is being run on. The string is generated by code derived from the RSA Data Security, Inc. MD5 Message-Digest Algorithm.

Getting Started With Visual Basic 6.0

Before using the control, it must be added to the Visual Basic toolbox.

1. Start Visual Basic 6.0.
2. Open an existing project or select **New Project**.
3. Select the Project Menu, choose **Components**.
4. Check the box next to **Outside In for ActiveX Control** and click **OK**.

The control appears in the toolbox and may be added to a form as any other control would be added.

Getting Started With Visual C++

Note:

If you are building 64-bit applications using a 32 bit IDE, you need to install the 32–bit control. 32-bit IDEs look for the 32–bit ActiveX control at compile time to create the callable wrapper for the component. Alternatively, you could create interop assembly for the control by running the tool TlbImp. More instructions on this tool and using the generated interop assembly can be found on MSDN.

To install the ActiveX control into the Microsoft Visual C++ 6.0 developers' environment:

1. Start Visual C++.
2. Open an Existing workspace or select a New workspace.
3. From the Project menu, select **Add to Project** then **Components and Controls**.
4. Open the Registered ActiveX Controls folder.
5. Select the ActiveX control and click **Insert**.

Getting Started with .NET Applications

Note:

If you are building 64-bit applications using a 32 bit IDE, you need to install the 32–bit control. 32-bit IDEs look for the 32–bit ActiveX control at compile time to create the callable wrapper for the component. Alternatively, you could create interop assembly for the control by running the tool TlbImp. More instructions on this tool and using the generated interop assembly can be found on MSDN.

To install the ActiveX control into the .NET developers' environment:

1. Start Visual Studio.
2. Open an Existing project or select a New project.
3. From the toolbox tab, select **Choose Items**.
4. Select the COM Components tab and select the ActiveX Control.
5. Select the ActiveX control and drop it on the dialog box.

Installing the ActiveX Control as Part of Your Application

Please do not ship the development installation package. Instead, add the following tasks to the installer for your application:

1. Copy the Filters directory onto the target machine.

2. Create a registry key for your application:

   ```
   HKEY_LOCAL_MACHINE\Software\OIX\executable name
   ```

   To minimize the chance of conflicts with other installations, the key name should be specified as a full path to your executable, with forward slashes for path separators.

   Note:

   Versions of this control prior to 8.3.5 had the default location of the FilterPath entry as HKLM\Software\OIX\OutsideX. Versions 8.3.7 and beyond will be using the HKLM\Software\OIX\OutsideX\{CLSID of control}\FilterPath. For the current version, the CLSID is {28AB815A-793C-44E1-8C03-1D73B06AD8DE}. If you cannot or do not use a unique FilterPath, then use this default value when installing the ActiveX product with your application.

3. Create a string value in that registry key with the name "FilterPath," whose value is the full path to the filters directory on the target machine.

4. Copy the outsidex852.ocx to the \system directory.

5. Register the outsidex852.ocx control.

   Note:

   Starting with the 8.3.7 release of Outside In Technology, the control has been branded with a new GUID/CLSID every time the API changes. This would require a rebuild of your application (after changing references to the new control and this varies from development environment to environment) to use this control.

The specific steps necessary to accomplish these tasks depend on the installer software you are using. However, if you are creating your own installer; you can register the outsidex852.ocx by calling regsvr32.exe outsidex852.ocx. Regsvr32.exe is also located in the System directory.

Note:

Versions of this control prior to 8.2.0 stored the filter path in a particular location in the system registry, even if multiple versions of the control were installed in different locations on the same system. Therefore only the last copy of the control installed on the system had access to its correct configuration information. Any other copy on the same system could experience errors or incompatibilities at runtime. Steps 2 and 3 above now enable multiple copies of the control to co-exist peacefully on the same system. Steps 2 and 3 can be skipped if the application will be responding to the GetTechPath event to provide the filter location.

The following is an example of a .reg file entry for registering the filter path:

```
[HKEY_LOCAL_MACHINE\SOFTWARE\OIX\c:/Program Files/App/myapp.exe]
"FilterPath"="c:\\Program Files\\App\\OIX\\Filters"
```

Libraries and Structure

The Outside In Technology modules have been renamed from previous versions by appending the version number. This is to prevent conflicts in applications required to run multiple versions of the technology simultaneously. We recommend that the previous version of the modules be removed before installing the update. If multiple versions are found in the technology directory, the latest version will be used.

Here is an overview of the files contained in the main installation directory for this product.

API DLLs

These DLLs implement the API. They should be linked with the developer's application. LIB files are included in the SDK.

File	Description
oswin*.dll	Interface to native GDI implementation (oswin32.dll is the module for Windows 32-bit implementation, and oswin64.dll is the module for Windows 64-bit implementation)
outsidex852.ocx	Outside In module installed in the System directory for Windows 32-bit and 64-bit implementation.
sccca852.dll	Content Access module (provides organized chunker data for the developer)
sccanno852.dll	Annotation module
sccch852.dll	Chunker (provides caching of and access to filter data for the display engine)
sccda852.dll	Data Access module
sccdu852.dll	Display Utilities module (includes text formatting)
sccfa852.dll	Filter Access module
sccfi852.dll	File Identification module (identifies files based on their contents). The File ID Specification may not be used directly by any application or workflow without it being separately licensed expressly for that purpose.
sccfmt852.dll	Formatting module (resolves numbers to formatted strings)
sccfnt852.dll	Font utility module
sccfs852.dll	Font services module
sccfut852.dll	Filter utility module
sccind852.dll	Indexing engine
scclo852.dll	Localization library (all strings, menus, dialogs and dialog procedures reside here)
sccole852.dll	OLE rendering module
sccsd852.dll	Schema Definition Module Manager (brokers multiple Schema Definition Modules)
sccta852.dll	Text Access module (provides straight text data for the developer)

File	Description
sccut852.dll	Utility functions (including IO subsystem)
sccvw852.dll	Viewer module (this is the DLL that your application loads, providing control of all viewer functions)
sccxt852.dll	XTree module
wvcore852.dll	The GDI Abstraction layer

Display Engine DLLs

File	Description
debmp852.dll	Bitmap (TIFF, GIF, BMP, PCX…)
dehex852.dll	Hexadecimal
devect852.dll	Vector/Presentation (PowerPoint, Impress, Freelance…)
dess852.dll	Spreadsheet/Database (Excel, Calc, Lotus 123…)
detree852.dll	Archive (ZIP, GZIP, TAR…)
dewp852.dll	Document (Word, Writer, WordPerfect…)

Filter and Export Filter Libraries

File	Description
vs*852.dll	Filters for specific file types (there are more than 150 of these filters, covering more than 600 file formats)

Premier Graphics Filters

File	Description
i*2852.dll	Import filters for premier graphics formats
isgdi32852.dll	Interface to premier graphics filters

Additional Files

File	Description
adinit.dat	Support file for the vsacad filter
cmmap000.bin	Tables for character mapping (all character sets)
oitnsf.id	Support file for the vsnsf filter
oit_font_metrics.db	Database with font information

Samples Overview

This SDK includes several sample applications that demonstrate the use of various features of the ActiveX control. Various samples are provided that were created with Visual C++ 6.0, Visual Basic 6.0 (32-bit only), and Inprise Delphi 4.0 (32-bit only).

- **DrawPage**: This sample demonstrates the DrawPage functionality. Functions demonstrated:

- Clear
- DeinitDrawPage
- ExtDrawPage
- GetDrawPageInfo
- PageFormatHeight
- PageFormatWidth
- PagePicture
- InitDrawPage
- ViewFile

- **RedirectOIX**: This sample demonstrates the Redirected I/O functionality. Functions demonstrated:
  - Clear
  - PrintOI
  - ViewFile
- **welcome**: A simple application to view WELCOME.DOC and the associated sample files. Functions demonstrated:
  - Clear
  - CopyToClip
  - PrintOI
  - ViewFile
- **simple**: This sample demonstrates the simplest implementation of the viewer technology. It shows the basics of how to create a view window, view a file, print the viewed file, and copy to the clipboard. Functions demonstrated:
  - Clear
  - CopyToClip
  - PrintOI
  - ViewFile
- **search**: This sample is essentially the same as **simple** except it adds the ability to search for strings in the file being viewed. Functions demonstrated:
  - Clear
  - Search
  - SearchNext
  - ViewFile

- **annotate**: This sample uses the viewer's Raw Text and Annotation ability to annotate all occurrences of the text "the" in the document. Functions demonstrated:
  - AddAnnotationHilite
  - Clear
  - CopyToClip
  - GetRawText
  - GoToAnnotation
  - RawTextEvent
  - SetActualCount
  - ViewFile
- **mdiview**: A simple multiple document interface viewer implementation. Demonstrates clipboard support, including the SelectAll function and the use of the ViewAs parameter within the ViewFile function to view text and HTML files in multiple character sets. Functions demonstrated:
  - CopyToClip
  - Search
  - SearchNext
  - SelectAll
  - ViewFile
- **options**: This sample shows how viewer options may be directly set by an application and how to access the viewer technology's dialogs and menus. Functions demonstrated:
  - Clear
  - CopyToClip
  - ClipboardOptions
  - DisplayOptions
  - PrintSetup
  - ViewFile

The following applications were created in Microsoft Visual Studio 2005. The following .Net sample applications are provided: SimpleCS, SearchAnnoCS, SimpleVB, SearchAnnoVB, SimpleCPP, and SearchAnnoCPP.

- **Simple(CS, VB, CPP)**: This sample demonstrates the simplest implementation of the Viewer Technology and Redirected I/O. It shows the basics of how to create a view window, view a file, print the viewed file, and copy to the clipboard. Functions demonstrated:
  - Clear

- CopyToClip
- PrintOI
- ViewFile

The following Redirected I/O events are also demonstrated:

- Close
- GetInfo
- Open
- Read
- Seek
- Tell

- **SearchAnno(CS, VB, CPP)**: This sample demonstrates searching and annotating documents. This sample uses the viewer's Raw Text and Annotation ability to annotate all occurrences of the text "the" in the document. Functions demonstrated:
  - AddAnnotationHilite
  - Clear
  - CopyToClip
  - GetRawText
  - GoToAnnotation
  - HiliteStyle
  - PrintOI
  - Search
  - SearchNext
  - SetActualCount
  - ViewFile

3

Viewing Documents

The Outside In viewing technology centers around the view window. Documents may be processed for viewing, printing or exchange via the clipboard. The methods, events and properties that control this process are described in this chapter.

This chapter includes the following sections:

Displaying Documents

One of the first requests made to the viewing control is to load a document. The viewer will automatically detect the type of file (based on file content) and will display the document. Documents are loaded by passing the ViewFile method (see ViewFile) a filename (or a flag to display the file selection dialog), and optional display information. When the viewer receives these parameters, the following will occur.

- Any file currently opened and displayed will be closed before the new document is processed.
- If specified, the system file selection dialog box will be displayed to collect the user's choice of document. In this case, the filename which is passed in the second parameter (FileSpec) will be used to initialize the file selection dialog box, otherwise the file will be validated and opened.
- Once the file has been identified and displayed, the FileInfoFileId properties (see FileInfoFileId) will contain information about the document type and the FileInfoName (see FileInfoName), and FileInfoDisplayName (see FileInfoDisplayName) will contain the actual file name and display name passed to the ViewFile method (see ViewFile), respectively.
- The Clear method (see Clear) may be called to reset the viewer to its non-viewing state.

The following example illustrates the use of these methods and properties.

```
Private Sub Clear_Click()
   Rem ** When the clear button is pressed, clear the file
   Rem from the viewer **

   oixctrl1.Clear
End Sub

Private Sub OpenFile_Click()

   Dim Error As Boolean
```

```
Rem ** Load the document returned from the Open File dialog box **
   Error = oixctrl1.ViewFile True, ""

Rem ** File information displayed **
   FileInfoFileIdName_Ctrl.Caption = oixctrl1.FileInfoFileIdName
   FileInfoFileID_Ctrl.Caption = oixctrl1.FileInfoFileID
   FileInfoName_Ctrl.Caption = oixctrl1.FileInfoName
   FileInfoDisplayName_Ctrl.Caption = oixctrl1.FileInfoDisplayName
End Sub
```

Special Considerations

There are a few formats that contain links to other documents. For example, an archive file (.ZIP) contains many files. The programmer may allow the user to open an archive file and subsequently view an embedded file by double clicking the filename as shown by the archive display engine. Similarly, documents with hyperlinks may be processed when the user selects a hyperlink object in the source document. In these situations, a ViewThisFile event (see ViewThisFile) will be generated to allow the programmer to handle the display of the embedded file as the viewer does not directly process these requests. It is up to the programmer to call the ViewFile with the appropriate parameters to display this new document.

The developer should *not* call the control's ViewFile method (see ViewFile) from within a ViewThisFile event (see ViewThisFile). This will produce unpredictable results. The programmer can, however, call another control's ViewFile method within the ViewThisFile event handler.

Notifications

Whenever the viewer is called to change the document being displayed, a FileChange event (see FileChange) will be generated. Two parameters allow the event handler to determine if a file is being opened or closed, and, if a file is being closed, whether or not it was due to a new file being opened. In this case, the user should expect another FileChange event to occur. The name of the file being opened or closed will be stored in the FileInfo* properties.

Outside In uses different display engines to present the document to the user through the viewing window. Each of these display engines can be configured with a default viewing mode, font, and other appearance-related options (for example, grid lines). When a different display engine is about to be loaded, the viewer will generate a DisplayEngineChange event (see DisplayEngineChange). The new display engine information can be found in the DisplayEngineName (see DisplayEngineName) and DisplayEngineType properties (see DisplayEngineType). These events will also be generated when a ViewFile method (see ViewFile) is called with a document that requires a different display engine that is currently loaded.

```
Private Sub oixctrl1_DisplayEngineChange()
   Rem ** Update information on the screen **
   DspEngineName_Ctrl.Caption = oixctrl1.DisplayEngineName
   DspEngineType_Ctrl.Caption = oixctrl1.DisplayEngineType
End Sub

Private Sub oixctrl1_FileChange(ByVal bOpen As Long, ByVal bOpenFollowing As
Long)
   If (bOpen) Then
      Rem ** File information displayed **
      FileInfoFileIdName_Ctrl.Caption = oixctrl1.FileInfoFileIdName
      FileInfoFileID_Ctrl.Caption = oixctrl1.FileInfoFileId
      FileInfoName_Ctrl.Caption = oixctrl1.FileInfoName
      FileInfoDisplayName_Ctrl.Caption = oixctrl1.FileInfoDisplayName
   Else
      FileInfoFileIdName_Ctrl.Caption = ""
```

```
      FileInfoFileID_Ctrl.Caption = ""
      FileInfoName_Ctrl.Caption = ""
      FileInfoDisplayName_Ctrl.Caption = ""
   End If
End Sub

Private Sub oixctrl1_ViewThisFile(ByVal FileSpec As Variant, ByVal iSpecType As
Integer, iReturn As Integer)
   Rem ** Display file in another Viewer window. Note: this
   Rem    nested file cannot be viewed in the same viewer.
   Rem    Therefore, to simulate this appearance, oixctrl2
   Rem    is the same size and location as oixctrl1 on
   Rem    the form and will be used to display embedded files
   Rem    (e.g. from an archive format). Visibility
   Rem    is swapped to view the embedded file. The Esc key swaps
   Rem    back to the primary viewer. **

   oixctrl2.ViewFile 0, FileSpec, iSpecType
   oixctrl2.Visible = True
   oixctrl1.Visible = False

   Rem ** file is being viewed **
   iReturn = 1
End Sub
```

Outside In attempts to identify the type of document specified in the ViewFile method (see ViewFile). However, there may be reasons that prevent the viewer from correctly identifying the file. In these cases, the viewer can be configured to display the document using a default-viewing format. This is configured by setting the FallbackFormat property (see FallbackFormat) to an appropriate value.

When viewing spreadsheets, databases or documents in draft mode, the DisplayFont property (see DisplayFont) may be set to a preferred font using a font object data type. Similarly, the viewer can be configured to scale the document fonts from 50% to 300% of the normal size (as specified in the document) with the FontScalingFactor property (see FontScalingFactor). The following code illustrates the use of these properties.

```
Private Sub Fallback_Ctrl_Click()
If (Fallback_Ctrl.ListIndex > 1) Then
   oixctrl1.FallbackFormat = Fallback_Ctrl.ListIndex - 1
End If
End Sub

Private Sub DisplayFont_Click()
   Dim NewFont As New StdFont

   Rem ** Show common font selection dialog, collect font info,
   Rem    update button text. **

   CommonDialog1.Flags = cdlCFBoth
   CommonDialog1.ShowFont
   DisplayFont.Caption = "DisplayFont: " + CommonDialog1.FontName

   Rem ** Put font information into StdFont variable and pass it
   Rem    off to oixctrl1. **

   NewFont.Size = CommonDialog1.FontSize
   NewFont.Name = CommonDialog1.FontName

   Set oixctrl1.DisplayFont = NewFont

End Sub

Private Sub FontScalingFactor_Ctrl_Click()
   Rem ** FontScalingfactor_Ctrl is a list box with scaling
   Rem    percentages listed. Each item has an ItemData value
   Rem    equal to the integer equivalent of the text. **
```

```
   oixctrl1.FontScalingFactor = FontScalingFactor_Ctrl.
   ItemData(FontScalingFactor_Ctrl.ListIndex)

End Sub
```

Printing

The Outside In viewer also provides an API subset to handle print request from the application. The programmer may want to handle some of the details of printing, such as displaying an Abort dialog, specifying an application provided hDC, or may choose to let the viewer default routines handle all print requests.

There are two methods associated with the printing subsystem: PrintSetup (see PrintSetup) and PrintOI (see PrintOI). The PrintSetup routine displays the system printer selection dialog box and allows the user to select the desired printer as well as change its settings. Upon return, the appropriate control properties will be populated with the selected printer information. The PrintOI method prints the currently viewed file. Parameters to the PrintOI routine control whether the Print Dialog box is displayed before printing, which of the printer properties to use for printer selection, and how the abort process is handled. The following code shows an example of using these methods.

```
Private Sub PrintSetup_Ctrl_Click()
   Rem ** Reset the abort event counter **
   PrinterAbort_Ctrl.Caption = 0

   Rem ** Invoke the system printer dialog box **
   oixctrl1.PrintSetup

   Rem ** Print the currently viewed file:
   Rem    P1   1   Display print setup dialog box(again)
   Rem    P2   128 Generate PrinterAbort Events. Other ORable
   Rem    flags include
   Rem      1 - Use the PrinterDC as stored in the property
   Rem          value instead of the one obtained from the
                default printer
   Rem      2 - If bit 1 is set, get the printer from the
                PrinterName, PrinterPort and PrinterDriver
   Rem          properties, otherwise, use the PrinterDC and
   Rem          use the PrinterName, PrinterPort and
   Rem          PrinterDriver properties only when displaying
   Rem          the standard Windows abort dialog.
   Rem    P3   0   Display the standard windows abort dialog
   Rem          with page numbers
   Rem    P4   0   Assume that the StartDoc printer sequence has
   Rem          already been performed.

   oixctrl1.PrintOI 1, 128, 0, 0

End Sub
```

If specified when calling the PrintOI method, the viewer generates PrinterAbort events (see PrinterAbort) throughout the print process. The event handler receives the device context to the printer, and an error value (for example, SP_OUTOFDISK), if applicable. Before returning from the event handler, the programmer must indicate whether the print process is to continue or should be aborted. An example printer abort event handler is shown in the following code sample.

```
Private Sub oixctrl1_PrinterAbort(ByVal hDC As Long, ByVal Err As Long,
   Continue As Long)

   Rem ** Update a on-screen counter to show the number of
   Rem    times this event was received **
```

```
   PrinterAbort_Ctrl.Caption = PrinterAbort_Ctrl.Caption + 1

   Rem ** Cancel the print request on error **
   If (Err) Then
      Continue = False
   Else
      Continue = True
   End If
End Sub
```

The following properties control how the viewer formats the document for printing.

- PrintCollate: TRUE. Multiple copies of the document are collated, otherwise, copies of each page will be collated together.
- PrintCopies: The number of copies of the viewed document to print.
- PrintStartPage / PrintEndPage: The starting and ending pages to be printed.
- PrinterDC: A system handle to the printer device context.
- PrinterDriver: The printer driver name (for example, PSCRIPT.DRV).
- PrinterName: The name of the printer (for example, HP LaserJet IV).
- PrinterPort: The system port of the printer (for example, LPT1).
- PrintFont: When no font is specified in the file, or when printing spreadsheets, databases or documents in draft form, this property specifies the font to be used.
- PrintMarginLeft, Right, Top, Bottom: These properties specify the four page margins to be used when printing a document.
- WhatToPrint: This property specifies (in viewer terms) what part of the document is to be printed. Valid values include:
  - PRINT_ALLPAGES
  - PRINT_SELECTION
  - PRINT_PAGERANGE (will use PrintStartPage/EndPage values)
  - PRINT_CURSECTION.
- UseDocPageSettings: TRUE. The printing information stored in the original document will be used to configure the print job.

Clipboard

The Outside In viewer provides the ability to exchange document text and graphics with other applications via the system clipboard. The CopyToClip method (see CopyToClip) will place the currently viewed document onto the system clipboard in the format specified by the ToClipboard property (see ToClipboard). Fifteen different data format are supported through this method including: TEXT, RTF, AMI, WINDIB, WINMETAFILE, and others. Before copying data from the viewer to the clipboard, however, the programmer should check the value of the read-only property ClipInfo (see ClipInfo) to determine if the viewer has completely processed the file and is ready to send the data to the clipboard.

For documents that do not specify a font or for spreadsheets, databases, or documents in draft mode, the font for the clipboard data may be specified using the ClipFont property (see ClipFont).

Another way to exchange data is through a mechanism known as "Drag-N-Drop." By setting the EnableDragNDrop property (see EnableDragNDrop) to TRUE, the user is allowed to drag selected text from the viewing window to a target application. Dragging files from the desktop or explorer to the viewer is not currently supported.

```
Private Sub Clipboard_Ctrl_Click()
   Rem ** Check the status of the clipboard and document
   Rem    processing. If ok, then
   Rem    place selected text on the clipboard as a windows
   Rem    metafile. **

   If (oixctrl1.ClipInfo > 0) Then
      oixctrl1.ToClipboard = SCCVW_CLIPFORMAT_WINDIB
      oixctrl1.CopyToClip
   End If
End Sub

Private Sub EnableDragNDrop_Ctrl_Click()
   Rem ** If checkbox is "checked", enable drag-n-drop from the
   Rem    viewer otherwise
   Rem    disable it. **

   If (EnableDragNDrop_Ctrl.Value = 1) Then ' if checked
      oixctrl1.EnableDragNDrop = True
   Else
      oixctrl1.EnableDragNDrop = False
   End If

End Sub
```

4

User Interface Options

This chapter describes user interface options. The Outside In Viewer for ActiveX contains a default user interface to process the user's mouse and keyboard events.

The following are examples of the types of events that can be processed:

- Text that is selected by the user may be accessed by the programmer for use in printing, searching or annotations.
- Right mouse clicks can invoke a context-sensitive menu for processing/viewing options.
- Default property values may be set by the user via viewer supplied dialog boxes.
- The viewer's scroll bars may be programmatically accessed and display options may be set for each of the viewer's display engines.

This chapter includes the following sections:

Text Selection

After a document has been loaded into the viewer, the user may scroll through the document in a manner similar to many of the word processing packages available. This also includes the ability to select text. Whenever a user changes the currently selected text or selects text for the first time, a SelChange event (see SelChange) will be generated. The programmer can identify the text that is selected by looking at the SelectionAnchor and SelectionEnd properties(see SelectionAnchor, and SelectionEnd) .

Text can be programmatically selected by setting the SelectionAnchor or SelectionEnd properties to a valid OixPos object or by calling the SelectAll method (see SelectAll) to cause all text in the viewer to be selected. For a discussion on how to set an OixPos object/variable, see Positions.

The following code shows how to use this API.

```
Private Sub oixctrl1_SelChange()
   Dim NumberofCharsSelected As Long

   Rem ** Each time the selection is changed, update a text field
```

```
   Rem    indicating the number of characters which are
   Rem    "selected" **

   NumberofCharsSelected =
   Abs(oixctrl1.GetActualCount(oixctrl1.SelectionAnchor) -
   oixctrl1.GetActualCount(oixctrl1.SelectionEnd))
   Selection_Ctrl.Caption = NumberofCharsSelected
End Sub

Private Sub SelectAll_Ctrl_Click()
   Rem ** When the button is pressed, select all of the text
   Rem    within the control. This will generate a SelChange
   Rem    event which in turn will update the selection count
   Rem    text field **

   oixctrl1.SelectAll
End Sub
```

Menus

When the user right clicks on the viewer window, the application will receive a ContextMenu event (see ContextMenu). The application then has the opportunity to present the user with a pop-up menu selection. If the application does not handle this event and the DoContextMenu property (see DoContextMenu) is set to TRUE, the default viewer context menu will be displayed, otherwise, the right click is ignored. In either case, the ContextMenu event will be generated.

The following are screen images of the viewer's default context menus.

Figure 1 Default Context Zoom Menu

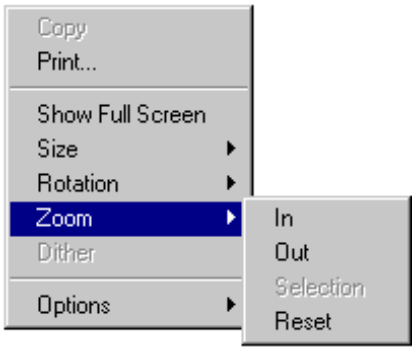


Figure 2 Default Context Size Menu

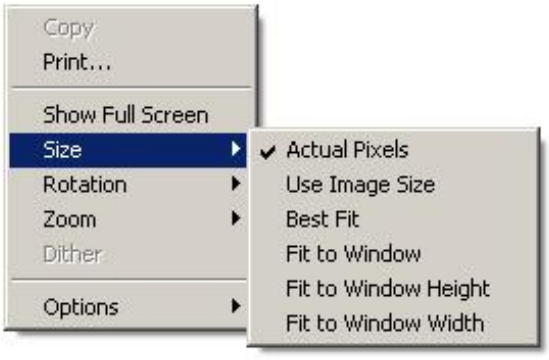


Figure 3 Default Context Rotation Menu

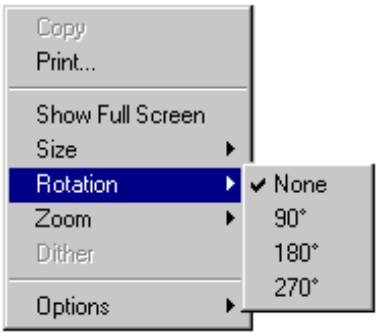
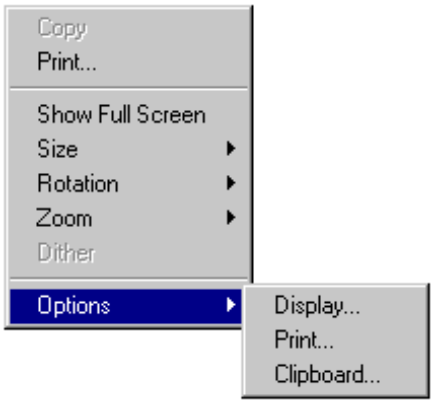


Figure 4 Default Context Options Menu



The following example shows how to respond to the ContextMenu event (see ContextMenu) to display a custom menu (also shown).

```
Private Sub ContextMenu_Ctrl_Click()
   Rem ** Checkbox to enable/disable control's default
   Rem    context menus is pressed.

   oixctrl1.DoContextMenu = ContextMenu_Ctrl.Value
End Sub

Private Sub oixctrl1_ContextMenu(ByVal lXPos As Long, ByVal
   lYPos As Long)

   Rem ** User has pressed the right mouse button. Therefore
   Rem    we will display our context menu **

   oixctrl1.DoContextMenu = False    ' Turn off control's
                                     ' default context menu

ContextMenu_Ctrl.Value = 0
   PopupMenu Form1.ContextMenu_Menu ' Pop up our context
                                     ' menu. Menu selections
                                     ' come back as events
End Sub
Private Sub OpenFile_Menu_Click()
   Rem ** Popup menu Open File is selected **

   OpenFile_Click ' Simulate a press of the Open File button
End Sub

Private Sub percent50_menu_Click()
   Rem ** Popup submenu 50% is selected **

   oixctrl1.FontScalingFactor = 50
```

```
   oixctrl1.WPDisplayMode = 1 ' Fontscaling factor only works in
                              ' draft mode so we'll change it for
                              ' the user
End Sub
Private Sub percent100_menu_Click()
   Rem ** Popup submenu 100% is selected **

   oixctrl1.FontScalingFactor = 100
End Sub
Private Sub percent200_menu_Click()
   Rem ** Popup submenu 200% is selected **
   oixctrl1.FontScalingFactor = 200
   oixctrl1.WPDisplayMode = 1 ' Change into
                          ' draft mode
                          ' so
                          ' FontscalingFactor
                          ' is shown
End Sub
```

Dialogs

Each of the viewing/printing/clipboard operations supply a default dialog box to allow the user to customize the processing of the document. These are invoked by calling the DisplayOpetions, PrintOptions, and ClipboardOptions methods (see DisplayOptions, PrintOptions, and ClipboardOptions). Prior to the display of these dialog boxes, the application will receive an EnableApp event (see EnableApp) with a parameter of FALSE to allow the developer to disable any code that might affect the view window. Changes in the dialog options will be reflected by updates to the corresponding property values once you click OK. Also, for each option changed, an OptionChange event (see OptionChange) will be generated. These events will be generated en-masse after clicking OK followed by an EnableApp event with a parameter of TRUE.

Changes to the property values will remain in effect only as long as the application is running. If these user preferences are to remain persistent, it is up to the application to save and restore the user choices upon application termination and startup.

The following code shows the use of these methods and events.

```
Private Sub DoPrtOptions_Menu_Click()
   CustomShowDialog 2, 0
End Sub

Private Sub CustomShowDialog(ByVal iDialog, ByVal iRemoveOption)
   Rem ** Display the dialog based on the parameter
   Select Case iDialog
      Case 1: oixctrl1.DisplayOptions
      Case 2: oixctrl1.PrintOptions
      Case 3: oixctrl1.ClipboardOptions
End Select
End Sub

Private Sub oixctrl1_OptionChange(ByVal lOptionId As Long)
   Rem ** This event handler writes all of the options out to an
   Rem    .ini file as they are changed **

   Select Case lOptionId
       Case SCCID_DIALOGFLAGS:
          bSuccess = OSWritePrivateProfileString("Options",
              "DialogFlags", oixctrl1.DialogFlags, "d:\sample.ini")

   ' other case statements omitted for brevity
   End Select
End Sub
```

The following images are the default dialog boxes:

Figure 5 Clipboard Options Dialogs

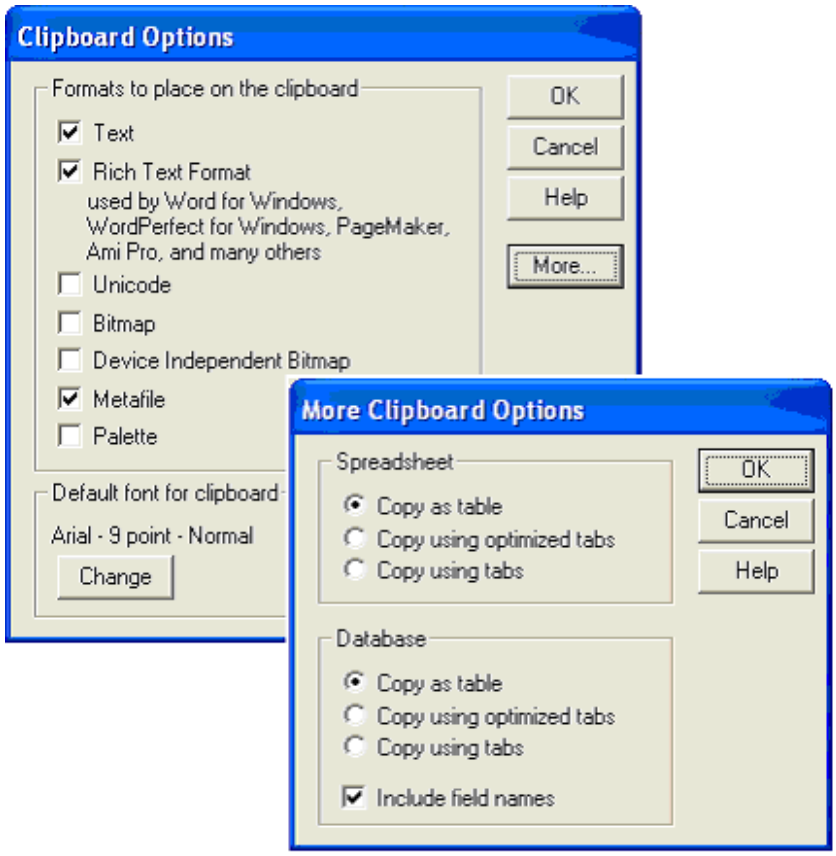


Figure 6 Display Options Dialogs

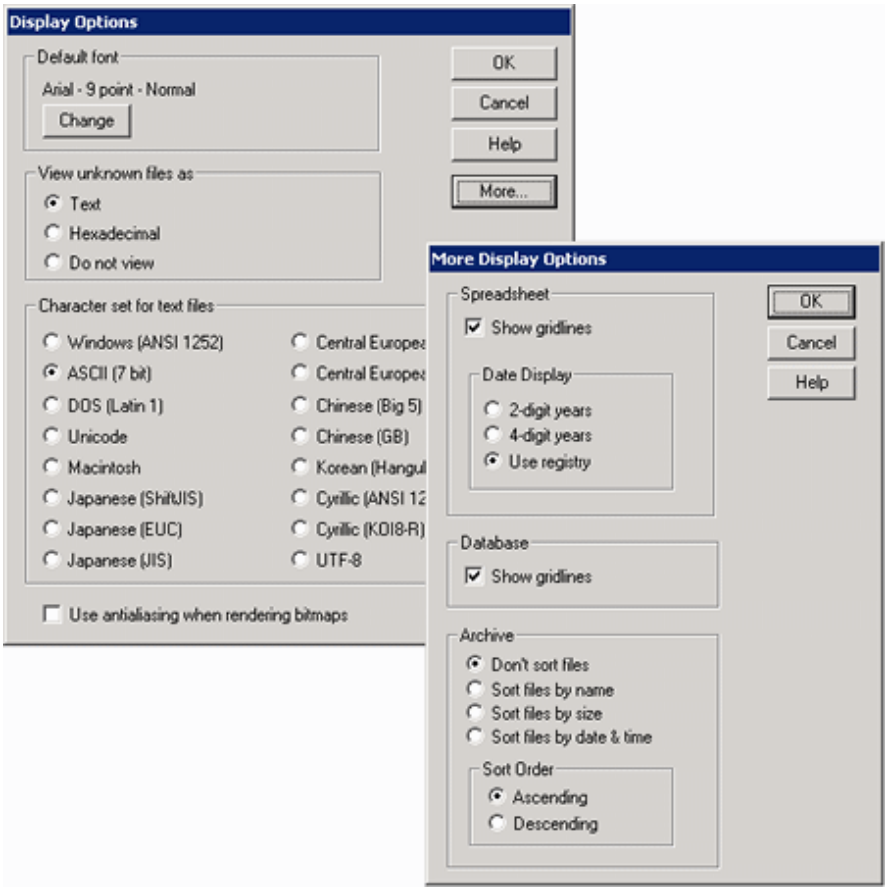
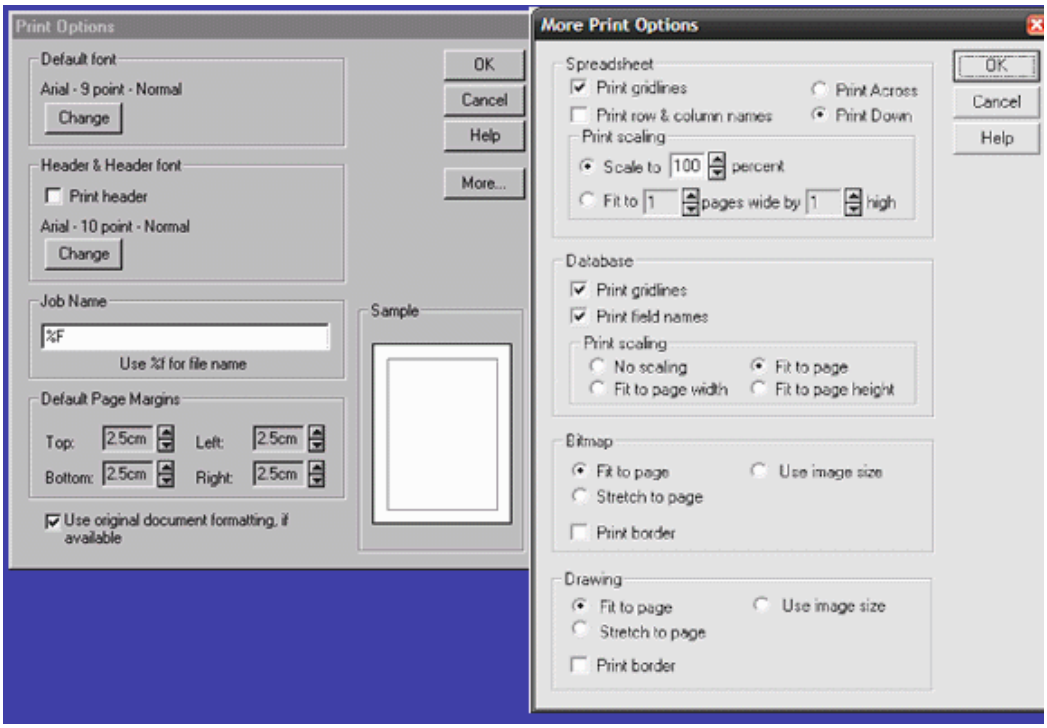


Figure 7 Print Options Dialogs

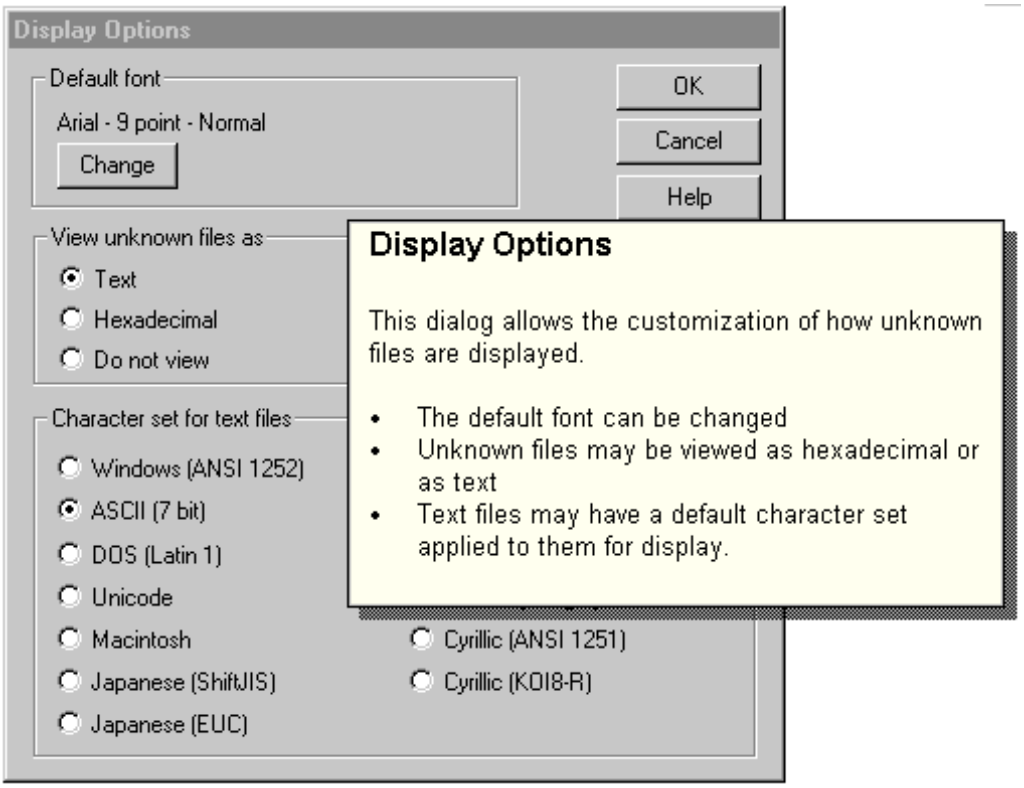


Each of the Display Options, Clipboard Options, Print Options and Font Selection dialogs has a help button to provide the user with additional information regarding the options presented. Each time a help button is pressed, the application will receive a DoHelp event (see DoHelp) indicating which dialog box the user currently has displayed. Because there is no default help file for the viewer, the developer should either display customized help, or remove the help button from the default dialogs to prevent the event from occurring.

```
Private Sub oixctrl1_DoHelp(ByVal iDialog As Integer)
   Dim l As Long
   l = iDialog
   nRet = OSWinHelp(Form1.hwnd, "sample.hlp", 8, l)
End Sub
```

The Help and More buttons may be removed from the default dialog boxes by setting the DialogFlags property (see DialogFlags). This property also controls which menu items appear on the default context menus. The following values may be OR-ed together to change default appearance of the menus and dialog boxes.

Figure 8 Display Options Dialog



Dialog box flags:

- SCCVW_DIALOG_NOHELP: Remove the help button.
- SCCVW_DIALOG_NOMORE: Remove the more button.

Menu Flags:

- SCCVW_DIALOG_NOADDOPTIONSTOMENU: Remove the Options submenu.
- SCCVW_DIALOG_NOADDDISPLAYTOMENU: Remove the Display Options from the Options submenu.

- SCCVW_DIALOG_NOADDDISPLAYTOMENU: Remove the Print Options from the Options submenu.
- SCCVW_DIALOG_NOADDCLIPBOARDTOMENU: Remove the Clipboard Options from the Options submenu.
- SCCVW_DIALOG_NOADDDOPRINTTOMENU: Remove the Print menu item.
- SCCVW_DIALOG_NOADDDOCOPYTOMENU: Remove the Copy menu item

Outside In has been modularized to allow for internationalization and/or customization of the default dialog boxes and string resources. By default, the name of the dialog resource library is SCCLO.DLL. Alternate resource libraries may be specified programmatically at runtime by setting the ResourceLibraryID property (see ResourceLibraryID). All resource libraries must conform to the naming convention SCCLO??.DLL. The ResourceLibraryID property would be set to access a different set of resources.

```
Private Sub DspOptions_Menu_Click()
   DspOptions_Menu.Checked = Not DspOptions_Menu.Checked
   ' Check to see if all three checks are off, if so remove
   ' the entire options menu

   If (Not ClpOptions_Menu.Checked And Not
           PrtOptions_Menu.Checked And Not
           DspOptions_Menu.Checked) Then
      CustomShowDialog 0, -4
   Else
      CustomShowDialog 0, 4
   End If
      If DspOptions_Menu.Checked = False Then
         CustomShowDialog 0, -1
   Else
      CustomShowDialog 0, 1
   End If
End Sub

Private Sub CustomShowDialog(ByVal iDialog, ByVal iRemoveOption)
   Rem ** Display the dialog based on the parameter **
   Select Case iDialog
      Case 1-3: ' shown in previous example
      Case 0: Select Case iRemoveOption
         Case -1:
            oixctrl1.DialogFlags =
            oixctrl1.DialogFlags Or SCCVW_DIALOG_NOADDDISPLAYTOMENU ' remove
display
                                       ' options from menu

         Case -2:
            oixctrl1.DialogFlags =
            oixctrl1.DialogFlags Or SCCVW_DIALOG_NOADDPRINTTOMENU ' remove
print options
                                       ' from menu
         Case -3:
            oixctrl1.DialogFlags =
            oixctrl1.DialogFlags Or SCCVW_DIALOG_NOADDCLIPBOARDTOMENU   '
remove clipboard
                                         ' options from menu
         Case -4:
            oixctrl1.DialogFlags =
            oixctrl1.DialogFlags Or SCCVW_DIALOG_NOADDOPTIONSTOMENU    ' remove
entire
                                       ' options menu
         Case 1:
            oixctrl1.DialogFlags =
            oixctrl1.DialogFlags And Not SCCVW_DIALOG_NOADDDISPLAYTOMENU    '
remove flag
```

```
         Case 2:
            oixctrl1.DialogFlags =
            oixctrl1.DialogFlags And Not SCCVW_DIALOG_NOADDPRINTTOMENU    '
remove flag
         Case 3:
            oixctrl1.DialogFlags =
            oixctrl1.DialogFlags And Not SCCVW_DIALOG_NOADDCLIPBOARDTOMENU    '
remove flag
         Case 4:
            oixctrl1.DialogFlags =
            oixctrl1.DialogFlags And Not SCCVW_DIALOG_NOADDOPTIONSTOMENU     '
remove flag
         End Select
   End Select
EndSub
```

Scroll Bars

The Outside In viewer, by default, displays horizontal and vertical scroll bars within the viewing window. These scroll bars allow the user to page through the file as needed. If the developer, however, needs to create and control scroll bars external to the ActiveX component, the default viewer scroll bars may be hidden by setting the HScrollbar and VScrollbar properties (see HScrollbar, and VScrollbar) to FALSE.

As the currently viewed document is read into the viewer, the scrollbar thumb size will change to indicate the amount of viewed text relative to the entire document size. The maximum and minimum values will also change to reflect the actual size of the document (or at least the amount that has been read by the viewer). The developer can coordinate the external scrollbars by reacting to the HScrollPageSize and VScrollPageSize events (see HScrollPageSize, and VScrollPageSize) to set the thumb size and the HScrollRange and VScrollRange events (see HScrollRange, and VScrollRange) to set the scrollbar minimum and maximum values. Because reading the document is a background process, the developer is likely to see many of these events generated before the entire document size is known. The values that are passed to the HScrollRange and VScrollRange event handlers are abstract and relative to the type of file being viewed and consequently the display engine in use. For example, for a bitmap image the maximum values will be the number of pixels in the X/Y directions.

When the document being viewed completely fits within the viewing window, the scrollbars will be disabled and the VScrollState and HScrollState events (see VScrollState, and HScrollState) will be generated to allow the external scrollbars to be disabled or hidden.

Whenever the user scrolls through the document either using the scroll bars or the page up/down keys, HScrollPosition and VScrollPostion events will be generated to allow the external scroll bars to be repositioned. The position values passed to the event handler will be relative to the minimum and maximum values for the scrollbar and will vary in units based on the type of file being viewed.

```
REM ** Two scrollbars external to the Outside In viewer
REM    Hscroll_Ctrl and Vscroll_Ctrl are used to demonstrate
REM    the use of these event handlers. **

Private Sub oixctrl1_HScrollState(ByVal bEnabled As Long)
   If bEnabled = 1 Then
      HScroll_Ctrl.Visible = True
   Else
      HScroll_Ctrl.Visible = False
   End If
End Sub
```

```
Private Sub oixctrl1_VScrollState(ByVal bEnabled As Long)
   If bEnabled = 1 Then
      VScroll_Ctrl.Visible = True
   Else
      VScroll_Ctrl.Visible = False
   End If
End Sub

Private Sub oixctrl1_HScrollPageSize(ByVal lPageSize As Long)
   HScroll_Ctrl.LargeChange = lPageSize
End Sub

Private Sub oixctrl1_VScrollPageSize(ByVal lPageSize As Long)
   VScroll_Ctrl.LargeChange = lPageSize
End Sub

Private Sub oixctrl1_HScrollRange(ByVal lMin As Long, ByVal lMax
   As Long)
   HScroll_Ctrl.Min = lMin
   HScroll_Ctrl.Max = lMax - HScroll_Ctrl.LargeChange
End Sub

Private Sub oixctrl1_VScrollRange(ByVal lMin As Long, ByVal lMax
   As Long)
   VScroll_Ctrl.Min = lMin
   VScroll_Ctrl.Max = lMax - VScroll_Ctrl.LargeChange
End Sub

Private Sub oixctrl1_VScrollPosition(ByVal lPosition As Long)
   VScroll_Ctrl.Value = lPosition
   End Sub

Private Sub oixctrl1_HScrollPosition(ByVal lPosition As Long)
   HScroll_Ctrl.Value = lPosition
   End Sub
```

To facilitate communication from the external scroll bars to the viewer, the methods HScroll and VScroll (see HScroll, and VScroll) may be invoked to scroll the viewed document programmatically. These methods take two parameters: the type of scroll (line up, line down, page up, page down or absolute position) and the position relative to the scroll bar's minimum and maximum values. Note the term 'position' is not related to the OixPos object that is used in searching, annotations, and text selection.

```
Private Sub HScroll_Ctrl_Change()
   oixctrl1.HScroll SCCSB_POSITION, HScroll_Ctrl.Value
   'SCCSB_POSITION Const = 7
End Sub

Private Sub VScroll_Ctrl_Change()
   oixctrl1.VScroll SCCSB_POSITION, VScroll_Ctrl.Value
End Sub
```

Miscellaneous

The Keydown event (see Keydown) is generated each time the user presses a key on the keyboard while the ActiveX control has focus. Each keystroke is passed as a virtual keycode to the event handler so that developer can define customized actions for specific keys.

```
Private Sub oixctrl2_Keydown(ByVal lVKey As Long)
   Rem ** If ESC is pressed, make the primary viewer oixctrl1
   Rem    visible **

   If (lVKey = vbKeyEscape) Then
      oixctrl2.Visible = False
      oixctrl1.Visible = True
```

```
   End If
End Sub
End Sub
```

The IdleBitmap method (see IdleBitmap) overrides the default empty viewer splash screen. The method takes two parameters: the module (DLL) instance returned from Windows API routine LoadModule and the resource ID of the bitmap stored in the module. The following example shows how to accomplish this from within Visual Basic. The sample.rc file is compiled with the standard rc.exe shipped with Visual C++. The resulting sample.res is added to the Visual Basic project using the Project/Add File menu option.

The bitmap will only display when the application is built as an .exe.

```
-----begin sample.rc-----
#include <windows.h>

1000 BITMAP oi.bmp
-----end sample.rc-----

Private Sub Form_Load()
       ' other initialization done here
   oixctrl1.IdleBitmap App.hInstance, 1000
End Sub
```

To change the magnification level of a bitmap or vector image, the ImgZoom and ImgShowFullScreen methods (see ImgZoom, and ImgShowFullScreen) may be called. ImgZoom can scale the image to a given percent level, zoom in/out, zoom to the current selection or restore the image to its original state. Each time the image is zoomed in or out, the magnification level will double or half, respectively. The ImgShowFullScreen method causes the viewer to take over the entire screen and display the image. The viewer will continue to display the image full screen until the user presses the ESC key or ImgShowFullScreen is invoked with a parameter of FALSE.

```
Private Sub oixctrl1_Keydown(ByVal lVKey As Long)

   Rem ** Handle special Ctrl-key combinations to zoom on an
   Rem    image. **

   If (lVKey And vbCtrlMask) > 0 Then
   Select Case lVKey And Not vbCtrlMask
   Case vbKeyAdd:      oixctrl1.IMGZoom 1           ' Zoom
                                                    ' In
   Case vbKeySubtract: oixctrl1.IMGZoom 2           ' Zoom
                                                    ' Out
   Case vbKeyMultiply: oixctrl1.IMGShowFullScreen 1 ' Zoom
                                                    ' Full
                                                    ' screen
    Case vbKey1:  oixctrl1.IMGXZoomPercent = 100    ' Zoom to
                                                    ' 100%
                  oixctrl1.IMGYZoomPercent = 100
                  oixctrl1.IMGZoom 0
    Case vbKey2:  oixctrl1.IMGXZoomPercent = 200   ' Zoom to
                                                   ' 200%
                  oixctrl1.IMGYZoomPercent = 200
                  oixctrl1.IMGZoom 0
    Case vbKey3:  oixctrl1.IMGXZoomPercent = 300   ' Zoom to
                                                   ' 300%
                  oixctrl1.IMGYZoomPercent = 300
                  oixctrl1.IMGZoom 0
    Case vbKey4:  oixctrl1.IMGXZoomPercent = 400   ' Zoom to
                                                   ' 400%
                  oixctrl1.IMGYZoomPercent = 400
                  oixctrl1.IMGZoom
   End Select
```

```
   End If
End Sub
```

When an archive file is displayed, the individual files contained within the document may be extracted by calling the ArchiveSave method (see ArchiveSave). This method can save all of the files in the document or just the selected files. A directory selection dialog box will be displayed for specification of the output location.

```
Private Sub SaveArchive_Menu_Click()
   oixctrl1.ArchiveSave 1 ' Save selected archive
                          ' file 0 = save all files
End Sub
```

Display Engine Options

The following property is available to customize the units for page margins.

- IntlFlags: TRUE. Use English units for page margins, otherwise use metric unit for page margins.

The following properties control how vector and bitmap images are displayed, printed and scaled in the viewer.

- ImgXZoomPercent, ImgYZoomPercent: When the ImgZoom method is called with a zoom type of 0, these properties indicate how much to zoom. For example, a *ZoomPercent of 200 will double the image size each time ImgZoom is invoked.
- BmpFitMode: When a bitmap is viewed this parameter determines how it displayed relative to the viewing window.
  - 1: Bitmap is shown actual size.
  - 2: Bitmap will be stretched to fit within the view window maintaining its aspect ratio. Depending on the window's aspect ratio and the bitmap's aspect ratio, extra space can appear in either the horizontal or vertical direction.
  - 3: Fit the bitmap into the window's height. Depending on the width of the window relative to the bitmap, the full width of the image may not fit within the viewer.
  - 4: Fit the bitmap into the window's width. Depending on the height of the window relative to the bitmap, the full height of the image may not fit within the viewer.
- BmpPrintAspect: When printing a bitmap, this property directs the printer to do one of the following:
  - 1: Stretch the image to fit within the print margins while still maintaining the bitmap's aspect ratio.
  - 2: Stretch the image to completely fill the area within the print margins. Aspect ratio will not be maintained.
  - 3: Use the size as specified by the image.
- BmpPrintBorder: TRUE. Draw a one-pixel border around the image when printing, otherwise no border is drawn.
- BmpDither: TRUE. Dither the bitmap on low resolution displays to more accurately represent the image, otherwise, do not dither the image.

- BmpDitherAvailable: This read-only property indicates whether the current image can be dithered.
- BmpRotation When displaying a bitmap, the following property values determine it's orientation:
  - 0: No rotation
  - 90: Bitmaps are rotated 90 clockwise
  - 180: Bitmaps are rotated 180 clockwise or are upside down.
  - 270: Bitmaps are rotated 270 clockwise.
- VecFitMode: Fit the vector drawing into the window disregarding the original image aspect ratio.
- VecPrintAspect: When printing vector images the property direct the printer to do one of the following actions:
  - 1: Stretch the image to fit within the print margins while still maintaining the vector image aspect ratio.
  - 2: Stretch the image to completely fill the area within the print margins. Aspect ratio will NOT be maintained.
  - 3: Use the size as specified by the drawing.
- VecPrintBorder: TRUE. Draw a one-pixel border around the image when printing, otherwise no border is drawn.
- VecPrintBackground: TRUE. If a vector image has a background defined, then it will be printed, otherwise no background will be printed.
- VecShowBackground: TRUE. If a vector image has a background defined, then it will be displayed in the viewer, otherwise backgrounds will not be displayed for vector images.

When the Word Processor display engine is loaded, the viewed document will be displayed according to the following properties.

- WPDisplayMode: The document will be displayed in one of the following modes. The default value is 2 - Normal.
  - 1: Draft mode. The document will be displayed using a single font as specified in the DisplayFont property (see DisplayFont). Embedded graphics and borders will not be displayed and all text will be wrapped to the size of the view window.
  - 2: Normal mode. The text will be wrapped to the size of the view window and all supported formatting will be applied.
  - 3: Preview mode. All supported formatted will be applied and the text will be wrapped as it will be printed.
- WPFitMode: When displaying the document in preview mode (see WPDisplayMode) this property controls how the preview will be scaled.
  - 1: Original. Show as actual size.

    - 2: Window. Fit preview within the window.
    - 3: Window Width. Fit preview within the width of the window. Not all of the page may be visible in the vertical direction.
- WPWrapToWindow: When displaying the document in draft or normal mode (see WPDisplayMode) this property determines whether or not the text is wrapped to the size of the viewer window.
    - TRUE: Text will be wrapped in draft and normal mode, otherwise text will not be wrapped, but may be viewed by scrolling the window to the right using the horizontal scroll bar.

The archive display engine has these properties.

- ArchiveSortOrder: When files are listed in the archive display engine, they will be sorted according to this property value:
    - 1: None. Display files in the order in which they appear in the archive.
    - 2: Name. Sort files by filename.
    - 3: Size. Sort files by file size.
    - 4: Date. Sort file by creation date.
- ArchiveSortDescending: TRUE. Reverses the order of ArchiveSortOrder (see ArchiveSortOrder); otherwise, files are sorted ascending according to ArchiveSortOrder.

The Database display engine displays, prints and copies to the clipboard document information according to the following properties.

- DBClipboard: Specifies the format of the clipboard data.
    - 1: Table. For RTF and AMI clipboard formats the data will be placed on the clipboard as a table.
    - 2: Optimized Tabs. Each field will be separated by a TAB character, except when a field is empty.
    - 3: Tabs. Each field will be delimited by a TAB character.
- DBFieldNamesToClip: TRUE. Field names will be copied to the clipboard; otherwise, field names are not copied.
- DBDraftMode: When displaying database data this property turns formatting on and off.
    - TRUE: Limited formatting is displayed; otherwise, all supported formatting is displayed
- DBShowGridLines: When displaying database data this property turns the background grid on/off.
    - TRUE: A dotted line grid is displayed; otherwise, grid lines are not displayed.
- DBPrintFit: Specifies how the database data will be fitted to the printed page.
    - 1: None. The data will be printed actual size on as many pages as is needed.

  - 2: Page. The data will be scaled to completely fit on one page.
  - 3: Width. The data will be scaled to fill the width of the page.
  - 4: Height. The data will be scaled to fill the height of the page.
- DBPrintGridLines: TRUE: Grid lines will be printed; otherwise, grid lines will not be printed.
- DBPrintHeadings: TRUE: Field names will be printed along with the data; otherwise, field names are omitted from the printed data.

When displaying spreadsheet documents, the properties in the following list control the clipboard data, the display and print formatting.

- SSClipboard: Specifies the format of the clipboard data.
  - 0: Table. For RTF and AMI clipboard formats the data will be placed on the clipboard as a table.
  - 1: Optimized Tabs. Each cell will be separated by a tab character, except when a cell is empty.
  - 2: Tabs. Each cell will be delimited by a tab character.
- SSRowColNamesToClip: TRUE: Row and Column headers will be copied to the clipboard; otherwise, headers are not copied.
- SSDraftMode: TRUE: Limited formatting is displayed; otherwise. all supported formatting is displayed.
- SSShowGridLines: TRUE: A dotted line grid is displayed; otherwise, grid lines are not displayed.
- SSPrintFit: Specifies how the spreadsheet will be fitted to the printed page.
  - 0: None. The spreadsheet will be printed actual size on as many pages as is needed.
  - 1: Page. The spreadsheet will be scaled to fit on one page.
  - 2: Width. The spreadsheet will be scaled to fill the width of the page.
  - 3: Height. The spreadsheet will be scaled to fill the height of the page.
  - 4: Scale. The spreadsheet will be scaled according to the SSPrintScalePercent property (see SSPrintScalePercent).
  - 5: FitToPages. The spreadsheet will be scaled to fit on SSPrintScaleXHigh and SSPrintScaleXWide pages (see SSPrintScaleXHigh, and SSPrintScaleXWide).
- SSPrintGridLines: TRUE: Grid lines will be printed; otherwise, grid lines will not be printed
- SSPrintHeadings: TRUE: Row and Column headings will be printed along with the data; otherwise, headings are omitted from the printed data.
- SSPrintScalePercent: Specifies the scale factor to be use when the SSPrintFit property (see SSPrintFit) is set to 4. This is a percentage of actual size.

- SSPrintScaleXHigh: Specifies how many vertical pages the printed spreadsheet is to fit within. This is used when the SSPrintFit property is set to 5.
- SSPrintScaleXWide: Specifies how many horizontal pages the printed spreadsheet is to fit within. This is used only when the SSPrintFit property is set to 5.
- SSPrintDirection: Specifies orientation:
  - 0: Print spreadsheet across and then down.
  - 1: Print spreadsheet down and then across.

5

Searching Documents

This chapter provides tips for searching documents.

This chapter includes the following sections:

Searching

Text search can be initiated by the developer by calling the Search method (see Search). This method's parameters specify search direction, case sensitivity, search starting location and can optionally display a default search dialog for search-text entry. If the text is found, this method returns 0. If an error occurred, search returns a -1, otherwise a return value of 1 indicates that the EOF was reached before the text was found. If the search is successful, the located text will be highlighted and scrolled into the viewing window. All properties related to text selection will be updated as well as the caret position which will be located at the beginning of the highlighted text.

SearchNext (see SearchNext) will continue (in either the same or a different direction) to look for the next occurrence of the text specified in the previous Search method. The return values and behavior are the same as for the Search method.

```
Private Sub Search_Ctrl_Click()
   'Dim ret As Integer
   ' If the search text has changed, call the Search method,
   ' otherwise call SearchNext

   If LastSearchText = SearchText_Ctrl.Text Then
      ret = oixctrl1.SearchNext(SCCVW_SEARCHFORWARD) ' search forward
   Else
   Rem ** Search for text from the SearchText edit box control
   '      with no dialog, case sensitivity
   '      from the current caret position and search forward
      ret = oixctrl1.Search(OIX_FALSE, SearchText_Ctrl.Text, SCCVW_SEARCHCASE,
SCCVW_SEARCHCURRENT, SCCVW_SEARCHFORWARD)
   End If

   If ret = 0 Then
      Search_Ctrl.Caption = "Search Next"   ' change the
                                            ' search button
                                            ' to a search
                                            ' next
      LastSearchText = SearchText_Ctrl.Text ' save off the
                                            ' text used in
                                            ' the search
   Else
      If ret = 1 Then ' EOF reached. Display message and
```

```
                    ' start over
      MsgBox "End of File Reached - wrapping to beginning of document"
      ret = oixctrl1.Search(OIX_FALSE, SearchText_Ctrl.Text, SCCVW_SEARCHCASE,
SCCVW_SEARCHTOP, SCCVW_SEARCHFORWARD)
      End If
   End If
End Sub

Private Sub SearchText_Ctrl_Change()
  LastSearchText = ""             ' When the text in the
                                  ' search text edit
  Search_Ctrl.Caption = "Search" ' box changes, reset the
                                  ' button caption to "Search"
End Sub
```

Positions

The ActiveX control uses the concept of a position to specify locations within a file. Each position is a placeholder or bookmark into the currently viewed document. It is important to understand how positions are manipulated to be able to use some of the more powerful features of the Outside In control. For example, a number of the annotation related methods/properties work with positions to specify the location in the file for the annotation.

The position variable is actually a COM object that is referred to by both Visual Basic and Visual C++ as an OixPos, and is easily manipulated in both environments. When working with an OixPos, the following rule of thumb should be observed. When passing an OixPos to a method, pass it as an object (or by value). That is, always create a NEW OixPos before sending it to a method. However, OixPos properties will hold a reference to a OixPos object.

For the currently viewed document there is always a caret position defined. The caret position is the location on the screen where the cursor is displayed and is stored relative to the beginning of the file. The caret position can only be manipulated by the user through the use of the mouse and/or keyboard and may appear anywhere within the displayed text of the document.

There are four methods available to set an OixPos object/variable:

1. SetPositionToCurrent (see SetPositionToCurrent) sets the position object to the location of the caret position.

2. If the user has selected an area of text, SetPositionToSelection (see SetPositionToSelection) sets two parameters to the starting and ending positions of the currently selected text.

3. You can also set the position directly using the SetrActualCount (see SetActualCount) method. This method sets a position relative to the number of characters from the beginning of the document. Because different document types have different notions of page layout, this method is mostly used with word-processing documents. For example, the number of characters into a spreadsheet and a presentation can take on completely different meanings.

4. The last method to set an OixPos object is FindPosition (see FindPosition). FindPosition takes a starting position and a technique for locating the resulting OixPos object. The found position can be any of the following:

   - 1: First position in the current page (starting position is ignored)
   - 2: Last position in the current page (starting position is ignored)

- 3: The position just to the left of the starting position
- 4: The position just to the right of the starting position
- 5: Start of the current selection (starting position is ignored)
- 6: End of the current selection (starting position is ignored)
- 7: Location at the top/left of the viewing window (starting position) is ignored)
- 8: Location at the bottom/right of the viewing window (starting position is ignored)
- 9: Beginning of the line which contains the starting position
- 10: End of the line which contains the ending position
- 11: Beginning of the previous line (relative to the starting position)
- 12: End of the previous line (relative to the starting position)
- 13: Beginning of the word to the left of the starting position
- 14: Beginning of the next word to the right of the starting position
- 15: Beginning of the section to the left of the starting position
- 16: Beginning of the next section to the right of the starting position

Four methods can be used to manipulate or access the position objects.

CopyPosition and ComparePositions (see CopyPosition, and ComparePositions) both take two OixPos objects as parameters and either copy the location information from one to the other or return which one is closer to the beginning of the file. GetActualCount (see GetActualCount) is the counterpart to the SetActualCount (see SetActualCount) method and returns the number of characters from the beginning of the file specified by the OixPos object.

The DisplayPosition (see DisplayPosition) method will redraw the currently viewed document relative to the OixPos parameter. The following flags determine where the position will be located relative to the viewing window.

- 1: Top of the view window
- 2: Middle of the view window
- 3: Bottom of the view window

The following example illustrates the use of these methods.

```
Private Sub Annotate_Ctrl_Click()
   Dim oixStart As New OixPos
   Dim oixEnd As New OixPos

   Rem ** Check to see if any text is selected. If so, set
   Rem    oixStart and oixEnd to the selected text. If not,
   Rem    set oixStart to the current caret position
   Rem    and oixEnd to the beginning of the next word. **

   If Selection_Ctrl.Caption <= 0 Then
      oixctrl1.SetPositionToCurrent oixStart
      oixctrl1.FindPosition oixStart, oixEnd, 14 ' set the end
                                                 ' position to
```

```
                                                ' be at the
                                                ' beginning of
                                                ' the next word
   oixctrl1.FindPosition oixEnd, oixStart, 13   ' set the
                                                ' beginning
                                                ' position of
                                                ' the "current
                                                ' word" word
   oixctrl1.SelectionAnchor = oixStart
      oixctrl1.SelectionEnd = oixEnd
   Else
       oixctrl1.SetPositionToSelection oixStart, oixEnd
       ' Swap the start and end positions if oixEnd is closer to
       ' the top of the document than oixStart

       If oixctrl1.ComparePositions(oixStart, oixEnd) 0 Then
         Dim oixTemp As New OixPos
         oixctrl1.CopyPosition oixTemp, oixStart
         oixctrl1.CopyPosition oixStart, oixEnd
         oixctrl1.CopyPosition oixEnd, oixTemp
      End If
   End If

   AnnotateSelection oixStart, oixEnd
   oixctrl1.DisplayPosition oixStart, 1   ' reposition text near
                                          ' the top of viewer

End Sub
```

Annotations

Outside In provides a powerful way to bookmark or annotate selected areas of text and locate previously defined annotations, bookmarks or URLs. Annotations, document-defined bookmarks and URLs are treated as individual annotation types. Document defined bookmarks and URLs are automatically annotated based on the information present in the file being processed. Annotations are user-defined and must be created programmatically using the Annotation API. There are three types of annotations: hilited (text is highlighted), hidden (text is hidden), and picture (a picture is inserted into the document).

To hilite and/or annotate a block of text, two OixPos objects are needed: the starting and ending position. These can be obtained by retrieving the user's current text selection, by using position methods to locate text, or by using the Search method (see Search). Given two position objects, the text can be hilited using the AddAnnotationHilite method, or can be hidden using the AddAnnotationHideText method (see AddAnnotationHilite, and AddAnnotationHideText).

Another way to add an annotation is to insert a picture as an annotation. This type of annotation will be inserted at the designated OixPos position when the AddAnnotationPicture method (see AddAnnotationPicture) is called with a picture object as a parameter. This picture will be displayed inline as the document is viewed.

```
Private Sub AnnotateSelection(lType As Integer, Optional ByRef
   OixStart As OixPos, Optional ByRef oixEnd As OixPos)

REM ** Create an annotation association with the selection
REM    between oixStart and oixEnd based on the type requested
REM    in the lType parameter. **

Dim Text As String
Select Case lType
   Case 0: ' Hilite
      Text = GetAnnotation(oixStart, annotateid)
      oixctrl1.AddAnnotationHilite annotateid, 1,
         SCCVW_ANNOTATION_SCLICK, 0, 0, Text, oixStart, oixEnd
```

```
   Case 1: ' Hide Text
   oixctrl1.AddAnnotationHideText annotateid, 0, oixStart,
      oixEnd
   Case 2: ' Picture
      Dim P As New StdPicture
      CommonDialog1.ShowOpen
      Set P = LoadPicture( CommonDialog1.Filename )
      oixctrl1.AddAnnotationPicture annotateid, P, 2, 0,
        oixStart

   End Select

annotateid = annotateid + 1
End Sub

Private Function GetAnnotation(ByRef Oix As OixPos, annotate As
   Integer) As String
   Rem ** AnnotationText_Form is a form with a Title text field,
   Rem    Annotation Text control and a message text control
   Rem    which is "popped-up" with a Show method to collect
   Rem    annotation text. **

   AnnotationText_Form.Left = oixctrl1.Left + 0.25 *
      oixctrl1.Width + Form1.Left
   AnnotationText_Form.Top = oixctrl1.Top + 0.4 * oixctrl1.Height
      + Form1.Top
   AnnotationText_Form.Title.Text = "Annotation #: " +
      Str(annotate)
   AnnotationText_Form.AnnotationText_Ctrl.Text = ""
   AnnotationText_Form.Show 1, Form1
   GetAnnotation = AnnotationText_Form.AnnotationText_Ctrl.Text
End Function
```

When adding an annotation, one of the parameters to the AddAnnotationHilite method (see AddAnnotationHilite) describes the user-interaction that will trigger an AnnotationEvent event (see AnnotationEvent). These include: single-click, double-click, and cursor-over. When one of these actions occurs, the action type and annotation information is passed to the AnnotationEvent event handler. The developer can then use this information to display additional annotation information.

Another parameter used when adding an annotation is an annotation style. Annotation styles are defined programmatically using the HiliteStyle method (see HiliteStyle). Styles must be defined before annotations are added. Each style is uniquely identified with an ID and define the way the viewer hilites the annotated text; foreground color, background color and font attributes can all be defined. Once defined, the unique style identifier can be passed as an argument to the AddAnnotationHilite method (see AddAnnotationHilite). Additionally, once a style has been defined and associated with an ID, the ID can not be reused or re-assigned until the currently loaded file changes.

```
Private Sub Form_Load()
   Dim fg As OLE_COLOR
   Dim bg As OLE_COLOR
   Dim ret As Boolean

   fg = RGB(255, 255, 255)
   bg = RGB(128, 128, 128)

   oixctrl1.HiliteStyle 1, 3, fg, bg, 0
End Sub

Private Sub oixctrl1_AnnotationEvent(ByVal lEvent As Long,
                                     ByVal varData As Variant)
   Rem ** If the user single clicks on the annotation, popup the
   Rem    annotation form **
```

```
   If lEvent And SCCVW_ANNOTATION_SCLICK Then
      ShowAnnotation oixctrl1.AnnotationDataType, lId, varData
   End If

End Sub

Private Sub ShowAnnotation(lType As Long, annotate As Long, data
   As Variant)
   Rem ** Display annotation text - called from AnnotationEvent
   Rem    and the AnnotationList_Ctrl_Click event **

   Select Case lType
      Case 0: ' User annotation
      AnnotationText_Form.Title.Text = "Annotation #: " +
         Str(annotate) + "(" +
                                       Hex(annotate) + ")"
      Case 1: ' URL
      AnnotationText_Form.Title.Text = "URL: " + Str(annotate) +
         "("
                                       + Hex(annotate) + ")"
      Case 2: ' Bookmark
      AnnotationText_Form.Title.Text = "Bookmark: " +
         Str(annotate) + "("
                                       + Hex(annotate) +")"
   End Select

   AnnotationText_Form.AnnotationPict.Visible = False
   AnnotationText_Form.AnnotationText_Ctrl = True
   AnnotationText_Form.AnnotationText_Ctrl.Text = data
   AnnotationText_Form.Left = oixctrl1.Left + 0.25 *
      oixctrl1.Width + Form1.Left
   AnnotationText_Form.Top = oixctrl1.Top + 0.4 * oixctrl1.Height
      + Form1.Top
   AnnotationPicture_Form.Show 0, Form1
End Sub
```

Annotations can be manipulated using the ClearAnnotations, FindAnnotation, and GoToAnnotation methods (see ClearAnnotations, FindAnnotation, and GoToAnnotation). Each of these methods take a parameter which allows the developer to select multiple annotations using an ID mask. An annotation will match if the logical "and" of the annotation ID and the ID mask is equal to the ID mask. For example, if the ID mask is 225 (11100001), the annotation ID of 227 (11100011) would match, however, the annotation ID of 226 (11100010) would not. Calling the ClearAnnotations method (see ClearAnnotations) will remove all matching annotations.

Locating existing annotations is accomplished using the FindAnnotation and GoToAnnotation methods (see FindAnnotation and GoToAnnotation). Both methods locate user-defined annotations, document-defined bookmarks and URLs. Also, these methods can start from the beginning/end or look for the next/previous annotation and both utilize matching via a mask ID as described in the preceding paragraph. The main difference between the two methods is while FindAnnotation will update the read-only properties, it will not update the view window to bring the annotation into view. GoToAnnotation will update the view window to display the matched annotation, however, it will not update the read-only property values. This can be done manually by calling the GetAnnotationData method (see GetAnnotationData) with the annotation ID and type (annotation, bookmark or URL).

One final method is provided to copy the OixPos position of the last found annotation to the caret position. If the AnnotationSetPos method (see AnnotationSetPos) is passed a TRUE value, the text associated with the annotation is selected, otherwise, the caret position is just moved to the beginning of the annotation and the view updated.

The Annotation methods and the Annotation event populate the following read-only properties. AnnotationId contains the last used annotation ID (see AnnotationId); AnnotationStartPos and AnnotationEndPos store the OixPos objects for the last used annotation (see AnnotationStartPos and AnnotationEndPos). Each annotation can have additional data that is associated with it. The data and its type for the last used annotation are stored in the AnnotationData and AnnotationData read-only properties (see AnnotationData). For document-defined bookmarks and URLs, the Annotation property should be interpreted as a BSTR object.

It should be noted that all annotations are volatile. Once a new file is loaded, the annotation information is cleared. Therefore, the developer should provide a mechanism to save and restore annotations each time a document is viewed.

```
Private Sub ListAnnotations_Click()
   Rem ** Use the FindAnnotation method to enumerate through the
   Rem    document and populate the AnnotationList_Ctrl list
   Rem    box. **

   Dim currentOix As New OixPos
   Dim Text As String

   oixctrl1.SetActualCount currentOix, 0   ' initialize
                                           ' currentOix to top
                                           ' of file

   While (oixctrl1.FindAnnotation(3, 0, 0, currentOix))
      oixctrl1.CopyPosition currentOix,
        oixctrl1.AnnotationEndPos
      Select case oixctrl1.AnnotationDataType
         Case 0:  ' Annotation
                  Text = "AN:"
         Case 1:  ' URL
                  Text = "URL:"
         Case 2:  ' Bookmark
                  Text = "BM:"
      End Select

      AnnotationListBox_Ctrl.AddItem Text +
         Str(oixctrl1.AnnotationId)
      AnnotationListBox_Ctrl.ItemData
         (AnnotationListBox_Ctrl.NewIndex) =
            oixctrl1.AnnotationId
   Wend

End Sub


Private Sub AnnotationListBox_Ctrl_Click()
   Rem ** On a single click of the annotation list box, scroll
   Rem    the document to the selected annotation. **

   If (oixctrl1.GotoAnnotation(0, 1,
      AnnotationListBox_Ctrl.ItemData
      (AnnotationListBox_Ctrl.ListIndex)) = 0)
   Then
      MsgBox "Annotation not found"
   Else
      oixctrl1.AnnotationSetPos 0  ' set the caret position
                                   ' to the annotation
      oixctrl1.SetFocus            ' reset the focus to the
                                   ' viewer so arrow keys work
   End If

End Sub

Private Sub AnnotationListBox_Ctrl_DblClick()
   Rem ** On a double click of the annotation list box, display    Rem    the
```

```
data of the annotation in the annotation
   Rem    data form. **

   oixctrl1.FindAnnotation 0, 2,
   AnnotationListBox_Ctrl.ItemData
      (AnnotationListBox_Ctrl.ListIndex), 0
         ShowAnnotation oixctrl1.AnnotationDataType,
            AnnotationListBox_Ctrl.ItemData
               (AnnotationListBox_Ctrl.ListIndex),
                  oixctrl1.AnnotationData

End Sub

Private Sub AnnotationListBox_Ctrl_KeyDown(KeyCode As Integer,
   Shift As Integer)
   REM ** If delete key is pressed while a annotation is selected
   REM    in the annotation list box, we delete it using
   REM    ClearAnnotation **

   If KeyCode = vbKeyDelete Then
      oixctrl1.ClearAnnotations 3,
          AnnotationListBox_Ctrl.ItemData
             (AnnotationListBox_Ctrl.ListIndex)
   End If
End Sub
```

Raw Text

Often used in conjunction with the Annotation API, the Raw Text methods, events and properties allow the developer to programmatically access the data in the viewer as if it were all text. For example, a routine could be written to search through the raw text as the document was being loaded and add an annotation for each occurrence of a given character string. When the document is viewed, each matched character string would be hilited.

To enable raw text processing, the SystemRawText property (see SystemRawText) must be set to TRUE, and a RawTextEvent event handler must be written (see RawTextEvent). During the initial reading of the document, the RawTextEvent event is passed an ID that locates the raw text. The GetRawText method (see GetRawText) will populate RawTextOffset, RawTextString, and RawTextCharSet properties based on this ID (see RawTextOffset, RawTextString, and RawTextCharSet). Because the document is read in small increments, the developer should expect to see many calls to the RawTextEvent handler (each with a different raw text locator) before the entire document has been read.

As each page of the document is read, the raw text is accumulated into a raw text buffer within the control. The ID that is passed to the RawTextEvent event handler is actually an offset into this raw text buffer. These offsets may be stored in array fashion for later use. Passing the offset value into the GetRawText method copies the raw text information into the read-only properties.

```
Private Sub MainOIX_RawTextEvent(ByVal lTextOffset As Long)
  Rem ** RawText event handler will only get called when user
  Rem checks the appropriate check box on the form **
  Dim oixStart As New OixPos
  Dim oixEnd As New OixPos
  Dim pos As Long
  Dim searchStr As String

  searchStr = "the"
  pos = 1
  MainOIX.GetRawText (lTextOffset)
  Cs = MainOIX.RawTextCharSet
  pos = InStr(pos, MainOIX.RawTextString, searchStr, vbTextCompare)
```

```
  While (pos)
    ' pos is 1 based as returned from
    ' InStr but character count is 0 based
    MainOIX.SetActualCount oixStart, lTextOffset + (pos - 1)
    MainOIX.SetActualCount oixEnd, lTextOffset + (pos - 1) +
      Len(searchStr)
    MainOIX.AddAnnotationHilite Annotateid, 1,
      SCCVW_ANNOTATION_SCLICK, 0, 0, "Search Text", oixStart,
      oixEnd
    pos = InStr(pos + 1, MainOIX.RawTextString, searchStr,
      vbTextCompare)
    Annotatedid = Annotateid + 1
  Wend
End Sub
```

6

Advanced Topics

This chapter describes advanced topics in the use of ActiveX.

This chapter includes the following sections:

Drawing Pages

The Drawing Page API provides a mechanism to render individual document pages onto a system device context. For example, this could be used to create thumbnail images, or could be used to tile pages to a printer device context, etc. There are two levels of complexity associated with this set of methods. The simplest use consists of invoking the DrawPage method or ExtDrawPage method (see DrawPage and ExtDrawPage) with the page number, output rectangle and scaling factors. The resulting picture object will be placed in the PagePicture property (see PagePicture).

If the application needs more control over the drawing process, the lower level methods can be used. To initialize the API, the InitDrawPage method (see InitDrawPage) is invoked. It takes no parameters and must be called before any of the other methods are called. The initialization routines *must* also be called before either the DrawPage or GetDrawPageInfo methods are invoked (see DrawPage, and GetDrawPageInfo). The GetDrawPageInfo method uses the page number, output resolution (units/inch), and two device context parameters to populate the read-only properties. The PageFormatWidth and PageFormatHeight properties (see PageFormatWidth, and PageFormatHeight) contain the width and height of the page in logical units (physical page size x output resolution). PageUnitsPerInch (see PageUnitsPerInch) stores the output resolution passed into the GetDrawPageInfo method.

The DrawPageEx method (see DrawPageEx) actually renders the desired page into the output device context. Upon successful completion the PageResultTop, PageResultLeft, PageResultRight, and PageResultBottom read-only properties (see PageResultTop, PageResultLeft, PageResultRight, and PageResultBottom) will be populated and the method will return TRUE. Otherwise, an Error event (see Error) will be generated and the method will return FALSE. The PageResult*Ex properties are populated with the respective coordinates in DEVICE units. The PageResult*Ex properties are also populated when DrawPage (see DrawPage) is invoked.

The DrawPageEx method (see DrawPageEx) takes a number of parameters.

- Page Number: This zero-based integer specifies which page, as defined by the original document, is to be rendered.

- Flags: There are 3 flags which may be OR-ed together to control rendering:
  - 1: Based on the colors used on the page, return a palette handle in the PagePaletteHandle property (see PagePaletteHandle).
  - 2: Assume OutputDC is *not* a metafile DC irrespective of what Windows reports.
  - 4: Ignore the background when rendering.
- Top, Left, Bottom, Right: Rectangle in device coordinates in which to render the page.
- FormatWidth, FormatHeight, UnitsPerInch: Specifies the width and height in terms of the unitsperpage resolution factor of the area to which the text should be formatted (wrapped, justified, etc.)
- OutputDC, FormatDC: The device context in which to render the output and the device context which allows queries relating to formatting. These may not be the same DC.
- Palette: The palette to be used when rendering the page. If a palette is requested in the Flags parameter, it will be returned in the PagePaletteHandle property.

When the OutputDC is *not* a metafile or the Flags variable has bit 2 set (assume OutputDC is NOT a metafile DC irrespective of what Windows reports), the OutputDC will be set to the MM_ANISOTROPIC (abstract) mapping mode, the window and viewport extents will BOTH be set so that the page is drawn into the rectangle defined by the Top, Left, Bottom, Right parameters. If the OutputDC is a metafile, the device context will still be set to the MM_ANISOTROPIC mapping mode but only the window extents will be set. This will have the effect of creating a scalable metafile that can be played to any rectangle by setting the viewport and origin prior to metafile rendering.

```
Private Sub Picture1_Click()

   Rem ** When this picture control is clicked, the viewer will
   Rem    create a thumbnail image of the first page of the
   Rem    viewed document inside this control. It is then saved
   Rem    out to a filename of choice **

   Form1.MousePointer = vbHourglass
   oixctrl1.InitDrawPage
   oixctrl1.DrawPage 0, 0, 0, Picture1.Height, Picture1.Width,
      10, 10, 120
   Picture1.Cls

   Rem ** Calculate aspect ratio of original image so as to
   Rem    "paint" it with a similar ratio.

   aratio =(oixctrl1.PageResultRight-oixctrl1.PageResultLeft) /
           (oixctrl1.PageResultBottom - oixctrl1.PageResultTop)
   If (aratio > 1) Then
      maxw = Picture1.Width - 1
      maxh = Picture1.Width / aratio
   Else
      maxh = Picture1.Height
      maxw = Picture1.Height * aratio - 1
   End If

   Form1.MousePointer = vbNormal

   Rem ** Collect output filename and create bitmap
```

```
   CommonDialog1.ShowSave
   If (Not CommonDialog1.CancelError) Then
      Picture1.PaintPicture oixctrl1.PagePicture, 0, 0, maxw,
         maxh,
         oixctrl1.PageResultLeft, oixctrl1.PageResultTop,
            oixctrl1.PageResultRight -
         oixctrl1.PageResultLeft, oixctrl1.PageResultBottom -
            oixctrl1.PageResultTop
      SavePicture Picture1.Image, CommonDialog1.filename
   End If

   Rem ** Clean up oixctrl1

   oixctrl1.DeinitDrawPage

End Sub
```

Memory IO

The Outside In Memory IO API allows the developer to get access to the same data that can be placed on the clipboard without having to use the clipboard operations. The copy method takes starting and ending OixPos objects and a flag to indicate the desired format for the data. The data can be requested as clipboard-formatted text or RTF and will be placed in the CopyBuffer property (see CopyBuffer). The size of the data is placed in the CopyBufferSize property (see CopyBufferSize).

```
Private Sub ExportRTE_Menu_Click()
Rem ** Export the selected text to an RTF file

Oixctrll.Coopy 1, oixctrll.SelectionAnchor,
   oixctrll.SelectionEnd
CommonDialog1.ShowSave
If CommonDialog1.CancelError = False Then

   Kill CommonDialog1.FileName ' delete existing file first

   ' Write out data in CopyBuffer

   Open CommonDialog1.FileName for Binary Access Write As #1
   Put #1, , oixctrll.CopyBuffer
   Close #1
   End If

End Sub
```

Redirected IO

In some cases, the programmer may need control over the file IO that is performed by the viewer. For example, suppose that the document wasn't disk-based but instead was stored in a database field. In this case, two options are available: 1) read the data out of the database field, store it in a temporary disk file and pass that temporary file name to the viewer; or 2) handle the viewer's request for file IO. The ActiveX control provides the programmer with this control through the use of events. There are eight event handlers that must be written to support Redirected IO. All event handlers will be passed at least three parameters: a file name, variant user data, and a file handle. Upon return, all event handlers indicate whether or not it was successful in completing the request.

When a file is opened, the Open event handler (see Open) is responsible for returning the file handle. The file handle may be anything the programmer wants and will be passed back to the other IO events. Similarly, the Close event (see Close) has the responsibility to close the document.

Data is passed to the viewer from the application during the Read event (see Read). The event handler is passed the number of bytes to be read as well as a buffer in which to place them. Reading is to take place at the current file position and it is the application's responsibility to keep track of this position. The user-data variant variable is a good place to store semi-static data such as this. When the read request is finished, the number of bytes read and a success indicator are returned to the viewer.

Quite often the viewer will need to read non-sequentially from the file. The current file position is manipulated through the Seek event handler (see Seek). The Seek event handler is passed an offset value and offset anchor flag. The anchor flag can be one of three values: current position, beginning of document, or end of document. The offset value indicates how many bytes to relocate the current file pointer relative to the anchor. The Seek event returns a success indicator to the viewer after file pointer repositioning.

There are two informational events that can be generated whenever the viewer needs more information regarding the file being processed. The Tell event handler (see Tell) returns the current file position relative to the beginning of the document. The GetInfo event handler (see GetInfo) provides specific information regarding the file. The following table shows some of the information that may be requested. Only the first three requests actually require action from the GetInfo event handler, however, all requests must be responded to using the following values to communicate success to the viewer.

GetInfoRequest	Return Type	Success Value
IOGETINFO_ISOLE2STORAGE	N/A	IOERR_FALSE
IOGETINFO_FILENAME	String	IOERR_OK
IOGETINFO_PATHNAME	String	IOERR_OK
IOGETINFO_OSHANDLE	N/A	IOERR_BADINFOID
IOGETINFO_HSPEC	N/A	IOERR_BADINFOID

There are many documents that contain references to other files. When handling the viewer IO the application must be prepared to also handle these embedded references. When the viewer encounters an embedded file reference a GenSecond event (see GenSecond) will be generated. The referenced filename will be passed to this event as a string. It then becomes the responsibility of the application to return the remaining parameters.

File	Description
FileSpec	The name of the file as a variant. If the application is going to handle the embedded file IO as well, this data can be anything.
SpecType	The type of data placed in the FileSpec variable. The only supported value for this parameter is 3, indicating redirected IO.
VarData	Variant data to be passed along with the FileSpec to the Open event
Result	Success=1; Failed=0

Upon successful return of this event handler, an Open event will be generated for the embedded file.

```
Private Sub oixctrl1_Close(ByVal FileSpec As Variant,
                           ByVal varFile As Variant,
                           ByVal varData As Variant,
```

```
                           pResult As Long)
   Rem ** IO Redirect Close event handler -- call 16-bit
   Rem    file IO routines **
   pResult = IOERR_OK
   lclose (varFile)
End Sub

Private Sub oixctrl1_GenSecond(ByVal FileSpec As Variant, ByVal varData As
Variant, ByVal varFile As Variant, ByVal FileName As String, pFileSpec As
Variant, SpecType As Long, pvarData As Variant, pResult As Long)
   Rem ** IO Redirect GenSecond event handler -- build fully
   Rem    qualified file from filename passed in. **

   Dim NewFileName As String
   Dim FilePath As String

   pResult = IOERR_OK
   FilePath = FileSpec
   pos = InStr(FilePath, "\")
   While (pos <> 0)
      OldPos = pos
      pos = InStr(pos + 1, FilePath, "\")
   Wend

   NewFileName = Left(FilePath, OldPos) + FileName
   pFileSpec = NewFileName
   SpecType = 13
   pvarData = "Anything can go here"

End Sub

Private Sub oixctrl1_GetInfo(ByVal FileSpec As Variant, ByVal varData As
Variant, ByVal varFile As Variant, ByVal InfoId As Long, pInfo As Variant,
pResult As Long)

   Rem ** IO Redirect GetInfo event handler -- respond to viewer
   Rem    requests for additional information. Currently, only
   Rem    the cases shown are valid requests. **

   Dim FilePath As String
   Dim FileName As String

   FilePath = FileSpec
   pResult = IOERR_BADINFOID

   Select Case InfoIf
      Case IOGETINFO_FILENAME:
         ' return just the filename portion of the filespec parameter'

         pos = InStr(FilePath, "\")
         While (pos <> 0)
            OldPos = pos
            pos = InStr(pos + 1, FilePath, "\")
         Wend

         FileName = Right(FilePath, Len(FilePath) - OldPos)
         pInfo = FileName
         pResult = IOERR_OK

      Case IOGETINFO_PATHNAME: ' return just the path to the
                               ' filespec
         pInfo = FilePath
         pResult = IOERR_OK

      Case IOGETINFO_ISOLE2STORAGE: ' must respond to with
                                    ' IOERR_FALSE
         pResult = IOERR_FALSE
      End Select
End Sub
```

```
Private Sub oixctrl1_Open(ByVal FileSpec As Variant, ByVal varData As Variant,
pvarFile As Variant, pResult As Long)
   Rem ** IO Redirect Open event handler -- call into the 16-bit
   Rem    low level file IO routines. The handle returned from
   Rem    the open is returned as pvarFile **

   Dim hFile As Long
   pResult = IOERR_OK

   hFile = lopen(FileSpec, OF_READ)
   pvarFile = hFile
   If (hFile < 0) Then
      pResult = IOERR_NOFILE
   End If
End Sub

Private Sub oixctrl1_Read(ByVal FileSpec As Variant, ByVal varData As Variant,
ByVal varFile As Variant, pData As Variant, ByVal Size As Long, pCount As Long,
pResult As Long)
   Rem ** IO Redirect Read event handler -- call into the 16-bit
   Rem    routines to perform low level read on the file. **

   Dim Buffer() As Byte
   Dim ReadResult As Long
   ReDim Buffer(Size) As Byte

   pResult = IOERR_OK
   pCount = lread(varFile, Buffer(0), Size)
   pData = Buffer

   If (pCount = -1) Then
      pResult = IOERR_UNKNOWN
   End If
End Sub

Private Sub oixctrl1_Seek(ByVal FileSpec As Variant, ByVal varData As Variant,
ByVal varFile As Variant, ByVal From As Integer, ByVal Offset As Long, pResult
As Long)
   Rem ** IO Redirect Seek event handler -- call into 16-bit
   Rem    routines to perform seek. **

   Dim Tally As Long
   pResult = IOERR_OK

   If (llseek(varFile, Offset, From) = -1) Then
      pResult = IOERR_UNKNOWN
   End If
End Sub

Private Sub oixctrl1_Tell(ByVal FileSpec As Variant, ByVal varData As Variant,
ByVal varFile As Variant, pOffset As Long, pResult As Long)
   Rem ** IO Redirect Tell event handler -- call into 16-bit
   Rem    routines to get current file position. **

   pResult = IOERR_OK
   pOffset = llseek(varFile, 0, FILE_CURRENT)
End Sub
```

7

Miscellaneous Topics

This chapter describes additional topics on the use of the Outside In Viewer for ActiveX.

This chapter includes the following sections:

System Read Ahead

Outside In uses a technique known as read-ahead in order to cache pages of text in memory for faster display. There are three elements that control this process. If the property SystemReadAhead (see SystemReadAhead) is set to FALSE, then the process which reads and caches the next few pages will be disabled. This means that there could be noticeable delays in the view refresh as the user scrolls through the document. It can also cause unusual scroll bar behavior.

When SystemReadAhead is enabled, the SystemTimer property (see SystemTimer) is a Boolean value that turns all background processing, including read-ahead, on and off. If the SystemTimer property is turned off, read-ahead can be controlled by the programmer by calling the SystemIdle method (see SystemIdle) to allow the viewer to perform background tasks such as read-ahead, caret blinking and auto-scroll.

Errors

There are two types of errors that can be generated by the ActiveX control. Errors, which occur due to incorrect use of the control, will be caught by the system and will generate an OLE exception. Errors that occur due to invalid parameter content (for example, missing files) will generate an Error event. By default the ErrorShowMsg property (see ErrorShowMsg) is set to TRUE and will display error messages as they occur. The property can be set to FALSE to disable this reporting. In either case, the Error event will be generated.

The Error event handler may access the ErrorCode and ErrorMsg properties (see ErrorCode, and ErrorMsg) to determine the nature of the error. The ErrorMsg is a BSTR variable and can be used to display errors as the developer sees fit.

```
Private Sub oixctrl1_Error()
Rem ** Error event handler for viewer

Dim Text As String

Select Case oixctrl1.ErrorCode

   Case SCCERR_ALLOCFAILED:
      Text = "SCCERR:" + oixctrl1.ErrorMsg
   Case SCCERR_INSUFFICIENTBUFFER:
      Text = "SCCERR:" + oixctrl1.ErrorMsg
```

```
      Case SCCERR_MEMORYLEAK:
         Text = "SCCERR:" + oixctrl1.ErrorMsg
      Case SCCERR_MEMTABLEFULL:
         Text = "SCCERR:" + oixctrl1.ErrorMsg
      End Select
      MsgBox Text, vbOKOnly, "Outside In Error"

   End Sub
```

A

SDK Reference

This appendix serves as a reference for the Software Developer's Kit.

Quick Reference

The tables in this section act as a pointer to more detailed information later in this chapter.

An asterisk in the following lists indicates a read-only action.

Viewing Documents

This section contains properties, methods and events used when viewing documents.

Selecting Files

PROPERTIES	METHODS	EVENTS
CurrentPageNumber*	Clear	DisplayEngineChange
DefaultInputCharSet	ViewFile	FileChange
DisplayEngineName*		GetCredentials
DisplayEngineType*		InformationMessage
DisplayFont		ViewThisFile
DocumentMemoryMode		
EMailDisplayMode		
EMailFitMode		
FallbackFormat		
FileInfoDisplayName*		
FileInfoFileId*		
FileInfoFileIdName*		
FileInfoName*		
FilterOptions		
FontScalingFactor		
HTMLDisplayMode		
HTMLFitMode		
MaintainZoom		
SystemUnicode		
UnmappableChar		

Printing

PROPERTIES	METHODS	EVENTS
PrintCollate	PrintSetup	PrinterAbort
PrintCopies	PrintOI	
PrintEndPage		
PrinterDC		

PROPERTIES	METHODS	EVENTS
PrinterDriver PrintJobName PrinterName PrinterPort PrintStartPage PrintFont PrintMarginLeft_ Right_ Top_ and Bottom UseDocPageSettings WhatToPrint		

Clipboard

PROPERTIES	METHODS	EVENTS
ClipFont ClipInfo* EnableDragNDrop ToClipboard	CopyToClip	None

User Interface Options

This section describes user interface options.

Text Selection

PROPERTIES	METHODS	EVENTS
SelectionAnchor SelectionEnd	SelectAll	SelChange

Menus

PROPERTIES	METHODS	EVENTS
DoContextMenu	None	ContextMenu

Dialogs

PROPERTIES	METHODS	EVENTS
DialogFlags ResourceLibraryID	ClipboardOptions DisplayOptions PrintOptions	DoHelp EnableApp OptionChange

Scroll Bar Interaction

PROPERTIES	METHODS	EVENTS
HScrollbar VScrollbar	HScroll VScroll	HScrollPageSize HScrollPosition HScrollRange HScrollState VScrollPageSize VScrollPosition

PROPERTIES	METHODS	EVENTS
		VScrollRange (obsolete) VScrollRangeMin VScrollRangeMax VScrollState

Miscellaneous

PROPERTIES	METHODS	EVENTS
HTMLCondCommentMode ParseXMPMetadata	IdleBitmap GetProperty	Keydown

Display Engine Options

This section presents Display Engine options.

Formatting

PROPERTIES	METHODS	EVENTS
FormatFlags IntlFlags TimeZoneOffset	None	None

Image

PROPERTIES	METHODS	EVENTS
AntiAliasMode BMPFitMode BMPPrintAspect BMPPrintBorder BMPDither BMPDitherAvailable* BMPRotation ImgXZoomPercent ImgYZoomPercent RenderEmbeddedFonts StrokeText VecFitMode VecPrintAspect VecPrintBorder VecPrintBackground VecShowBackground	ImgZoom ImgShowFullScreen	None

Word Processing

PROPERTIES	METHODS	EVENTS
EMailViewDisabled GetCustomEMailHeader SetCustomEmailHeader WPDisplayMode WPFitMode	None	None

PROPERTIES	METHODS	EVENTS
WPEmailHeaderOutputMode WPWrapToWindow		

Archive Options

PROPERTIES	METHODS	EVENTS
ArchiveSortOrder ArchiveSortDescending	ArchiveSave	None

Database/Spreadsheets

PROPERTIES	METHODS	EVENTS
DBClipBoard DBDraftMode DBFieldNamesToClip DBPrintGridLines DBPrintHeadings DBPrintFit DBShowGridLines SSClipBoard SSDraftMode SSPrintDirection SSPrintGridLines SSPrintHeadings SSPrintFit SSPrintScalePercent SSPrintScaleXHigh SSPrintScaleXWide SSShowGridLines SSShowHiddenCells	None	None

Searching

PROPERTIES	METHODS	EVENTS
None	Search SearchNext	None

Positions

PROPERTIES	METHODS	EVENTS
None	ComparePositions CopyPosition DisplayPosition FindPosition GetActualCount SetPositionToCurrent SetPositionToSelection SetActualCount	None

Annotations

PROPERTIES	METHODS	EVENTS
AnnotationData* AnnotationDataType* AnnotationEndPos* AnnotationId* AnnotationStartPos*	AddAnnotationHilite AddAnnotationHideText AddAnnotationInsertText AddAnnotationPicture ClearAnnotations FindAnnotation AnnotationSetPos GetAnnotationData GoToAnnotation HiliteStyle	AnnotationEvent

Raw Text

PROPERTIES	METHODS	EVENTS
RawTextCharSet* RawTextLength* RawTextOffset* RawTextString* SystemRawText	GetRawText	RawTextEvent

Advanced Topics

The following tables describe more advanced topics.

Drawing Pages

PROPERTIES	METHODS	EVENTS
PageFormatHeight* PageFormatWidth* PagePaletteHandle* PagePicture* PageResultBottom* PageResultLeft* PageResultRight* PageResultTop* PageUnitsPerInch*	DeinitDrawPage DrawPage DrawPageEx ExtDrawPage GetDrawPageInfo GetPageCount InitDrawPage	None

Memory I/O

PROPERTIES	METHODS	EVENTS
CopyBuffer CopyBufferSize	Copy	None

Redirected I/O

PROPERTIES	METHODS	EVENTS
None	None	Close GenSecond

PROPERTIES	METHODS	EVENTS
		GetInfo Open Read Seek Tell

System

PROPERTIES	METHODS	EVENTS
FIFlags ReorderMethod StatusEvents SysLotusNotesPath SystemTimer SystemReadAhead	SystemIdle	GetTechPath ReadAheadDone Stat

Miscellaneous

The following are other items referenced in this chapter.

Errors

PROPERTIES	METHODS	EVENTS
ErrorCode* ErrorMsg* ErrorShowMsg	None	Error

Properties

The following properties are used in the SDK.

Note:

All property options are Local in scope except where noted.

AnnotationData

Read only. This contains the user data value of the last annotation accessed.

Type

VARIANT

Default

0

Related to

SCCVW_FINDANNOTATION (Outside In Viewer)

AnnotationDataType

Read only. This contains the format of the last annotation accessed. Valid values are:

- ADTYPE_NODATA: User Data. The annotation was created by using one of the AddAnnotation* methods.
- ADTYPE_URL: URL. The annotation is a URL hyperlink that was present in the original file format.
- ADTYPE_BOOKMARK: Bookmark. The annotation is a named bookmark that was present in the original file format.

Type

Short

Default

ADTYPE_NODATA

Related to

SCCVW_GETANNOTATIONDATA (Outside In Viewer)

AnnotationEndPos

Read only. This contains the position object of the last annotation accessed.

Type

OixPos

Default

none

Related to

SCCVW_FINDANNOTATION (Outside In Viewer)

AnnotationId

Read only. This contains the Id value of the last annotation added or found.

Type

Long

Default

0

Related to

SCCVW_FINDANNOTATION (Outside In Viewer)

AnnotationStartPos

Read only. This contains the position object of the last annotation accessed. This position is a zero-based character count.

Type

OixPos

Default

none

Related to

SCCVW_FINDANNOTATION (Outside In Viewer)

AntiAliasMode

This property determines whether bitmaps with higher color resolution than the screen should use a dithering algorithm to improve the display.

Note:

This option is Global in scope.

Type

BOOL

Default

True

Related to

SCCID_ANTIALIAS (Outside In Viewer)

ArchiveSortDescending

If the archive listing is sorted (see ArchiveSortOrder), this property determines the direction that the list will be sorted.

Type

BOOL

Default

FALSE

Related to

SCCID_ARCSORTORDER (Outside In Viewer)

ArchiveSortOrder

Determines the order in which the files are displayed in the archive display engine. One of the following values:

- SCCVW_SORT_NONE: Display files in the order they appear in the archive.
- SCCVW_SORT_NAME: Display files sorted by their name.
- SCCVW_SORT_SIZE: Display files sorted by their size.
- SCCVW_SORT_DATE: Display files sorted by their date.

Type

Short

Default

SCCVW_SORT_NONE

Related to

SCCID_ARCSORTORDER (Outside In Viewer)

BMPDither

Determines whether or not to use a dithering algorithm on bitmaps with higher color resolution than the screen in order to get a better color display. One of the following values:

- TRUE: Use dithering algorithm.
- FALSE: Do not use dithering algorithm.

Type

BOOL

Default

FALSE

Related to

SCCID_BMPDITHER (Outside In Viewer)

BMPDitherAvailable

Read only. This property indicates whether or not the current image can be dithered. For instance, dithering is not possible or necessary for a 16-color bitmap on a 256-color display, but is possible for a 256 bitmap on a 16-color display. One of the following values:

- TRUE: Current image can be dithered.
- FALSE: Current image cannot be dithered.

Type

BOOL

Default

TRUE

Related to

SCCID_BMPDITHERAVAILABLE (Outside In Viewer)

BMPFitMode

Determines how the display engine (vector or bitmap) displays (stretches/shrinks) an image in relation to the size of the view window. One of the following values:

- SCCVW_FITMODE_BEST: If the view window is smaller than the original image, this option will fit it to the window. If the view window is larger than the original image, the image will be displayed at its original size.
- SCCVW_FITMODE_IMAGESIZE: Scale to image size.
- SCCVW_FITMODE_PIXELS: The size of the window has no effect.
- SCCVW_FITMODE_WINDOW: The image will be stretched to fill as much of the window as possible while maintaining its proper aspect ratio.
- SCCVW_FITMODE_WINDOWHEIGHT: Height of the window. Depending on the image, its full width may or may not fit inside the window.
- SCCVW_FITMODE_WINDOWWIDTH: The bitmap will be stretched so its full width fits in the width of the window.

Invalid values are reset to the last set value.

Type

Short

Default

SCCVW_FITMODE_ORIGINAL

Related to

SCCID_BMPFITMODE (Outside In Viewer)

BMPPrintAspect

The option indicates how the image will be stretched when printed. It can be one of the following values:

- SCCVW_PRINTASPECT_IMAGESIZE: Uses the size as specified by the image.
- SCCVW_PRINTASPECT_SCALE (previously referred to as SCCVW_PRINTASPECT_ORIGINAL): The image will be sized up to fill as much of the area inside the print margins as possible while still maintaining the original aspect ratio.

- SCCVW_PRINTASPECT_STRETCH: The image will be stretched horizontally and vertically to totally fill the area inside the print margins.

Invalid values will be reset to the last set value.

Type

Short

Default

SCCVW_PRINTASPECT_SCALE

Related to

SCCID_BMPPRINTASPECT (Outside In Viewer)

BMPPrintBorder

Determines if a one-pixel border is to be printed around images. One of the following values:

- TRUE: Print a one-pixel border.
- FALSE: Do not print a one-pixel border.

Type

BOOL

Default

TRUE

Related to

SCCID_BMPPRINTBORDER (Outside In Viewer)

BMPRotation

Determines how the bitmap is to be rotated. These rotations are absolute from the graphics initial state. One of the following values:

- SCCVW_ROTATION_NONE: Bitmaps should have no rotation.
- SCCVW_ROTATION_90: Bitmaps should be rotated 90 degrees clockwise.
- SCCVW_ROTATION_180: Bitmaps should be rotated 180 degrees clockwise.
- SCCVW_ROTATION_270: Bitmaps should be rotated 270 degrees clockwise.

Type

Short

Default

SCCVW_ROTATION_NONE

Related to

SCCID_BMPROTATION (Outside In Viewer)

ClipFont

This property allows the developer to set the default clipboard font. The developer may implement a font dialog or set this value directly. If the developer chooses to implement his own dialog, be advised that a default font will be used for any printer font selected from the dialog.

The developer should set the ToClipboard property (see ToClipboard) to RTF (0x2) to copy text to the clipboard in this font.

Type

LPFONTDISP

Related to

SCCID_DEFAULTCLIPBOARDFONT (Outside In Viewer)

ClipInfo

Read only. This property holds flags that specify the readiness of the view window to manipulate clipboard information.

Currently, valid values for this property are:

- SCCVW_NOTREADYCOPYTOCLIP: Not ready to copy to clipboard
- SCCVW_CANCOPYTOCLIP: Ready to copy to clipboard
- SCCVW_CANCUTTOCLIP: Ready to cut to clipboard
- SCCVW_CANPASTEFROMCLIP: Items available on clipboard for pasting

Type

ULONG

Default

SCCVW_NOTREADYCOPYTOCLIP

Related to

SCCVW_GETCLIPINFO (Outside In Viewer)

CopyBuffer

Read only. This property contains a string that contains the data requested by a call to the Copy method (see Copy). If no call has been made to the Copy method, then the value will be a null string.

Type

BSTR

Default

Null

Related to

SCCVW_COPY (Outside In Viewer)

CopyBufferSize

Read only. This property contains the size of the data produced in the copy buffer as a result of a call to the Copy method (see Copy). If no call has been made to the Copy method, then the value will be zero.

Type

Long

Default

0

Related to

SCCVW_COPY (Outside In Viewer)

CurrentPageNumber

Read only. This property contains the page number of the page currently displayed in the view window. A value of 0 is returned to indicate that the current page number is not available.

The page number is available when the word processing display engine is in preview mode, and not available when it is in draft or normal mode. For other formats such as spreadsheets, presentations, and multi-page images, the page number is equivalent to the number of the current sheet, slide, or image.

Type

Long

Default

0

Related to

SCCVW_GETCURRENTPAGENUMBER (Outside In Viewer)

DBClipBoard

Determines which format the database takes when it is copied to the clipboard. One of the following values:

- SCCVW_CLIPSUBFORMAT_TABLE: In supporting clipboard formats (RTF & AMI), the form will be copied as a table.
- SCCVW_CLIPSUBFORMAT_OPTIMIZEDTABS: The form will be copied using a tab stop for each field, except when a field is empty.

- SCCVW_CLIPSUBFORMAT_TABS: The form will be copied using a tab stop for each field.

Type

Short

Default

SCCVW_CLIPSUBFORMAT_TABLE

Related to

SCCID_DBCLIPBOARD (Outside In Viewer)

DBDraftMode

Determines how much formatting information is included when displaying a database or spreadsheet. One of the following values:

- TRUE: A limited set of formatting information is included.
- FALSE: All supported formatting information is included.

Type

BOOL

Default

TRUE

Related to

SCCID_DBDRAFTMODE (Outside In Viewer)

DBFieldNamesToClip

Determines if field headings / row and column names will the copied to the clipboard along with the data. One of the following values:

- TRUE: Field headings will be copied.
- FALSE: Field headings will not be copied.

Type

BOOL

Default

TRUE

Related to

SCCID_DBFIELDNAMESTOCLIP (Outside In Viewer)

DBPrintFit

Determines how the database printouts are fitted to the paper size and default print margins.

If the scaling is too much (one-fifth the original size) it is limited to that amount and will continue on additional pages. Aspect ratio is maintained. One of the following values:

- SCCVW_DBPRINTFITMODE_NOMAP: Printing will occur in its original size onto as many pages as are required to fit the data.
- SCCVW_DBPRINTFITMODE_FITTOPAGE: The form will be fitted onto one page, scaling to the image width or height depending on the page size and database size.
- SCCVW_DBPRINTFITMODE_FITTOWIDTH: The form will be scaled on the printed page so it is no larger than one page wide.
- SCCVW_DBPRINTFITMODE_FITTOHEIGHT: The form will be scaled on the printed page so it is no larger than one page high

Type

Short

Default

SCCVW_DBPRINTFITMODE_NOMAP

Related to

SCCID_DBPRINTFITTOPAGE (Outside In Viewer)

DBPrintGridLines

Determines if a dotted line is printed between fields. One of the following values:

- TRUE: A dotted line is printed between fields.
- FALSE: A dotted line is not printed between fields.

Type

BOOL

Default

TRUE

Related to

SCCID_DBPRINTGRIDLINES (Outside In Viewer)

DBPrintHeadings

Determines if field headings will be printed along with the data. One of the following values:

- TRUE: Field headings will be printed.
- FALSE: Field headings will not be printed.

Type

BOOL

Default

TRUE

Related to

SCCID_DBPRINTHEADINGS (Outside In Viewer)

DBShowGridLines

Determines if a dotted line is displayed between fields. One of the following values:

- TRUE: A dotted line is displayed between fields.
- FALSE: A dotted line is not displayed between fields.

Type

BOOL

Default

TRUE

Related to

SCCID_DBSHOWGRIDLINES (Outside In Viewer)

DefaultInputCharSet

This property defines the character set to be used when the technology cannot determine the character set used to encode the text of an input file. When all other means of determining the file's character set are exhausted, Outside In will assume that an input document is encoded in the character set specified by this option. This is most often used when reading plain-text files, but may also be used when reading HTML or PDF files. Possible values are the CS_ values defined in the header files.

Note:

This option supersedes the FallbackFormat option for selecting the character set assumed for plain-text files. For backwards compatibility, use of deprecated character-set -related values is still currently supported for FallbackFormat, though internally such values will be translated into equivalent values for the DefaultInputCharSet. As a result, if an application were to set both options, the last such value set for either option would be the value that takes effect.

Note:

This option is Global in scope.

Type

LONG

Default

CS_SYSTEMDEFAULT

Related to

SCCID_DEFAULTINPUTCHARSET (Outside In Viewer)

DialogFlags

Determines how dialogs are displayed and how dialog choices are added to the menus. It can be one or more of the following OR-ed together.

- SCCVW_DIALOG_NO3D: The dialogs available in the Viewer should not use Microsoft's CTL3DV2.DLL to give its dialogs a 3D effect. If CTL3DV2.DLL is not available, this flag has no effect. Note: CTL3DV2.DLL is loaded dynamically (LoadLibrary) so systems without it will not cause a problem.
- SCCVW_DIALOG_NOADDOPTIONSTOMENU: The dialog menu retrieved from SCCVW_GETDISPLAYINFO contains a popup with three additional items called Display..., Print..., and Clipboard.... Setting this flag will disable this extra popup.
- SCCVW_DIALOG_NOADDDISPLAYTOMENU: The dialog should not display just the Display... item from the menu just described.
- SCCVW_DIALOG_NOADDFONTZOOMTOMENU: Normally, the menu retrieved from SCCVW_GETDISPLAYINFO includes a popup menu called Font Size, which allows the user to enlarge, reduce or reset the font size relative to its original size. This flag disables this popup menu item.
- SCCVW_DIALOG_NOADDPRINTTOMENU: The dialog should not display just the Print... item from the menu just described.
- SCCVW_DIALOG_NOADDCLIPBOARDTOMENU: The dialog should not display just the Clipboard... item from the menu just described.
- SCCVW_DIALOG_NOADDDOPRINTTOMENU: The dialog menu retrieved from SCCVW_GETDISPLAYINFO should not display the Print... menu item.
- SCCVW_DIALOG_NOADDDOCOPYTOMENU: The dialog menu retrieved from SCCVW_GETDISPLAYINFO should not display the Copy menu item.
- SCCVW_DIALOG_NOHELP: The dialogs available in the viewer technology should not display "Help" buttons.
- SCCVW_DIALOG_NOMORE: The dialogs available in the viewer technology should not display "More" buttons. A "More" button allows access to another dialog with more obscure options.

- SCCVW_DIALOG_NOADDSHOWFULLSCREEN: The dialog should not display the "Show Full Screen" menu option from the context menu

Type

Short

Default

0

Related to

SCCID_DIALOGFLAGS (Outside In Viewer)

DisplayEngineName

Read only. This property contains the name of the display engine. These names include: "Spreadsheet," "Database," "Document," "Bitmap," "Archive," "Hex," and "Drawing."

Type

BSTR

Default

Null

Related to

SCCVW_GETDISPLAYINFO (Outside In Viewer)

DisplayEngineType

Read only. This property contains a numeric identifier for the type of display engine in use. The types are as follows:

- SCCVWTYPE_NONE: No Display Engine is loaded
- SCCVWTYPE_UNKNOWN: Unknown Display Engine
- SCCVWTYPE_WP: Word Processor Display Engine
- SCCVWTYPE_SS: Spreadsheet Display Engine
- SCCVWTYPE_DB: Database Display Engine
- SCCVWTYPE_HEX: Hexadecimal Display Engine
- SCCVWTYPE_IMAGE: Bitmap graphics Display Engine
- SCCVWTYPE_ARCHIVE: Archive Display Engine
- SCCVWTYPE_VECTOR: Vector Display Engine

Type

SHORT

Default

SCCVWTYPE_NONE

Related to

SCCVW_GETDISPLAYINFO (Outside In Viewer)

DisplayFont

This property allows the developer to set the default display font. The developer may implement a font dialog or set this value directly.

Type

LPFONTDISP

Related to

SCCID_DEFAULTDISPLAYFONT (Outside In Viewer)

DoContextMenu

This property determines if the built-in context menu will be displayed when the user performs a right-mouse click. If this is set to FALSE then the control will not display a context menu and the developer will have to implement the ContextMenu event (see ContextMenu) in order to provide one.

Type

BOOL

Default

TRUE

Related to

SCCVW_CONTEXTMENU (Outside In Viewer)

DocumentMemoryMode

This property determines the maximum amount of memory that the internal caching system may use to store the document's data, from 4 MB to 1 GB. The more memory that is available to the caching system, the less often it needs to re-read data from the document. It can be one of the following values:

- SCCDOCUMENTMEMORYMODE_SMALLEST : 4MB
- SCCDOCUMENTMEMORYMODE_SMALL : 16MB
- SCCDOCUMENTMEMORYMODE_MEDIUM : 64MB
- SCCDOCUMENTMEMORYMODE_LARGE : 256MB
- SCCDOCUMENTMEMORYMODE_LARGEST : 1 GB

Note:

This option is Global in scope.

Type

LONG

Default

SCCDOCUMENTMEMORYMODE_LARGE (256MB)

Related to

SCCID_DOCUMENTMEMORYMODE (Outside In Viewer)

EMailDisplayMode

This property indicates how the word processor display engine displays the email file. It can be one of the following:

- SCCVW_WPMODE_DRAFT: The document is displayed using only a single font and size (determined by the DisplayFont property. For more information, see DisplayFont). It does not display embedded graphics, graphic borders, or table borders. Text is wrapped to the size of the view window.
- SCCVW_WPMODE_NORMAL: All supported formatting is displayed. The text is wrapped to the size of the view window.
- SCCVW_WPMODE_PREVIEW: All supported formatting is displayed. The text is wrapped as it will be printed.
- SCCVW_WPMODE_WEBLAYOUT: Display all supported formatting, wrap the text as it would appear in a browser.

Type

Short

Default

SCCVW_WPMODE_WEBLAYOUT

Related to

SCCID_EMAILDISPLAYMODE (Outside In Viewer)

EMailFitMode

Controls the size of email files when using preview or weblayout mode.

Data

One of the following values:

- SCCVW_FITMODE_ORIGINAL: Sizes the preview page to the actual size.

- SCCVW_FITMODE_WINDOWWIDTH: Sizes the preview page to the width of the window.
- SCCVW_FITMODE_WINDOW: Sizes the preview page to the window. Sizes the weblayout to the width of the window.

EMailViewDisabled

This property determines whether or not the E-mail header will be displayed.

Note:

This option is Global in scope.

Type

BOOL

Default

False

Related to

SCCID_WPDISABLEEMAILHEADER (Outside In Viewer)

EnableDragNDrop

Enables the drag and drop capability of the control. This allows users to click on a highlighted selection in a document and drag it to a target application that can accept it.

Type

BOOL

Default

FALSE

Related to

SCCID_OLEFLAGS (Outside In Viewer)

ErrorCode

Read only. Contains a code for the error status of the last operation performed by the control. This code can refer to both errors in the ActiveX control or in the Outside In Technology. It is 0 if there is no error.

Type

LONG

Default

0

Related to

SCCVW_BAILOUT (Outside In Viewer)

ErrorMsg

Read only. Contains an error message that corresponds to error status reported by the ErrorCode property (see ErrorCode). It is a zero length string if there is no error.

Type

BSTR

Default

0

Related to

SCCVW_BAILOUT (Outside In Viewer)

ErrorShowMsg

If this is set to TRUE then the control will automatically display error messages as errors occur. Otherwise, the messages will not be shown.

Type

Short int (possible future extension)

Default

TRUE

Related to

SCCVW_BAILOUT (Outside In Viewer)

FallbackFormat

This property defines how the viewer displays unidentifiable documents. Valid values for this property are as follows:

- FI_ANSI: ANSI
- FI_ANSI8: ANSI18
- FI_ASCII: ASCII
- FI_ASCII8: ASCII8
- FI_CHINESEBIG5: CHINESEBIG5
- FI_CHINESEGB: CHINESEGB
- FI_DONTVIEW: DONTVIEW
- FI_HANGEUL: HANGEUL
- FI_HEX: HEX

- FI_MAC: MAC
- FI_MAC8: MAC 8
- FI_SHIFTJIS: SHIFTJIS
- FI_UNICODE: Unicode

Note:

This option is Global in scope.

Type

Int

Default

FI_ANSI

Related to

SCCID_FALLBACKFORMAT (Outside In Viewer)

FIFlags

This property is set to use flags related to File Identification behavior. It can be one of the following:

- FALSE: The File Identification code will identify all formats that are supported by the ActiveX control without performing extended file identification on unknown files.
- TRUE: The File Identification code will run an extended test on all files that are not identified (for example, FI_UNKNOWN). This will try to identify files as 7-bit text file (FI_7BITTEXT) or as an ANSI text file (FI_ANSI8).

Note:

This option is Global in scope.

Type

Bool

Default

FALSE

Related to

SCCID_FIFLAGS (Outside In Viewer)

FileInfoDisplayName

Read only. String used when a human-readable form of the file name is needed. This string is either set specifically by the developer in the ViewFile method (see ViewFile) or is generated by the viewer based on the real name of the file. This property is updated whenever the FileChange event (see FileChange) occurs.

Type

BSTR

Default

Null

Related to

SCCVW_GETFILEINFO, SCCVW_SETDISPLAYNAME (Outside In Viewer)

FileInfoFileId

Read only. The ID number associated with this type of file. These values are listed in the header files.

Type

Short

Default

0

Related to

SCCVW_GETFILEINFO (Outside In Viewer)

FileInfoFileIdName

Read only. This is a string that is a description of the file ID number. For example, "WordPerfect 5.0." See the header files for a listing of these strings.

Type

BSTR

Default

NULL

Related to

SCCVW_GETFILEINFO (Outside In Viewer)

FileInfoName

Read only. The full path to the actual file being viewed.

Type

BSTR

Default

Null

Related to

SCCVW_GETFILEINFO (Outside In Viewer)

FilterOptions

This property determines the behavior of certain filters. This property can contain one or more of the following values OR-ed together:

- OIX_ENABLEJPG: Enable access to any files using JPEG compression.
- OIX_ENABLELZW: Enable access to any files using LZW compression.
- OIX_FILTER_REORDERED_BIDI: Enable reordering of bi-directional text from PDF files in correct reading order.
- OIX_IGNOREPASSWORD: Disable password verification of files where the contents can be processed without validation of the password.

The following is a list of file types affected when OIX_ENABLELZW option is turned off:

- GIF files
- TIF files using LZW compression
- PDF files that use internal LZW compression
- TAZ and TAR archives containing files that are identified as FI_UNIXCOMP
- ZIP and self-extracting archive (.EXE) files containing "shrunk" files
- Postscript files using LZW compression

Although this option can disable access to files in ZIP or EXE archives stored using LZW compression, any files in such archives that were stored using any other form of compression will still be accessible.

The following is a list of file types affected when OIX_ENABLEJPG option is turned off:

- JPG files
- Postscript files containing JPG images
- PDFs containing JPEG images

As of Release 8.4.0, only the PST and MDB Filters support the OIX_IGNOREPASSWORD option.

Note:

This option is Global in scope.

Type

LONG

Default

OIX_ENABLEJPG | OIX_ENABLELZW

Related to

SCCID_IGNORE_PASSWORD, SCCID_FILTERJPG, SCCID_FILTERLZW, and SCCID_PDF_FILTER_REORDER_BIDI (Outside In Viewer)

FontScalingFactor

This property scales fonts in the view of a file by a percentage of the original size. The valid range of values is 50% to 300%.

Type

long

Default

100

Related to

SCCID_FONTSCALINGFACTOR (Outside In Viewer)

FormatFlags

This option determines the formatting options of data. It can be one of the following values:

- SCCOPT_FLAGS_ALLISODATETIMES: When this flag is set, all Date and Time values are converted to the ISO 8601 standard. This conversion can only be performed using dates that are stored as numeric data within the original file.
- 0: All flags turned off

Type

LONG

Default

0

Related to

SCCOPT_FORMATFLAGS (Outside In Viewer)

HScrollbar

This property controls the visibility of the horizontal scroll bar. TRUE means that the scrollbar is always visible, but may be disabled if unnecessary. A FALSE value makes the scrollbar invisible.

The Outside In Viewer does not have the ability to automatically show/hide scrollbars. This property takes the place of the ScrollFlags property because there are only two real values for each scrollbar.

Type

BOOL

Default

TRUE

Related to

SCCID_SCROLLFLAGS (Outside In Viewer)

HTMLCondCommentMode

Some HTML includes a special type of comment that will be read by particular versions of browsers or other products. This property allows you to control which of those comments are included in the output.

Data

- One or more of the following values OR-ed together:
- HTML_COND_COMMENT_NONE: Don't output any conditional comments. Note: setting any other flag will negate this.
- HTML_COND_COMMENT_IE5: include the IE 5 comments
- HTML_COND_COMMENT_IE6: include the IE 6 comments
- HTML_COND_COMMENT_IE7: include the IE 7 comments
- HTML_COND_COMMENT_IE8: include the IE 8 comments
- HTML_COND_COMMENT_IE9: include the IE 9 comments
- HTML_COND_COMMENT_ALL: include all conditional comments including the versions listed above and any other versions that might be in the HTML.

Default

HTML_COND_COMMENT_NONE

Related to

SCCID_HTML_COND_COMMENT_MODE (Outside In Viewer)

HTMLDisplayMode

This property indicates how the word processor display engine displays the HTML file. It can be one of the following:

- SCCVW_WPMODE_DRAFT: The document is displayed using only a single font and size (determined by the DisplayFont property. For more information, see DisplayFont). It does not display embedded graphics, graphic borders, or table borders. Text is wrapped to the size of the view window.
- SCCVW_WPMODE_NORMAL: All supported formatting is displayed. The text is wrapped to the size of the view window.
- SCCVW_WPMODE_PREVIEW: All supported formatting is displayed. The text is wrapped as it will be printed.
- SCCVW_WPMODE_WEBLAYOUT: Display all supported formatting, wrap the text as it would appear in a browser.

Type

Short

Default

SCCVW_WPMODE_WEBLAYOUT

Related to

SCCID_HTMLDISPLAYMODE (Outside In Viewer)

HTMLFitMode

Controls the size of HTML files when using preview or weblayout mode.

Data

One of the following values:

- SCCVW_FITMODE_ORIGINAL: Sizes the preview page to the actual size.
- SCCVW_FITMODE_WINDOWWIDTH: Sizes the preview page to the width of the window.
- SCCVW_FITMODE_WINDOW: Sizes the preview page to the window. Sizes the weblayout to the width of the window.

Default

SCCVW_FITMODE_ORIGINAL

ImgXZoomPercent

The percent used to stretch/shrink the horizontal size of the image. For instance, setting the percent to 200 will show 2 display pixels for every image pixel, that is, twice the horizontal size.

Type

Long

Default

100

Related to

SCCID_VECZOOM, SCCID_BMPZOOM (Outside In Viewer)

ImgYZoomPercent

The percent used to stretch/shrink the vertical size of the image. For instance, setting the percent to 200 will show 2 display pixels for every image pixel, that is, twice the vertical size.

Type

Long

Default

100

Related to

SCCID_VECZOOM, SCCID_BMPZOOM (Outside In Viewer)

IntlFlags

These flags relate to setting options for international support. If IntlFlags is set to TRUE, then print margins are displayed in inches; otherwise, print margins will be displayed in metric units.

Type

BOOL

Default

The default value is retrieved from the operating system.

Related to

SCCID_INTLFLAGS (Outside In Viewer)

Data

- SCCVW_ENGLISHUNITS: If set, the print margins are displayed to the users in inches. If not set, the print margins are displayed in metric units.
- SCCVW_12HOURTIME: If set, time variables are formatted for display using a 12-hour clock. If not set, time variables are formatted for display using a 24-hour clock.

MaintainZoom

This option tells the Viewer to maintain its zoom and rotation settings when changing sections within a file. As each page of a presentation file is displayed as a new section, this allows a user to zoom to their desired level once, and keep that setting as they page through the presentation, rather than needing to re-zoom on each page.

- If this option is not set, then when a new section is displayed for a vector file, the view will be initialized to the defined VECFitMode, and a bitmap file will be initialized to the defined BMPFitMode and rotation will be reset to None. This is the same behavior as when opening a new graphic file.
- If this option is set, then new vector sections within a file will initialize to the current IMGZoom, and new bitmap sections will be initialized to the current IMGZoom and BMPRotation.

Type

BOOL

Default

True

Related to

SCCID_MAINTAINZOOM (Outside In Viewer)

PageFormatHeight

Read only. This property is filled with the formatted page height in logical units after a call to DrawPage, GetDrawPageInfo, or DrawPageEx (see DrawPage GetDrawPageInfo, and DrawPageEx).

Type

LONG

Default

0

Related to

SCCVW_GETDRAWPAGEINFO, SCCVW_DRAWPAGE (Outside In Viewer)

PageFormatWidth

Read only. This property is filled with the formatted page width in logical units after a call to DrawPage, GetDrawPageInfo, or DrawPageEx (see DrawPage GetDrawPageInfo, and DrawPageEx).

Type

LONG

Default

0

Related to

SCCVW_GETDRAWPAGEINFO, SCCVW_DRAWPAGE (Outside In Viewer)

PagePaletteHandle

This property is filled with a palette handle after a call to DrawPage (see DrawPage).

Type

OLE_HANDLE

Default

none

Related to

SCCVW_DRAWPAGE (Outside In Viewer)

PagePicture

Type

LPPICTUREDISP

Default

none

Related to

SCCVW_DRAWPAGE (Outside In Viewer)

Description

This property is filled with the Picture object after a call to DrawPage (see DrawPage).

PageResultBottom

Read only. Bottom coordinate of rectangle in device units that was actually filled when a page was drawn into the output device context. This property is only set by a call to DrawPageEx (see DrawPageEx).

Type

LONG

Default

0

Related to

SCCVW_GETDRAWPAGEINFO, SCCVW_DRAWPAGE (Outside In Viewer)

PageResultLeft

Read only. Left coordinate of rectangle in device units that was actually filled when a page was drawn into the output device context. This property is only set by a call to DrawPageEx (see DrawPageEx).

Type

LONG

Default

0

Related to

SCCVW_GETDRAWPAGEINFO, SCCVW_DRAWPAGE (Outside In Viewer)

PageResultRight

Read only. Right coordinate of rectangle in device units that was actually filled when a page was drawn into the output device context. This property is only set by a call to DrawPageEx (see DrawPageEx).

Type

LONG

Default

0

Related to

SCCVW_GETDRAWPAGEINFO, SCCVW_DRAWPAGE (Outside In Viewer)

PageResultTop

Read only. Top coordinate of rectangle in device units that was actually filled when a page was drawn into the output device context. This property is only set by a call to DrawPageEx (see DrawPageEx).

Type

LONG

Default

0

Related to

SCCVW_GETDRAWPAGEINFO, SCCVW_DRAWPAGE (Outside In Viewer)

PageUnitsPerInch

Read only. This property is filled with the units per inch that PageFormatHeight and PageFormatWidth properties (see PageFormatHeight, and PageFormatWidth) represent after a call to DrawPage, GetDrawPageInfo, or DrawPageEx (see DrawPage, GetDrawPageInfo, and DrawPageEx).

Type

LONG

Default

0

Related to

SCCVW_GETDRAWPAGEINFO, SCCVW_DRAWPAGE (Outside In Viewer)

ParseXMPMetadata

Adobe's Extensible Metadata Platform (XMP) is a labeling technology that allows you to embed data about a file, known as metadata, into the file itself. This option determines if XMP data will be parsed into normal OIT document properties. Enabling this option may cause the loss of some regular data in premium graphics filters (such as Postscript), but won't affect most formats (such as PDF).

One of the following values:

- TRUE: Parse the XMP data into normal OIT document properties.
- FALSE: Do not parse XMP data.

Type

BOOL

Default

FALSE

Related to

SCCID_PARSEXMPMETADATA (Outside In Viewer)

PrintCollate

This property, if TRUE, cause multiple copies of a printout to be collated. If FALSE, multiple copies will be printed together.

Type

BOOL

Default

FALSE

Related to

SCCID_PRINTCOLLATE

PrintCopies

This property indicates the number of copies of the file, selection, or range to print.

Type

int

Default

1

Related to

SCCID_PRINTCOPIES (Outside In Viewer)

PrintEndPage

This property indicates the page printing should end on. It is only valid if the WhatToPrint property (see WhatToPrint) has the value SCCVW_PRINT_PAGERANGE. This value resets automatically when a new file is loaded.

Type

int

Default

1

Related to

SCCID_PRINTENDPAGE (Outside In Viewer)

PrinterDC

This property holds the handle of the printer device context that may be used with the PrintOI method (see PrintOI), if the developer chooses to control printing in this way.

Type

OLE_HANDLE

Default

NULL

Related to

SCCVW_PRINTEX (Outside In Viewer)

PrinterDriver

This property is used with the PrinterName and PrinterPort properties (see PrinterName, and PrinterPort) to optionally specify a printer in the PrintOI method (see PrintOI).

Type

BSTR

Default

NULL

Related to

SCCVW_PRINTEX (Outside In Viewer)

PrinterName

This property contains the name for a printer. This may be used in the PrintOI method (see PrintOI), if the developer chooses to control printing in this way.

Type

BSTR

Default

NULL

Related to

SCCVW_PRINTEX (Outside In Viewer)

PrinterPort

This property is used in conjunction with the PrinterName property (see PrinterName) to specify a printer in the PrintOI method (see PrintOI).

Type

BSTR

Default

NULL

Related to

SCCVW_PRINTEX (Outside In Viewer)

PrintFont

This property allows the developer to set the default print font. The developer may implement a font dialog or set this value directly. If the developer chooses to

implement his own dialog, be advised that a default font will be used for any printer font selected from within the dialog.

Type

LPFONTDISP

Related to

SCCID_DEFAULTPRINTFONT (Outside In Viewer)

PrintJobName

This property is used to specify the text to be printed in the header on each page if printing of headers is enabled.

Type

BSTR

Default

Empty string

Related to

SCCID_PRINTJOBNAME (Outside In Viewer)

PrintMarginLeft/Right/Top/Bottom

These properties contain the print margins in centimeters (if IntlFlags is not set) or inches (if IntlFlags is set). If a value outside the range is assigned, the value will be clipped to the range.

Type

Float

Default

1 Inch (2.54 cm)

Range

0 to 4.23 Inches (0 to 10.76 cm)

Related to

SCCID_DEFAULTPRINTMARGINS (Outside In Viewer)

PrintStartPage

This property indicates the page on which printing should start. It is only valid if the WhatToPrint property (see WhatToPrint) has the value SCCVW_PRINT_PAGERANGE. This value resets automatically when a new file is loaded.

Type

int

Default

0

Related to

SCCID_PRINTSTARTPAGE (Outside In Viewer)

RawTextCharSet

Read only. This is a value that represents which character set the RawTextString (see RawTextString) is in. This should be the native character set for the platform or Unicode. Refer to the header files for character set values.

Type

Short

Default

0

Related to

SCCVW_GETRAWTEXT (Outside In Viewer)

RawTextLength

Read Only. This is a value that represents the size of the RawTextString (see RawTextString) buffer created when requested through GetRawText (see GetRawText).

Type

Long

Default

0

Related to

SCCVW_GETRAWTEXT (Outside In Viewer)

RawTextOffset

Read only. This is a value that represents the number of characters from the beginning of the document to the first character in the RawTextString (see RawTextString).

Type

Long

Default

0

Related to

SCCVW_GETRAWTEXT (Outside In Viewer)

RawTextString

Read only. This is the string that represents the raw text requested by GetRawText (see GetRawText).

Type

BSTR

Default

NULL

Related to

SCCVW_GETRAWTEXT (Outside In Viewer)

RenderEmbeddedFonts

This option allows you to disable the use of embedded fonts in PDF input files. One of the following values:

- TRUE: the embedded fonts in the PDF input will be used to render text.
- FALSE: the embedded fonts will not be used and the fallback is to use fonts available to Outside In to render text.

Type

BOOL

Default

TRUE

Related to

SCCOPT_RENDER_EMBEDDED_FONTS (Outside In Viewer)

ReorderMethod

This property indicates how the viewer reorders bidirectional text.

It can be one of the following values:

- SCCVW_REORDER_NOVALUE has been replaced by SCCUT_REORDER_UNICODE_OFF: No bidirectional support.
- SCCVW_REORDER_NONE is obsolete.

- SCCVW_REORDER_UNICODE has been replaced by SCCUT_REORDER_UNICODE_RTOL: Characters are reordered using the Unicode bidirectional algorithm in right-to-left order.
- SCCVW_REORDER_FULL is obsolete.
- SCCUT_REORDER_UNICODE_OFF: This disables any processing for unicode characters. This option is the default.
- SCCUT_REORDER_UNICODE_LTOR: Characters displayed using the Unicode bidirectional algorithm in left-to-right order.
- SCCUT_REORDER_UNICODE_RTOL: Characters displayed using the Unicode bidirectional algorithm in right-to-left order.

Type

Short

Default

SCCVW_REORDER_NONE

Related to

SCCID_REORDERMETHOD (Outside In Viewer)

ResourceLibraryID

This property allows multiple versions of the resource localization library to exist and to be changed programmatically. The default localization library name is SCCLO.DLL. To select a different library name, this property must be set to a string corresponding to characters added to this default name. For example, to choose the SCCLOXX.DLL file, ResourceLibraryID must be set to XX.

This addition is not limited to two characters, it may be 0 – 5 characters. If ResourceLibraryID is set to a zero-length string, it will revert to its default of LO.

Type

BSTR

Default

LO

Related to

SCCID_RESOURCELIBRARYID (Outside In Viewer)

SelectionAnchor

This property sets the anchor, or beginning, of the selection area. If the value for this property is invalid, it will not be set and will remain at its previous value. For more information, see Positions. This property is also populated with the start of user-selected text.

Type

OixPos

Default

none

Related to

SCCVW_SETSELECTION (Outside In Viewer)

SelectionEnd

This property sets the end position of the selection area. If the value for this property is invalid, it will not be set and will remain at its previous value. For more information, see Positions. This property is also populated with the ending position of user-selected text.

Type

OixPos

Default

none

Related to

SCCVW_SETSELECTION (Outside In Viewer)

SSClipBoard

Determines what format the database or spreadsheet takes when it is copied to the clipboard.

- SCCVW_CLIPSUBFORMAT_TABLE: In supporting clipboard formats (RTF & AMI), the form will be copied as a table.
- SCCVW_CLIPSUBFORMAT_OPTIMIZEDTABS: The form will be copied using a tab stop for each field, except when a field is empty.
- SCCVW_CLIPSUBFORMAT_TABS: The form will be copied using a tab stop for each field.

Type

Short

Default

SCCVW_CLIPSUBFORMAT_TABLE

Related to

SCCID_SSCLIPBOARD (Outside In Viewer)

SSDraftMode

Determines how much formatting information is included when displaying a database or spreadsheet.

- TRUE: A limited set of formatting information is included.
- FALSE: All supported formatting information is included.

Type

BOOL

Default

TRUE

Related to

SCCID_SSDRAFTMODE (Outside In Viewer)

SSPrintDirection

This option will determine the direction that the spreadsheet will be printed. This value resets when the view window is opened. The following values are valid:

- SCCVW_SSPRINTDIRECTION_ACROSS: Print spreadsheet across first, then down
- SCCVW_SSPRINTDIRECTION_DOWN: Print spreadsheet down first, then across

Type

Long

Default

SCCVW_SSPRINTDIRECTION_ACROSS

Related to

SCCID_SSPRINTSCALEXWIDE (Outside In Viewer)

SSPrintFit

Determines how the database or spreadsheet printouts are fitted to the paper size and default print margins.

If the scaling is too much (one-fifth the original size) it is limited to that amount and will continue on additional pages. Aspect ratio is maintained. The following values are valid:

- SCCVW_SSPRINTFITMODE_NOMAP: Printing will occur in its original size onto as many pages as are required to fit the data.
- SCCVW_SSPRINTFITMODE_SCALE: The spreadsheet will be scaled on the printed page using the scale value in the SSPrintScalePercent property (see SSPrintScalePercent). This value is valid for spreadsheets only.

- SCCVW_SSPRINTFITMODE_FITTOPAGES: The spreadsheet will be scaled on the printed page to fit to the number of pages specified in the SSPrintScaleXHigh and SSPrintScaleXWide (see SSPrintScaleXHigh, and SSPrintScaleXWide). This value is valid for spreadsheets only.
- SCCVW_SSPRINTFITMODE_FITTOPAGE: Fit to one page.
- SCCVW_SSPRINTFITMODE_FITTOWIDTH: Scale to one page width.
- SCCVW_SSPRINTFITMODE_FITTOHEIGHT: Scale to one page height.
- SCCVW_SSPRINTFITMODE_SCALE: Scale to value specified in SSPrintScalePercent (see SSPrintScalePercent).

Type

Short

Default

SCCVW_SSPRINTFITMODE_NOMAP

Related to

SCCID_SSPRINTFITTOPAGE (Outside In Viewer)

SSPrintGridLines

Determines if a dotted line is printed between fields. The following values are valid:

- TRUE: A dotted line is printed between fields.
- FALSE: A dotted line is not printed between fields.

Type

BOOL

Default

TRUE

Related to

SCCID_SSPRINTGRIDLINES (Outside In Viewer)

SSPrintHeadings

Determines if field headings will be printed along with the data. The following values are valid:

- TRUE: Field headings will be printed.
- FALSE: Field headings will not be printed.

Type

BOOL

Default

TRUE

Related to

SCCID_SSPRINTHEADINGS (Outside In Viewer)

SSPrintScalePercent

This option will scale the spreadsheet pages by the percentage specified. Values of 30 to 300% are valid.

Type

Long

Default

100

Related to

SCCID_SSPRINTSCALEPERCENT (Outside In Viewer)

SSPrintScaleXHigh

This option will fit the printing to the number of vertical pages specified. This value resets when the view window is opened.

Type

Long

Default

1

Related to

SCCID_SSPRINTSCALEXHIGH (Outside In Viewer)

SSPrintScaleXWide

This option will fit the printing to the number of horizontal pages specified.

Type

Long

Default

1

Related to

SCCID_SSPRINTSCALEXWIDE (Outside In Viewer)

SSRowColNamesToClip

This property has been deprecated.

SSShowGridLines

Determines if a dotted line is displayed between fields. The following values are valid:

- TRUE: A dotted line is displayed between fields.
- FALSE: A dotted line is not displayed between fields.

Type

BOOL

Default

TRUE

Related to

SCCID_SSSHOWGRIDLINES (Outside In Viewer)

SSShowHiddenCells

Determines whether hidden rows and columns will be shown when displaying a spreadsheet. This can be one of the following values:

- TRUE: Show hidden cells.
- FALSE: Do not show hidden cells

Type

BOOL

Default

FALSE

Related to

SCCID_SSSHOWHIDDENCELLS (Outside In Viewer)

StatusEvents

This option determines if Status events are to be generated by the control, and if they are working.

Use of the Status Callback Function

An application's status callback function will be called periodically by Outside In to provide a status message. Currently, the only status message defined is OIT_STATUS_WORKING, which provides a "sign of life" that can be used during unusually long processing operations to verify that Outside In has not stopped working. If the application decides that it would not like to continue processing the

current document, it may use the return value from this function to tell Outside In to abort.

The status callback function has two return values defined:

- OIT_STATUS_CONTINUE: Tells Outside In to continue processing the current document.
- OIT_STATUS_ABORT: Tells Outside In to stop processing the current document.

The following is an example of a minimal status callback function.

```
VTDWORD MyStatusCallback( VTHANDLE hUnique, VTDWORD dwID, VTSYSVAL
pCallbackData, VTSYSVAL pAppData)
{
    if(dwID == OIT_STATUS_WORKING)
    {
        if( checkNeedToAbort( pAppData ) )
            return (OIT_STATUS_ABORT);
    }

    return (OIT_STATUS_CONTINUE);
}
```

Type

Bool

Default

False

Related to

VWSetStatCallback (Outside In Viewer)

StrokeTest

This option is used to stroke out (display as graphical primitives) text in an AutoCAD file. Setting this option to FALSE would improve performance, but the visual fidelity may be compromised. One of the following values:

- TRUE: Stroke out text in the AutoCAD file.
- FALSE: Use system fonts to render the text.

Type

BOOL

Default

TRUE

Related to

SCCOPT_RENDER_EMBEDDED_FONTS (Outside In Viewer)

SysLotusNotesPath

This property allows the developer to specify the location of a Lotus Notes or Domino installation for use by the NSF filter. A valid Lotus installation directory must contain the file nnotes.dll.

Note:

This option is Global in scope.

Type

String

Default

If this option isn't set, then OIT will first attempt to load the Lotus library according to the operating system's PATH environment variable, and then attempt to find and load the Lotus library as indicated in HKEY_CLASSES_ROOT\Notes.Link.

Related to

SCCID_LOTUSNOTESPATH (Outside In Viewer)

SystemRawText

If set to TRUE then the developer will receive the RawTextEvent (see RawTextEvent) as new text becomes available.

Note:

This option is Global in scope.

Type

BOOL

Default

FALSE

Related to

SCCID_SYSTEMFLAGS (Outside In Viewer)

SystemReadAhead

If this property is set to FALSE, then the control's process that reads through the rest of the file in the background is disabled. When the property is set to FALSE, the file will be read on demand as the user scrolls down through the document. This will cause unusual scroll bar behavior as the user scroll down.

Note:

This option is Global in scope.

Type

BOOL

Default

TRUE

Related to

SCCID_SYSTEMFLAGS (Outside In Viewer)

SystemTimer

If set to FALSE, then it disables the control's internal timer that controls background reading, caret blinking and auto scroll. If this is set to TRUE, the developer must call the SystemIdle method (see SystemIdle) for the control to perform any background operations. It is recommended that this property be left set to its default of TRUE.

Note:

This option is Global in scope.

Type

BOOL

Default

TRUE

Related to

SCCID_SYSTEMFLAGS (Outside In Viewer)

SystemUnicode

If this property is set to TRUE, then Unicode will be enabled. This affects the methods GetRawText, Search, and Get AnnotationData (see GetRawText, Search, and GetAnnotationData).

Note:

This option is Global in scope.

Type

long

Default

FALSE

Related to

SCCID_SYSTEMFLAGS (Outside In Viewer)

TimeZoneOffset

This property controls the offset to GMT that is applied during date formatting, allowing date values to be displayed in a selectable time zone. The option defaults to the current GMT time display. A flag tells the technology to query the OS and use the OS's defined timezone.

This option affects the formatting of numbers that have been defined as date values. This option will not affect dates that are stored as text.

Note:

Daylight savings is not supported. The sent time in msg files when viewed in Outlook can be an hour different from the time sent when an image of the msg file is created.

Scope

Global

Type

Short - Integer Parameter from -96 to 96, representing 15 minute offsets from GMT, or SCC_TIMEZONE_USENATIVE, which will query the OS for the timezone set on the machine

Default

0 : GMT Time

Related to

SCCID_TIMEZONE (Outside In Viewer)

ToClipboard

This property controls the format that the viewer copies to the clipboard when the CopyToClip method (see CopyToClip) is called. This property can contain one or more of the following values OR-ed together:

- SCCVW_CLIPFORMAT_TEXT: Text in whatever character set is appropriate for the operating system
- SCCVW_CLIPFORMAT_RTF: Rich Text Format
- SCCVW_CLIPFORMAT_UNICODE: Unicode text format
- SCCVW_CLIPFORMAT_WINBITMAP: Windows Bitmap
- SCCVW_CLIPFORMAT_WINDIB: Windows Device Independent Bitmap

- SCCVW_CLIPFORMAT_WINMETAFILE: Windows Metafile
- SCCVW_CLIPFORMAT_WINPALETTE: Windows Palette

Type

ULONG

Default

SCCVW_CLIPFORMAT_WINMETAFILE

Related to

SCCID_TOCLIPBOARD (Outside In Viewer)

UnmappableChar

This property defines what replacement character to use when the viewer cannot find the character to be displayed in any font on the system.

This value is the Unicode value for the replacement character.

Type

Short

Default

0x002a - '*'

Related to

SCCID_UNMAPPABLECHAR (Outside In Viewer)

UseDocPageSettings

Used to select the document's page layout information when printing.

Type

BOOL

Default

TRUE

Related to

SCCID_USEDOCPAGESETTINGS (Outside In Viewer)

Data

- TRUE: The document's native page margins and paper size are used when available from the filter.
- FALSE: The printer's page margins and paper size are used instead of those of the native document.

VecFitMode

Determines how the display engine (vector or bitmap) displays (stretches/shrinks) an image in relation to the size of the view window. One of the following values:

- SCCVW_FITMODE_IMAGESIZE: Scale to the image size.
- SCCVW_FITMODE_ORIGINAL: The size of the window has no effect.
- SCCVW_FITMODE_WINDOW: The image will be stretched to fill as much of the window as possible while maintaining its proper aspect ratio.
- SCCVW_FITMODE_WINDOWHEIGHT: The image will be stretched so its full height fits in the height of the window. Depending on the image, its full width may or may not fit inside the window.
- SCCVW_FITMODE_WINDOWWIDTH: The bitmap will be stretched so its full width fits in the width of the window.
- SCCVW_FITMODE_STRETCHWINDOW: The image will be stretched to fill the window. The image's aspect ratio is NOT maintained.
- SCCVW_FITMODE_BEST: If the view window is smaller than the original image, this option will fit it to the window. If the view window is larger than the original image, the image will be displayed at its original size.

Invalid values will be reset to the last set value.

Type

Short

Default

SCCVW_FITMODE_ORIGINAL

Related to

SCCID_VECFITMODE (Outside In Viewer)

VecPrintAspect

The option indicates how the image will be stretched when printed. It can be one of the following values:

- SCCVW_PRINTASPECT_SCALE (previously referred as SCCVW_PRINTASPECT_ORIGINAL): The image will be sized up to fill as much of the area inside the print margins as possible while still maintaining the original aspect ratio.
- SCCVW_PRINTASPECT_STRETCH: The image will be stretched horizontally and vertically to totally fill the area inside the print margins.

Invalid values will be reset to the last set value.

Type

Short

Default

SCCVW_PRINTASPECT_SCALE (previously referred as SCCVW_PRINTASPECT_ORIGINAL)

Related to

SCCID_VECPRINTASPECT (Outside In Viewer)

VecPrintBackground

If this property is set to TRUE, then the background of the vector image will be printed. If it is set to FALSE, then the background will not be printed.

Type

BOOL

Default

TRUE

Related to

SCCID_VECPRINTBACKGROUND (Outside In Viewer)

VecPrintBorder

Determines if a one-pixel border is to be printed around images. One of the following values:

- TRUE: Print a one-pixel border.
- FALSE: Do not print a one-pixel border.

Type

BOOL

Default

TRUE

Related to

SCCID_VECPRINTBORDER (Outside In Viewer)

VecShowBackground

If TRUE, the background of a vector image as defined in the file will be displayed. If it is FALSE, the background will not be displayed.

Type

BOOL

Default

TRUE

Related to

SCCID_VECSHOWBACKGROUND (Outside In Viewer)

VScrollbar

This property controls the visibility of the vertical scroll bar. TRUE means that the scrollbar is always visible, but may be disabled if unnecessary. A FALSE value makes the scrollbar invisible.

The Outside In Viewer does not have the ability to automatically show/hide scrollbars. This property takes the place of the ScrollFlags property because there are only two real values for each scrollbar.

Type

BOOL

Default

TRUE

Related to

SCCID_SCROLLFLAGS (Outside In Viewer)

WhatToPrint

This option indicates how much of a file should be printed when the PrintOI method (see PrintOI) is called. It can be one of the following values. This value resets automatically when a new file is loaded.

Type

ULONG

Default

0

Related to

SCCID_WHATTOPRINT (Outside In Viewer)

Data

- SCCVW_PRINT_ALLPAGES: The entire document will be printed
- SCCVW_PRINT_SELECTION: Just the selected area will be printed
- SCCVW_PRINT_PAGERANGE: The pages in the range PrintStartPage and PrintEndPage (see PrintStartPage, and PrintEndPage) will be printed.
- SCCVW_PRINT_CURSECTION: The current section will be printed.

WPDisplayMode

This property indicates how the word processor display engine displays the document. This does not apply to Email and HTML files. It can be one of the following:

- SCCVW_WPMODE_DRAFT: The document is displayed using only a single font and size (determined by the DisplayFont property. For more information, see DisplayFont). It does not display embedded graphics, graphic borders, or table borders. Text is wrapped to the size of the view window.
- SCCVW_WPMODE_NORMAL: All supported formatting is displayed. The text is wrapped to the size of the view window.
- SCCVW_WPMODE_PREVIEW: All supported formatting is displayed. The text is wrapped as it will be printed.
- SCCVW_WPMODE_WEBLAYOUT: Display all supported formatting, wrap the text as it would appear in a browser.

Type

Short

Default

SCCVW_WPMODE_PREVIEW

Related to

SCCID_WPDISPLAYMODE (Outside In Viewer)

WPEmailHeaderOutputMode

This property controls viewing and printing of email headers.

Note:

The property WPMimeHeaderOutput has been deprecated, and this takes its place.

Note:

This option is Global in scope.

Data

One of these values:

- SCCUT_WP_EMAILHEADERSTANDARD: Displays a standardized set of headers based on the type of document being viewed. These are:
  - Emails - "To", "From", "Subject", "CC", "BCC", "Date Sent", "Importance", and "Attachments"

- Journal Entries - "Entry Type", "Subject", "Start", "Duration", "Company", "Categories", and "Attachments"
- Tasks - "Subject", "Start Date", "Due Date", "Status", "Priority", "Percent Complete", "Owner", "Categories", and "Attachments"
- Contacts - "First Name", "Family Name", "Middle Name", "Title", "Suffix", "Job Title", "File As", "Email", "Business Phone", "Home Phone", "Business Address", "Home Address", "Mobile Phone", "Business Fax", "IM Address", "Company", "Webpage", and "Attachments"
- Appointments - "To", "Location", "Subject", "Start Time", "End Time", "Required Attendee", "Optional Attendee", "Importance", "Categories", and "Attachments"
- Posts - "From", "Creation Time", "Subject", "Conversation Topic", and "Attachments"
- Notes - "Creation Time", "Categories", and "Attachments"
- Distribution Lists - "Subject" and "Attachments".

- SCCUT_WP_EMAILHEADERALL: Displays all available email headers.
- SCCUT_WP_EMAILHEADERNONE : Display no email headers.
- SCCUT_WP_EMAILHEADERCUSTOM : Custom headers are displayed (Cannot use this to value set this property - but the control would return this value if custom headers are being used).

Type

Long

Default

SCCUT_WP_EMAILHEADERSTANDARD

Related to

SCCID_WPEMAILHEADEROUTPUT (Outside In Viewer)

WPFitMode

This property controls the size of word processor pages when using the preview mode. One of the following values:

- SCCVW_FITMODE_ORIGINAL: Sizes the preview page to the actual size.
- SCCVW_FITMODE_WINDOW: Sizes the preview page to the window.
- SCCVW_FITMODE_WINDOWWIDTH: Sizes the preview page to the width of the window.

Type

Short

Default

SCCVW_FITMODE_ORIGINAL

Related to

SCCID_WPFITMODE (Outside In Viewer)

WPWrapToWindow

This property indicates how the word processing display engine wraps text in the window in Normal and Draft modes. Text always wraps to the page size in preview mode. If the option is TRUE (default) the text will wrap to fit the window. If the option is FALSE, text will not wrap and the horizontal scroll bar will be available for scrolling to the unwrapped text.

Type

BOOL

Default

TRUE

Related to

SCCID_WRAPTOWINDOW (Outside In Viewer)

Methods

The following methods are used in the SDK.

AddAnnotationHideText

This method allows the developer to remove a section of text from the viewed document.

Usage

```
Void AddAnnotationHideText (long lId, VARIANT varData, OixPos StartPos, OixPos
EndPos)
```

Returns

Void

Parameters

- lId: Unique positive number that may be used later to remove, jump to, or otherwise identify the particular annotation or set of annotations. Initialize this value to 0 or a negative number if you are not using this functionality. This parameter is required. There are no invalid values. Negative values are not recommended because they may conflict with the hyperlink and bookmark masks.
- varData: Additional user data associated with the annotation.
- lStartPos: Starting text position for the hidden area expressed as a position object. If the value is invalid, then the method will generate an error and not create an

annotation. If the value is not passed, then it will use the currently selected text. If no text is selected, then no annotation will be placed.

- lEndPos: Ending text position for the hidden area expressed as a position object. If the value is invalid, then the method will generate an error and not create an annotation. If the value is not passed, then it will use the currently selected text. If no text is selected, then no annotation will be placed.

Related to

SCCVW_ADDANNOTATION (Outside In Viewer)

AddAnnotationHilite

This method allows the developer to add hilited text to the viewed document.

Usage

```
Void AddAnnotationHilite( long lId, short istyle, short iEvent, short
iBackground, short iForeground, VARIANT varData, OixPos StartPos, OixPos EndPos)
```

Returns

Void

Parameters

- lId: Unique number that may be used later to remove, jump to, or otherwise identify the particular annotation or set of annotations. Initialize this value to 0 or a negative number if you are not using this functionality. This parameter is required. The use of the high bit is reserved for internal annotation tagging.
- iStyle: Set the hilite to a previously defined style from the method HiliteStyle (see HiliteStyle). If this parameter is set to zero, then the Background and Foreground values will be used to determine the hilite style. If this value is negative, then an error will occur.
- iEvent: Indicates the type of events the developer would like to receive for this annotation. The following values may be combined. If the value is 0 the developer will receive no events:
  - SCCVW_EVENTDOUBLECLICK: Double click on Annotation
  - SCCVW_EVENTSINGLECLICK: Single click on Annotation
  - SCCVW_EVENTSINGLERIGHTCLICK : Single right click on Annotation
  - SCCVW_EVENTTRANSITIONINTO: Transition Cursor into the Annotation
  - SCCVW_EVENTTRANSITIONOUTOF: Transition Cursor out of the Annotation

  This parameter is required. There are no invalid values.
- iBackground: Set to a value to determine the text background color. The following values are valid:
  - OIX_BDEFAULT: Do not change background color
  - OIX_BBLACK: Black background

- OIX_BDARKRED: Dark Red background
- OIX_BDARKGREEN: Dark Green background
- OIX_BDARKYELLOW: Dark Yellow background
- OIX_BDARKBLUE: Dark Blue background
- OIX_BDARKMAGENTA: Dark Magenta background
- OIX_BDARKCYAN: Dark Cyan background
- OIX_BLIGHTGRAY: Light Gray background
- OIX_BGRAY: Gray background
- OIX_BRED: Red background
- OIX_BGREEN: Green background
- OIX_BYELLOW: Yellow background
- OIX_BBLUE: Blue background
- OIX_BMAGENTA: Magenta background
- OIX_BCYAN: Cyan background
- OIX_BWHITE: White background

If the value is invalid, the method will generate an error and not create an annotation. If the style parameter is non-zero, then this parameter will be ignored.

- iForeground: Set to a value to determine the text foreground color.
  - OIX_FDEFAULT: Do not change foreground color
  - OIX_FBLACK: Black foreground
  - OIX_FDARKRED: Dark Red foreground
  - OIX_FDARKGREEN: Dark Green foreground
  - OIX_FDARKYELLOW: Dark Yellow foreground
  - OIX_FDARKBLUE: Dark Blue foreground
  - OIX_FDARKMAGENTA: Dark Magenta foreground
  - OIX_FDARKCYAN: Dark Cyan foreground
  - OIX_FLIGHTGRAY: Light Gray foreground
  - OIX_FGRAY: Gray foreground
  - OIX_FRED: Red foreground
  - OIX_FGREEN: Green foreground
  - OIX_FYELLOW: Yellow foreground
  - OIX_FBLUE: Blue foreground

- OIX_FMAGENTA: Magenta foreground
- OIX_FCYAN: Cyan foreground
- OIX_FWHITE: White foreground

If the value is invalid, then the method will generate an error and not create an annotation. If the style parameter is non-zero, then the value of this parameter will be ignored.

- varData: Additional user data associated with the annotation. This parameter is not required.
- StartPos: Starting text position for the hilited area expressed as a position object (see Positions). If the value is invalid, then the method will generate an error and not create an annotation. If the value is not passed, then it will use the currently selected text. If no text is selected, then no annotation will be placed.
- EndPos: Ending text position for the hilited area expressed as a position object. If the value is invalid, then the method will generate an error and not create an annotation. If the value is not passed, then it will use the currently selected text. If no text is selected, then no annotation will be placed.

Related to

SCCVW_ADDANNOTATION. (Outside In Viewer)

AddAnnotationInsertText

This method allows the developer to insert text in the viewed document. It is only valid when viewing word processing, spreadsheet, database or metafile documents.

Usage

```
Void AddAnnotationInsertText( long lId, VARIANT varData, OixPos InsertPosition,
VARIANT varInsertText)
```

Returns

Void

Parameters

- lId: Unique number that may be used later to remove, jump to or otherwise identify the particular annotation or set of annotations. Initialize this value to 0 or a negative number if you are not using this functionality. This parameter is required. There are no invalid values.
- varData: Additional user data associated with the annotation. This parameter can be NULL.
- InsertPosition: The position at which the text is to be inserted.
- varInsertText: The UNICODE text to be inserted. This is passed by reference.

Related to

SCCVW_ADDANNOTATION (Outside In Viewer)

AddAnnotationPicture

This method allows the developer to insert a bitmap or icon in the viewed document.

Usage

```
Void AddAnnotationPicture( long lId, LPPICTUREDISP pdPicture, short iEvent,
VARIANT varData, OixPos Pos)
```

Returns

Void

Parameters

- lId: Unique positive number that may be used later to remove, jump to, or otherwise identify the particular annotation or set of annotations. Initialize this value to 0 or a negative number if you are not using this functionality. This parameter is required. There are no invalid values.
- pdPicture: Pointer to an IPictureDisp (Picture Object) interface that contains the image to be inserted into the viewed document. This parameter is required. If the Picture is an Icon, then an icon will be inserted into document. Otherwise, a bitmap will be inserted into the document. If the Picture is invalid, then an error will be generated and no annotation will be placed.
- iEvent: Indicates the type of events the developer would like to receive for this annotation. These values may be combined. If the value is 0 the developer will receive no events. The following values are valid:
  - SCCVW_EVENTSINGLECLICK: Single click on annotation
  - SCCVW_EVENTDOUBLECLICK: Double click on annotation
  - SCCVW_EVENTSINGLERIGHTCLICK : Single right click on Annotation
  - SCCVW_EVENTTRANSITIONINTO: Transition cursor into the annotation
  - SCCVW_EVENTTRANSITIONOUTOF: Transition cursor out of the annotation

  This parameter is required. If the value is invalid, then the method will generate an error and not create an annotation.
- varData: Text associated with this annotation. Allows the developer to easily associate text with the annotation. This parameter is not required. If the parameter is not passed then no text will be associated with the annotation. Any attempt to retrieve the text will return a zero-length string. If a zero length string or a NULL string pointer is passed, then no text will be associated with the annotation. There are no invalid values.
- Pos: Text position for the annotation expressed as a position object. If the value is invalid, then the method will fire an error and not create an annotation. If the value is not passed, then it will use the current caret position. If there is no current position, then no annotation will be placed.

Related to

SCCVW_ADDANNOTATION (Outside In Viewer)

AnnotationSetPos

This method sets the caret to the position of the last annotation added or found. Optionally, it can select the text the annotation hilites or hides. This method is more accurate than using the Start and End positions to set the location. If there isn't a current annotation Start and End position, then nothing happens.

Usage

```
Void AnnotationSetPos( bool bSelect )
```

Returns

Void

Parameters

- bSelect: If this is TRUE, then the annotated document text is selected. Otherwise, the caret is placed at the start of the annotation. For picture annotations, this parameter has no effect. This parameter is required and has no invalid values.

Related to

SCCVW_DISPLAYPOSITION (Outside In Viewer)

ArchiveSave

Initiates a save when dealing with an archive document.

Usage

```
Void ArchiveSave(short nSaveType)
```

Returns

Void

Parameters

nSaveType can be one of the following values:

- SCCVW_ARCHIVE_SAVESELECTION: Saves the selected files to the directory of choice.
- SCCVW_ARCHIVE_SAVEALL: Saves all the files to the directory of choice.

Related to

SCCID_ARCSAVEEVENT (Outside In Viewer)

Clear

Use this method to close the currently viewed file and reset the control window to its non-viewing state.

Usage

```
void Clear()
```

Related to

SCCVW_CLOSEFILE (Outside In Viewer)

ClearAnnotations

This message allows the developer to selectively remove annotations.

Usage

```
Void ClearAnnotations( short iMask, long lRemoveId )
```

Returns

Void

Parameters

- iMask: This must be set to one of the following values:
  - OIX_CLEARANNO_ALL
  - OIX_CLEARANNO_MASK
  - OIX_CLEARANNO_ABSOLUTE

  If RemoveId is used as a mask to remove annotations, then any annotation for which Id & RemoveId = RemoveId will be removed. This parameter is required. If it is invalid, then an error will be generated and no annotations will be removed.
- lRemoveId: This parameter specifies which annotations to remove in conjunction with the mask. RemoveId must be a positive number if the value for the Mask parameter is OIX_CLEARANNO_ALL or OIX_CLEARANNO_MASK. This parameter is required. If the value is invalid (a negative number with Mask of OIX_CLEARANNO_ALL or OIX_CLEARANNO_MASK), then an error will be generated and no annotations will be removed.

Related to

SCCVW_CLEARANNOTATIONS (Outside In Viewer)

ClipboardOptions

Displays a dialog that lets the user set a number of aspects of how files are copied to the clipboard.

Usage

```
Short ClipboardOptions()
```

Returns

- OIX_DIALOG_OK: Dialog successful
- OIX_DIALOG_CANCEL: Dialog canceled by user

Parameters

none

Related to

SCCVW_DODIALOG (Outside In Viewer)

ComparePositions

This method allows the developer to compare two Position objects. The return value is -1 if PositionA is less than the PositionB, 0 if the two positions are equal, and 1 if PositionA is more than PositionB.

Usage

```
Short ComparePositions(OixPos PositionA, OixPos PositionB)
```

Returns

- -1: if PositionA is less than PositionB
- 0: if PositionA equals PositionB
- 1: if PositionA is greater than PositionB

Parameters

- PositionA: This value indicates the previously set position object to use in the comparison.
- PositionB: This value indicates the previously set position object to use in the comparison. If either PositionA or PositionB is invalid, an error will occur.

Related to

SCCVW_COMPPOSITIONS (Outside In Viewer)

Copy

This method allows the developer to copy the area defined by the two position indices into memory in a given format. The return value is TRUE if the method is successful, otherwise the value is FALSE.

Usage

```
Void Copy(long lFormatId, OixPos StartPos, OixPos EndPos)
```

Returns

Void

Parameters

- lFormatId: This parameter indicates the output format. It must be one of the following values:

- SCCVW_CLIPFORMAT_TEXT: Clipboard Text, text in whatever character set is appropriate for the operating system.
- SCCVW_CLIPFORMAT_RTF: Rich Text Format

This parameter is required. If the parameter is invalid, then an error will occur.

- StartPos: This parameter indicates the starting position object of the area to be copied. This parameter is required. If the parameter is invalid, then an error will occur.
- EndPos: This parameter indicates the ending position object of the area to be copied. This parameter is required. If the parameter is invalid, then an error will occur.

Related to

SCCVW_COPY (Outside In Viewer)

CopyPosition

Use this method to set the value of one position to the value of another position.

Usage

```
Void CopyPosition(OixPos DestPos, OixPos SourcePos)
```

Returns

Void

Parameter

- DestPos: This is a number that designates the position object to copy into.
- SourcePos: This is a number that designates the position object to copy from.

CopyToClip

This method cause the viewer to copy the current selection (if any) to the clipboard.

Usage

```
Void CopyToClip()
```

Returns

Void

Parameters

none

Related to

SCCVW_COPYTOCLIP (Outside In Viewer)

DeinitDrawPage

This function cleans up the memory allocated by InitDrawPage (see InitDrawPage). After calling DeinitDrawPage, InitDrawPage must be called again before DrawPage or GetDrawPageInfo can be called (see DeinitDrawPage, InitDrawPage, DrawPage, and GetDrawPageInfo).

Usage

```
Void DeinitDrawPage()
```

Returns

Void

Related to

SCCVW_DEINITDRAWPAGE (Outside In Viewer)

DisplayOptions

Displays a dialog that lets the user set a number of aspects of how files are viewed.

Usage

```
BOOL DisplayOptions()
```

Returns

- OIX_DIALOG_OK: Dialog successful.
- OIX_DIALOG_CANCEL: Dialog canceled by user.

Related to

SCCVW_DODIALOG (Outside In Viewer)

DisplayPosition

This method allows the developer to bring any Position into view and provides some control over the placement of the position object in the view.

Usage

```
Void DisplayPosition(OixPos Position, short iDisplayOptions)
```

Returns

Void

Parameters

- Position: This is the Position object to use to adjust the display.
- iDisplayOptions: This option determines where in the display the indicated position should appear. If it is set to some other value or not passed at all, then the

control will display the position with as little scrolling of the screen as possible. The options are:

- SCCVW_DISPLAYNEARTOP: The position will be displayed near the top of the view.
- SCCVW_DISPLAYNEARMIDDLE: The position will be displayed near the middle of the view.
- SCCVW_DISPLAYNEARBOTTOM: The position will be displayed near the bottom of the view.

Related to:

SCCVW_DISPLAYPOSITION (Outside In Viewer)

DrawPage

This method will draw a specified page and return that page as a Picture Object in the PagePicture property (see PagePicture). If the sizing information is not provided, appropriate defaults will be used in conjunction with SCCVW_GETDRAWPAGEINFO to create the image. The InitDrawPage method (see InitDrawPage) must be called before attempting DrawPage invocation (see DrawPage).

Usage

```
BOOL DrawPage (long lPageNumber, long lTop, long lLeft, long lBottom, long
lRight, long lFormatWidth, long lFormatHeight, short iUnitsPerInch)
```

Returns

Boolean indicating success.

Parameters

- lPageNumber: The zero-based page number to draw into the picture. The parameter is required. If the parameter is invalid, then an error will be fired and the operation will abort.
- lTop, lLeft, lBottom, lRight: These parameters specify a rectangle in HIMETRIC units (.01 mm) that is equivalent to FormatHeight and FormatWidth. If these parameters indicate a zero width or zero height rectangle, then an error will be fired and the operation will be aborted.
- lFormatWidth: This is the width of the wrappable part of the page. For instance, if your page is 8.5 inches wide and you want .5 inch margins, then FormatWidth should represent 7.5 inches. If the parameter is zero or negative, then an error will be fired and the operation will be aborted.
- lFormatHeight: This is the height of the wrappable part of the page. For instance, if your page is 11 inches wide and you want .5 inch margins, the FormatHeight should represent 10 inches. If the parameter is zero or negative, then an error will be fired and the operation will be aborted.
- iUnitsPerInch: This is the number of units per inch that FormatWidth and FormatHeight are in. For instance, if FormatHeight and FormatWidth are in twips, then UnitsPerInch would be 1440. If UnitsPerInch is zero or less, then an error will be fired and the operation will abort.

Related to

SCCVW_DRAWPAGE (Outside In Viewer)

DrawPageEx

This method draws a page of the currently viewed file to a rectangle and device of the developer's choice. The rectangle that is actually filled with by the rendered output is placed in the properties PageResultTop, PageResultLeft, PageResultBottom, PageResultRight (see PageResultTop, PageResultLeft, PageResultBottom, and PageResultRight).

Usage

```
BOOL DrawPageEx(long lPageNumber, long lFlags, long lTop, long lLeft, long
lBottom, long lRight, long lFormatHeight, long lFormatWidth, short
iUnitsPerInch, OLE_HANDLE hOutputDC, OLE_HANDLE hFormatDC, OLE_HANDLE hPalette)
```

Returns

- TRUE: Indicates Success
- FALSE: The page was not drawn

Parameters

- lPageNumber: The zero-based page number to draw into the picture. The parameter is required. If the parameter is invalid, then an error will be generated and the operation will abort.
- lFlags: This parameter contains options for the how the page is drawn. The values may be OR-ed together to select more than one option at a time.
  - SCCVW_DPFLAG_RETURNPALETTE: Return a Palette in the PagePaletteHandle property (see PagePaletteHandle).
  - SCCVW_DPFLAG_NOTMETAFILE: Assume hOutputDC is not a metafile DC, even if Windows says it is.
  - SCCVW_DPFLAG_IGNOREBACKGROUND: Ignore the background when rendering the page.
  - SCCVW_DPFLAG_DETERMINEOUTPUTTYPE: Attempt to determine the output device context type at runtime and emulate either a screen or printer. By default, draw page will emulate a printer. This is useful for enabling transparency effects when using creating a metafile.

  This parameter is required. There are no invalid values.
- lTop, lLeft, lBottom, lRight: These parameters specify a rectangle in HIMETRIC units (.01 mm) that is equivalent to lFormatHeight and lFormatWidth. If these parameters indicate a zero width or zero height rectangle, then an error will be generated and the operation will be aborted.
- lFormatWidth: This is the width of the wrappable part of the page. For instance, if your page is 8.5 inches wide and you want .5 inch margins, then lFormatWidth should represent 7.5 inches. If the parameter is zero or negative, then an error will be generated and the operation will be aborted.

- lFormatHeight: This is the height of the wrappable part of the page. For instance, if your page is 11 inches wide and you want .5 inch margins, the lFormatHeight should represent 10 inches. If the parameter is zero or negative, then an error will be generated and the operation will be aborted.
- iUnitsPerInch: This is the number of units per inch that FormatWidth and FormatHeight are in. For instance, if FormatHeight and FormatWidth are in twips, then UnitsPerInch would be 1440. If UnitsPerInch is zero or less, then an error will be generated and the operation will abort.
- hOutputDC: This is a handle to the device context to draw into. This parameter is required and an invalid value will cause an error.
- hFormatDCA: This is a handle to a device (or information) context used to wrap text and retrieve other formatting information. This may not be the same as hOutputDC. This parameter is required and an invalid value will cause an error.
- hPalette: The handle to the palette to be used for drawing the picture. An invalid value will cause an error.

Related to

SCCVW_DRAWPAGE (Outside In Viewer)

ExtDrawPage

This method will draw a specified page and return that page as a Picture Object in the PagePicture property. If the sizing information is not provided, appropriate defaults will be used in conjunction with SCCVW_GETDRAWPAGEINFO to create the image. The InitDrawPage method must be called before attempting DrawPage invocation.

Usage

```
BOOL ExtDrawPage (long lPageNumber, long lTop, long lLeft, long lBottom, long
lRight, long lFormatWidth, long lFormatHeight, short iUnitsPerInch, short
sFormatID)
```

Returns

Boolean indicating success.

Parameters

- lPageNumber: The zero-based page number to draw into the picture. The parameter is required. If the parameter is invalid, then an error will be fired and the operation will abort.
- lTop, lLeft, lBottom, lRight: These parameters specify a rectangle in HIMETRIC units (.01 mm) that is equivalent to FormatHeight and FormatWidth. If these parameters indicate a zero width or zero height rectangle, then an error will be fired and the operation will be aborted.
- lFormatWidth: This is the width of the wrappable part of the page. For instance, if your page is 8.5 inches wide and you want .5 inch margins, then FormatWidth should represent 7.5 inches. If the parameter is zero or negative, then an error will be fired and the operation will be aborted.

- lFormatHeight: This is the height of the wrappable part of the page. For instance, if your page is 11 inches wide and you want .5 inch margins, the FormatHeight should represent 10 inches. If the parameter is zero or negative, then an error will be fired and the operation will be aborted.
- iUnitsPerInch: This is the number of units per inch that FormatWidth and FormatHeight are in. For instance, if FormatHeight and FormatWidth are in twips, then UnitsPerInch would be 1440. If UnitsPerInch is zero or less, then an error will be fired and the operation will abort.
- sFormatID: This should be one of the following:
  - OIX_DRAWPAGE_FORMAT_WMF: Draw the specified page as a Windows Metafile.
  - OIX_DRAWPAGE_FORMAT_EMF: Draw the specified page as an Enhanced Windows Metafile.

Related to

SCCVW_DRAWPAGE (Outside In Viewer)

FindAnnotation

This method allows the developer to find an annotation and retrieve information associated with the annotation. If no matching annotations are found, then the method returns FALSE, else the method returns TRUE. The information for the found annotation can be retrieved from the various Annotation properties.

Usage

```
bool FindAnnotation( short iFindType, short iMask, long lFindId, OixPos
SearchPos )
```

Returns

Boolean indicating success.

Parameters

- iFindType: Indicates the type of find operation to be performed. The following values are valid:
  - OIX_FINDFIRST: Find first or only matching annotation.
  - OIX_FINDLAST: Find last or only matching annotation.
  - OIX_FINDPREV: Find previous matching annotation.
  - OIX_FINDNEXT: Find next matching annotation.

  This parameter is required. If the value is invalid, then the method will generate an Error event (see Error) and the operation will abort.
- iMask: Indicates how the annotations will be matched. The following values are valid:
  - OIX_ANNOTATION_ALL: Find all annotations - this includes internal annotations.

- OIX_MASK: Use FindId as a mask to find annotations.
- OIX_ABSOLUTE: Find annotations whose Id's match FindId.
- OIX_ANNOTATION_HYPERTAG
- OIX_ANNOTATION_BOOKMARKTAG

If FindId is used as a mask to find annotations, then any annotation for which Id & FindId = FindId will be removed. This parameter is required. If it is invalid then an error will be generated and the operation will abort.

- lFindId: This parameter specifies which annotations to find in conjunction with the mask value. FindId must be a positive number if the value for the Mask parameter is OIX_ANNOTATION_MASK or OIX_ANNOTATION_ABSOLUTE. This parameter is required. If the value is invalid (a negative number with Mask of OIX_ANNOTATION_MASK or OIX_ANNOTATION_ABSOLUTE), an error will be generated and the operation will abort.
- SearchPos: Indicates the starting position for the search as a position object. This parameter is not required. If no value is passed, then the starting position will be the start of the document. If the value is invalid, then an error will be generated and the operation will abort.

  For finding previous or next annotations, an updated SearchPos must be provided as a starting position.

Related to

SCCVW_FINDANNOTATION (Outside In Viewer)

FindPosition

Use this method to set a Position relative to another position. Many of the methods of finding the position are only supported for word processing file types.

Usage

```
Void FindPosition(OixPos StartPos, OixPos DestPos, short iFindMethod)
```

Returns

Boolean indicating success.

Parameters

- StartPos: Use this Position object to determine where to start from.
- DestPos: The Position object that will be set by this method.
- iFindMethod: How to find the position used to set the DestPos. The follow values are valid:
  - SCCVW_FINDFIRSTPOS: Find the first position in the current section (page).
  - SCCVW_FINDLASTPOS: Find the last position in the current section (page).
  - SCCVW_FINDPREVPOS: Find the position in the document that appears before the position identified by StartPos.

- SCCVW_FINDNEXTPOS: Find the position in the document that appears after the position identified by StartPos.
- SCCVW_FINDANCHORSELECTPOS: Find the position in the document that identifies the start of the current selection.
- SCCVW_FINDENDSELECTPOS: Find the position in the document that identifies the end of the current selection.
- SCCVW_FINDSTARTDISPLAYPOS: Find the position in the document that identifies the top/left most position in the currently displayed area.
- SCCVW_FINDENDDISPLAYPOS: Find the position it the document that identifies the bottom/right most position in the currently displayed area.
- SCCVW_FINDSTARTLINEPOS: Find the position in the document that identifies the start of the line that contains StartPos.
- SCCVW_FINDENDLINEPOS: Find the position in the document that identifies the end of the line that contains StartPos.
- SCCVW_FINDPREVLINEPOS: Find the position in the document that identifies the start of the line previous to the line that contains StartPos.
- SCCVW_FINDNEXTLINEPOS: Find the position in the document that identifies the start of the line following the line that contains StartPos.
- SCCVW_FINDPREVWORDPOS: Find the position in the document that identifies the start of the word previous to StartPos.
- SCCVW_FINDNEXTWORDPOS: Find the position in the document that identifies the start of the word that follows StartPos.
- SCCVW_FINDPREVSECTION: Find the section previous to the current position.
- SCCVW_FINDNEXTSECTION: Find the section that follows the current section.
- SCCVW_FINDPREVSELECTPOS: Find the position in the document that identifies the previous selected node when there are multiple selections (Archive files only).
- SCCVW_FINDNEXTSELECTPOS:Find the the position in the document that identifies the next selected node when there are multiple selections (Archive files only).

Related to

SCCVW_FINDPOSITION (Outside In Viewer)

GetActualCount

This method will return the actual character count when passed a position object.

Usage

```
long GetActualCount(OixPos Position)
```

Returns

Long

Parameters

- Position: This is a position object. It needs to be an allocated object, not an object reference.

Related to

SCCVW_MAPPOSITION (Outside In Viewer)

GetAnnotationData

This method provides the developer a way to retrieve extended data associated with annotations that are internally created by the viewer technology. The technology currently creates annotations to support hyperlinks and bookmarks defined in the original file. The data for the annotation is returned in the various Annotation properties. If the annotation is not found, the method returns FALSE. If the annotation is found the method returns TRUE.

Usage

```
BOOL GetAnnotationData(long lId, short iAnnotationType)
```

Returns

Boolean indicating found

Parameters

- lId: This is the Id of the annotation. If the annotation type is User, then this is the Id of that was used to add the annotation. For URL and Bookmark types, this number is the zero-based index of the URLs/Bookmarks in the file. This parameter is required. Any value will be accepted.
- iAnnotationType: This is the type of annotation to get. The following values are valid:
  - ADTYPE_NODATA: User
  - ADTYPE_URL: URL
  - ADTYPE_BOOKMARK: Bookmark

Related to

SCCVW_GETANNOTATIONDATA (Outside In Viewer)

GetCustomEMailHeader

This method will allow the developer to query the custom email headers that are visible or hidden. This information is only used when WPEmailHeaderOutput is set to SCCUT_WP_EMAILHEADERCUSTOM.

Usage

BOOL GetCustomEMailHeader (BOOL bVisible, LONG lHeaderID, long *lSubType, BSTR *strMimeHeaderName, BSTR *strMimeHeaderLabel)

Returns

TRUE: Indicates success

FALSE: Could not retrieve the information

Parameters

- bVisible: Flag indicating whether information about a header that is visible (rendered) or invisible is being requested.
- lHeaderID: Either the ID of a predefined email header field (SCCCA_MAIL_*) or an identifer between NONSTANDARD_HEADER_ID_BASE and NONSTANDARD_HEADER_ID_TOP for tracking a user-defined header.
- lSubType: The type(s) of documents in which to either show or hide this header. These can be joined with a bitwise OR operator. Available subtypes are:
  - SCCUT_MAILTYPE_EMAIL
  - SCCUT_MAILTYPE_JOURNAL
  - SCCUT_MAILTYPE_CONTACT
  - SCCUT_MAILTYPE_NOTE
  - SCCUT_MAILTYPE_APPOINTMENT
  - SCCUT_MAILTYPE_TASK
  - SCCUT_MAILTYPE_POST
  - SCCUT_MAILTYPE_DISTROLIST
- strMimeHeaderName : A Unicode string containing the value of a user-specified MIME header name.
- strMimeHeaderLabel: Unicode string that will be used as the label for a user-defined MIME header. This value is only used for user-defined headers.

Note:

Support for user-defined MIME headers is intended to allow Outside In to selectively display MIME headers that are not included in the predefined set of email headers known to Outside In. It is likely that most developers using Outside In will not need to specify user-defined MIME headers. Knowledge of the particular MIME headers present in the input email files is necessary in order to take advantage of this capability.

Related to

SCCID_MAILHEADERVISIBILE and SCCID_MAILHEADERHIDDEN (Outside In Viewer)

GetDrawPageInfo

This method will get page information about the requested page. It is to be used within calls to the InitDrawPage and DeinitDrawPage methods to get the best page size information before a call is made to DrawPageEx. This returns the calculated FormatWidth and FormatHeight in the PageFormatWidth and PageFormatHeight properties. If the information is not retrieved the method returns FAL

Usage

```
BOOL GetDrawPageInfo (LONG lPageNumber, SHORT iUnitsPerInch, OLE_HANDLE
hOutputDC, OLE_HANDLE hFormatDC)
```

Returns

- TRUE: Indicates Success
- FALSE: The Draw Page Info could not be calculated

Parameters

- lPageNumber: This is the zero-based number of the page to get information about. This parameter is required. An invalid value will cause an error and the operation will abort.
- iUnitsPerInch: This is the number of units per inch that PageFormatWidth and PageFormatHeight will be returned in. This parameter is required. An invalid (less than 1) value will cause an error and the operation will abort.
- hOutputDC: This is a handle to the device context to draw into. This parameter is required and an invalid value will cause an error.
- hFormatDC: This is a handle to a device (or information) context used to wrap text and retrieve other formatting information. This may not be the same as hOutputDC. This parameter is required and an invalid value will cause an error.

Related to

SCCVW_GETDRAWPAGEINFO (Outside In Viewer)

GetPageCount

This method counts the number of pages in the currently viewed file, as paginated to fit a rectangle and device of the OEM's choice. As this requires that the view window wrap every page in the document, this message can take some time to complete when processing extremely large documents.

Usage

```
BOOL GetPageCount (LONG lTop, LONG lLeft, LONG lBottom, LONG lRight,
                                    LONG lFormatWidth, LONG lFormatHeight,
                                    SHORT iUnitsPerInch, OLE_HANDLE hFormatDC,
                                    LONG *lNumPages)
```

Returns

- TRUE: Indicates Success

- FALSE: Error Occurred

Parameters

- lTop, lLeft, lBottom, lRight: Rectangle in device units that is equivalent to lFormatHeight and lFormatWidth. This rectangle must be in device units.
- lFormatHeight: Height of the wrappable part of the page. For instance, if your page in 11 inches tall and you want .5 inch margins, lFormatHeight should represent 10 inches.
- lFormatWidth: Width of the wrapable part of the page. For instance, if your page is 8.5 inches wide and you want .5 inch margins, lFormatWidth should represent 7.5 inches.
- iUnitsPerInch: This is the number of units per inch that PageFormatWidth and PageFormatHeight will be returned in. This parameter is required. An invalid (less than 1) value will cause an error and the operation will abort.
- hFormatDC: This is a handle to a device (or information) context used to wrap text and retrieve other formatting information.
- lNumPages : The number of pages (returned) .

GetProperty

This method retrieves any property information generated by the import filter. If a property is available, this function returns FALSE, otherwise it returns TRUE.

Usage

```
BOOL GetProperty(long lPropMode, BSTR *pPropName, BSTR *pPropData, long lPropId)
```

Returns

Boolean indicating success

Parameters

- lPropMode: One of the following:
  - SCCVW_GETPROP_FIRST: Retrieves the information for the first property generated by the filter.
  - SCCVW_GETPROP_NEXT: Retrieves the information for the next property generated by the filter.
  - SCCVW_GETPROP_PREV: Retrieves the information for the next property generated by the filter.
  - SCCVW_GETPROP_BYID: Retrieves the information (if any) for a property specified in the structure. Values are specified in the header files as SCCCA_*. User defined properties can not be specified.
  - SCCVW_GETPROP_BYINDEX: Retrieves a property specified by a zero-based index from the property array.

- pPropName : Pointer to string containing Property Name. This pointer needs to be freed if applicable.
- pPropData : Pointer to string containing Property Value. This pointer needs to be freed if applicable.
- lPropId : The property index or ID - required if retrieving by ID or index.

Related to

SCCID_GETPROPERTY(Outside In Viewer)

GetRawText

This method allows the developer to get the raw text buffer that contains the character identified by its actual character count from the beginning of the document. The RawText properties will be updated with the requested text block. Returns TRUE if the raw text block was found, otherwise returns FALSE.

Usage

```
BOOL GetRawText(long lTextOffset)
```

Returns

Boolean indicating found

Parameters

- lTextOffset: Set this to the actual character count, 0 based from the beginning of the document, that defines the raw text buffer being requested. This parameter is required. Invalid values will always cause the method to return FALSE.

Related to

SCCVW_GETRAWTEXT (Outside In Viewer)

GoToAnnotation

This method allows the developer to move the view/selection from annotation to annotation. If no matching annotations are found, then the method returns FALSE, otherwise the method returns TRUE.

Usage

```
BOOL GoToAnnotation( short iFindType, short iMask, long lFindId )
```

Return

Boolean indicating success.

Parameters

- iFindType: Indicates the type of GoToAnnotation operation to be performed. The following values are valid:
  - OIX_GOTOFIRST: Goto first or only matching annotation.

- OIX_GOTOLAST: Goto last or only matching annotation.
- OIX_GOTOPREV: Goto previous matching annotation.
- OIX_GOTONEXT: Goto next matching annotation.

This parameter is required. If the value is invalid, then the method will generate an Error event (see Error) and the operation will abort.

- iMask: Indicates how the annotations will be matched. The following values are valid:
  - OIX_ANNOTATION_ALL: Find all annotations - this includes internal annotations.
  - OIX_MASK: Use FindId as a mask to find annotations.
  - OIX_ABSOLUTE: Find annotations whose Id's match FindId.
  - OIX_ANNOTATION_HYPERTAG
  - OIX_ANNOTATION_BOOKMARKTAG

  If FindId is used as a mask to find annotations, then any annotation for which Id & FindId = FindId will be removed. This parameter is required. If it is invalid then an error will be fired and operation will abort.
- lFindId: This parameter specifies which annotations to goto in conjunction with the mask value.

Related to

SCCVW_GOTOANNOTATION (Outside In Viewer)

HiliteStyle

This method allows the developer a unique StyleId that indicates a custom hiliting style. The method returns TRUE if the style cannot be set. FALSE is returned if the method is successful.

Usage

```
BOOL HiliteStyle( long lStyleid, long lOptions, OLE_COLOR dwForeColor,
OLE_COLOR dwBackColor, short iCharAttr )
```

Returns

Boolean indicating success

Parameters

- lStyleId: A unique ID for this style that can be used when adding highlight annotations with the AddAnnotationHilite method. This parameter is required. A negative value will result in an error.
- lOptions: This parameter defines the options that are being set with this method. The following values may be OR-ed together. This parameter is required. There are no invalid values.
  - SCCVW_USEFOREGROUND: Use the ForeColor parameter to set the style.

- SCCVW_USEBACKGROUND: Use the BackColor parameter to set the style.
- SCCVW_USECHARATTR: Use the CharAttr parameter to set the style.

- dwForeColor: The color to use for the foreground of the hilited area. The parameter is required and there are no invalid values.
- dwBackColor: The color to use for the background of the hilited area. The parameter is required and there are no invalid values.
- iCharAttr: Defines the font attribute to be used for the highlight text if CharAttr was specified in the Options parameter. The valid values for CharAttr are listed in the header files.

Related to

SCCVW_HILITESTYLE (Outside In Viewer)

HScroll

This method is used by the developer to cause horizontal scrolling. dwScrollOp directs the control to scroll one line left or right, one page left or right, or to a specific position. If this parameter specifies that the view should scroll to a specific position, lPosition specifies that position, which must be within the scroll range.

Usage

```
LONG HScroll( ULONG dwScrollOp, LONG lPosition )
```

Returns

- 0: Indicates success (ERROR_SUCCESS)
- >0: Indicates failure, lPosition out of HScroll range (ERROR_INVALID_SCROLLBAR_RANGE)

Parameters

- dwScrollOp: May be one of the following values:
  - SCCSB_LINELEFT: Scroll one line left
  - SCCSB_LINERIGHT: Scroll one line right
  - SCCSB_PAGELEFT: Scroll one page left
  - SCCSB_PAGERIGHT: Scroll one page right
  - SCCSB_POSITION: Scroll to an absolute position
- lPosition: If dwScrollOp = SB_THUMBPOSITION, this parameter contains the absolute position to scroll to.

Related to

SCCVW_HSCROLL (Outside In Viewer)

IdleBitmap

This method may be called to set the bitmap for display when the view window is idle (no document loaded).

Usage

```
Void IdleBitmap(HINSTANCE hModule, UINT ResID);
```

Returns

Void

Parameters

- HModule: The instance handle of the .dll or .exe module containing the bitmap resource.
- ResID: The resource ID of the bitmap resource.

Related to

SCCVW_SETIDLEBITMAP (Outside In Viewer)

ImgShowFullScreen

If the show parameter is TRUE, then the image will be displayed using the entire screen until either the method is called again with a parameter of FALSE, or the user presses a key.

Usage

```
Void ImageShowFullScreen (BOOL bShow)
```

Returns

Void

Parameters

- bShow: If this is set to TRUE, then the image is shown full screen. If this is set to FALSE, then the image is shown normally. The parameter is required.

Related to

SCCID_VECSHOWFULLSCREEN, SCCID_BMPSHOWFULLSCREEN (Outside In Viewer)

ImgZoom

This method is used to zoom the image.

Usage

```
Void ImgZoom (short iZoomType)
```

Returns

Void

Parameters

- iZoomType: This parameter indicates what kind of zoom operation to perform on the image. The valid options are:
  - SCCVW_ZOOM_NOP: Zoom to percent set in ImgX(&Y)ZoomPercent properties
  - SCCVW_ZOOM_IN: Zooms In
  - SCCVW_ZOOM_OUT: Zooms Out
  - SCCVW_ZOOM_SELECTION: Zooms to the current selection
  - SCCVW_ZOOM_RESET: Restores the display to its original state (as determined by BMPFitMode and VecFitMode)

  The parameter is required. An invalid value will result in an error.

Related to

SCCID_VECZOOMEVENT, SCCID_BMPZOOMEVENT (Outside In Viewer)

InitDrawPage

This function prepares the control to execute the DrawPage or GetDrawPageInfo methods. If these methods are called without calling InitDrawPage, they will fail. If the method is able to allocate the necessary memory to call DrawPage or GetDrawPageInfo, then it returns TRUE. Otherwise, it will return FALSE and an error will be fired.

Usage

```
BOOL InitDrawPage()
```

Returns

- TRUE: Indicates Success
- FALSE: Indicates Error (fires Error Event)

Related to

SCCVW_INITDRAWPAGE (Outside In Viewer)

PrintOI

This method causes the viewer to print the file currently being viewed. The developer may want direct control of some of the printing process. PrintOI uses the values of some of the printer properties to control the printing process. Originally, these properties have values corresponding to the default behavior of the printing process. These properties include:

- PrintCollate

- PrintCopies
- PrinterDC
- PrinterDriver
- PrinterName
- PrinterPort
- PrintFont
- PrintMarginLeft, Right, Top, Bottom
- WhatToPrint

Usage

```
Void PrintOI ( BOOL bDoSetupDialog, ULONG dwFlags, BOOL bDoAbortDialog, BOOL
bStartDocAlreadyDone)
```

Returns

Void

Parameters

- bDoSetupDialog: If TRUE, the Windows Print Setup common dialog appears before printing.
- dwFlags: These flags determine if and how the print properties will be used. This parameter may have any of the following values OR-ed together:
  - USECOLLATE 0x00000200: Use the value of the PrintCollate property.
  - USECOPIES 0x00000400: Use the value of the PrintCopies property.
  - USEPRINTERDC 0x00000001: Use the device handle stored in PrinterDC (see PrinterDC) as the output device instead of the default printer.
  - USEPRINTERNAME 0x00000002: If USEPRINTERDC is not set, use PrinterName, PrinterPort, and PrinterDriver as the output device (see PrinterName, PrinterPort, and PrinterDriver) instead of the default printer. If USEPRINTERDC is set, PrinterName, PrinterPort, and PrinterDriver are used for display in the standard Abort dialog.
  - USEABORTPROC 0x00000080: Fire the PrinterAbort event (see PrinterAbort) instead of using the default Windows abort procedure.
  - USEPRINTHEADER: Print a header containing the page number and the text specified in the PrintJobName property.
- bDoAbortDialog: If TRUE, an abort dialog will be displayed with page numbers while the file is being printed.
- bStartDocAlreadyDone: This needs to be set only if the USEPRINTERDC flag is set. This parameter tells the viewer that a StartDoc has already been called for the printer device context (DC). The viewer will not do a StartDoc or EndDoc itself, assuming that the developer will call these functions.

Related to

SCCVW_PRINTEX (Outside In Viewer)

PrintOptions

This method displays a dialog that lets the user set a number of aspects for printing files.

Usage

```
short PrintOptions()
```

Returns

OIX_DIALOG_OK: Dialog successful

OIX_DIALOG_CANCEL: Dialog canceled by user

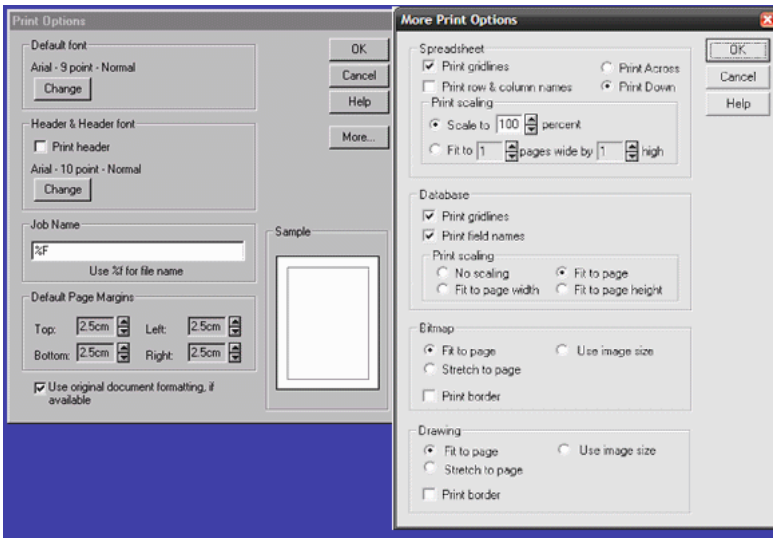
Parameters

none

Related to

SCCVW_DODIALOG (Outside In Viewer)

Figure 1 Print Options Dialogs



PrintSetup

This method brings up the Print Setup Common Dialog from which the user may select printers and change their settings.

Usage

```
Void PrintSetup()
```

Returns

Void

Parameters

none

Related to

SCCVW_PRINTSETUP (Outside In Viewer)

Search

Use this method to search for text inside the currently viewed document. The search can be performed either with or without a dialog box. The function will return 0 if the search string was found, -1 if an error occurred, or 1 if the search reached either end of the file before finding the search string.

Usage

```
Short Search(BOOL bDialog, String szSearchText, short iSearchType, short
iSearchFrom, short iSearchDirection)
```

Returns

- 0: If the search string was found
- -1: If an error occurred
- 1: If the search reached either end of the file before finding the search string

Parameters

- bDialog: If Dialog is set to TRUE, then a search dialog will be displayed for user input. The search dialog is initialized from the other parameters. The SearchFrom parameter will be ignored and the search will take place from the current position if the dialog is used. If Dialog is set to FALSE, then the search will start without displaying a dialog. This parameter is required and does not have any invalid values.
- szSearchText: The text to search for. This parameter is required and will produce an error event if it is invalid.
- iSearchType: This parameter indicates whether the search should be case sensitive or not. The valid values are:
  - SCCVW_SEARCHCASE: Case-sensitive search
  - SCCVW_SEARCHNOCASE: Case-insensitive search

  The parameter is required and will produce an error event if it is invalid.
- iSearchFrom: This parameter tells the control where to start the search. The parameter will be ignored if the search dialog is used. When the dialog is used, the search always starts from the current caret position. The valid values are:
  - SCCVW_SEARCHTOP: Search From Top
  - SCCVW_SEARCHBOTTOM: Search From Bottom
  - SCCVW_SEARCHCURRENT: Search From Current Caret Position

  The parameter is required and will produce an error event if it is invalid.

- ISearchDirection: This parameter indicates in which direction to perform the search. The valid values are:
  - SCCVW_SEARCHFORWARD: Search forward
  - SCCVW_SEARCHBACK: Search backward

  The parameter is required and will produce an error if it is invalid.

Related to

SCCVW_SEARCH, SCCVW_SEARCHDIALOG (Outside In Viewer)

SearchNext

Use this method to scan the file for the same text string specified in a previous call to the Search method. The function will return 0 if the search string was found, -1 if an error occurred, or 1 if the search reached either end of the file before finding the search string.

Usage

```
Short SearchNext(short iSearchDirection)
```

Returns

- 0: If the search string was found
- -1: If an error occurred
- 1: If the search reached either end of the file before finding the search string

Parameters

- ISearchDirection: This parameter indicates in which direction to perform the search. The valid values are:
  - SCCVW_SEARCHSAME: Search in same direction as original search
  - SCCVW_SEARCHFORWARD: Search forward
  - SCCVW_SEARCHBACK: Search backward

  The parameter is required and will produce an error if it is invalid.

Related to

SCCVW_SEARCHNEXT (Outside In Viewer)

SelectAll

This method causes the view window to select all contained data.

Usage

```
Void SelectAll()
```

Returns

Void

Parameters

none

Related to

SCCVW_SELECTALL (Outside In Viewer)

SetActualCount

This method will set the position object to a location count characters from the beginning of the document.

Usage

```
Void SetActualCount(OixPos Position, long Count)
```

Returns

Void

Parameters

- Position: This is a position object. It needs to be an allocated object, not an object reference.
- Count: This is the actual character count to set the Position object to.

Related to

SCCVW_MAPPOSITION (Outside In Viewer)

SetCustomEmailHeader

This method will allow the developer fine-grained control over what email headers are rendered. This information is only used when WPEmailHeaderOutput is set to SCCUT_WP_EMAILHEADERCUSTOM.

Usage

BOOL SetCustomEMailHeader (BOOL bVisible, LONG lHeaderID, long lSubType, BSTR strMimeHeaderName, BSTR strMimeHeaderLabel)

Returns

TRUE: Indicates success

FALSE: Could not set custom header.

Parameters

- bVisible: Flag indicating whether the header that is visible (rendered) or hidden.

- lHeaderID: Either the ID of a predefined email header field (SCCCA_MAIL_*) or an identifer between NONSTANDARD_HEADER_ID_BASE and NONSTANDARD_HEADER_ID_TOP for tracking a user-defined header.
- lSubType: The type(s) of documents in which to either show or hide this header. These can be joined with a bitwise OR operator. Available subtypes are:
  - SCCUT_MAILTYPE_EMAIL
  - SCCUT_MAILTYPE_JOURNAL
  - SCCUT_MAILTYPE_CONTACT
  - SCCUT_MAILTYPE_NOTE
  - SCCUT_MAILTYPE_APPOINTMENT
  - SCCUT_MAILTYPE_TASK
  - SCCUT_MAILTYPE_POST
  - SCCUT_MAILTYPE_DISTROLIST
- strMimeHeaderName : A Unicode string containing the value of a user-specified MIME header name.
- strMimeHeaderLabel: Unicode string that will be used as the label for a user-defined MIME header. This value is only used for user-defined headers.

Note:

Support for user-defined MIME headers is intended to allow Outside In to selectively display MIME headers that are not included in the predefined set of email headers known to Outside In. It is likely that most developers using Outside In will not need to specify user-defined MIME headers. Knowledge of the particular MIME headers present in the input email files is necessary in order to take advantage of this capability.

Related to

SCCID_MAILHEADERVISIBILE and SCCID_MAILHEADERHIDDEN (Outside In Viewer)

SetPositionToCurrent

This method allows the developer to set a position object to the location of the caret in the currently viewed document.

Usage

```
Void SetPositionToCurrent(OixPos Position)
```

Returns

Void

Parameters

- Position: This is the Position object to set to the current location of the caret.

SetPositionToSelection

This method allows the developer to set two position objects based on the currently selected text.

Usage

```
Void SetPositionToSelection(OixPos StartPos, OixPos EndPos)
```

Returns

Void

Parameters

- StartPos: This is the Position object that will be set to the location of the beginning of the current selection. If there is no current selection, it will be set to the beginning of the document.
- EndPos: This is the Position object that will be set to the location of the end of the current selection. If there is no current selection, it will be set to the end of the document.

Related to

SCCVW_DISPLAYPOSITION (Outside In Viewer)

SystemIdle

This method should be called repeatedly by the developer to the control during the viewing process. This message allows the control to blink the cursor and read ahead in the file being viewed. This message is only needed if the SystemTimer property (see SystemTimer) has been set to FALSE.

Usage

```
SystemIdle()
```

Related to

SCCVW_IDLE (Outside In Viewer)

ViewFile

Call this method to open and display the contents of a file. If another file is already open, it will be closed automatically.

Usage

```
BOOL ViewFile(BOOL bDialog, VARIANT FileSpec, SHORT iSpecType, SHORT iViewAs,
BOOL bUseDisplayName, BSTR szDisplayName, BOOL bDeleteOnClose, Ioixredirect
*pIoRedir)
```

Returns

- TRUE: Indicates success
- FALSE: The file could not be viewed

Parameters

- bDialog: If this parameter is TRUE, then a View File dialog will be displayed by the control. The file selected from this dialog will then be viewed. If FALSE, then no dialog will be displayed and the file specified by the szFileSpec parameter will be viewed. This parameter is required. There are no invalid values. Note that no dialog will appear if the iSpecType parameter is IOTYPE_REDIRECT.
- FileSpec: This is the fully qualified path of the file to be viewed. If the Dialog parameter has been set to TRUE, then the specification passed in this parameter will be used to initialize the View File dialog. This parameter is required. If the bDialog parameter is set to FALSE and the path is invalid, then an error will be produced. If the bDialog parameter is true, then there are no invalid values from the szFileSpec. If the FileSpec type is "IO Redirect," then the FileSpec will represent some sort of user data that identifies the data to import.
- iSpecType: This indicates what kind of information is in the FileSpec. The valid values are:
  - IOTYPE_OEMPATH
  - IOTYPE_UNICODEPATH
  - IOTYPE_REDIRECT
- iViewAs: Indicates which format to force the file to be viewed in. The valid formats are:
  - FI_ANSI: Display as text and assume the ANSI character set.
  - FI_ANSI8: Display as text and assume the ANSI character set. Ignore characters over 0x7F.
  - FI_ASCII: Display as text and assume the PC character set.
  - FI_ASCII8: Display as text and assume the PC character set. Ignore characters over 0x7F.
  - FI_MAC: Display as text and assume the MAC character set.
  - FI_MAC8: Display as text and assume the MAC character set. Ignore characters over 0x7F.
  - FI_UNICODE: Display as text and assume the Unicode character set.
  - FI_HEX: Display as hexadecimal.
  - FI_SHIFTJIS: Display as text and assume the Japanese ShiftJis character set.
  - FI_CHINESEGB: Display as text and assume the Chinese GB character set.
  - FI_HANGEUL: Display as text and assume the Korean Hangeul character set.
  - FI_CHINESEBIG5: Display as text and assume the Chinese big 5 character set.

Invalid values default to 0.

- bUseDisplayName: If TRUE, then use the display name that follows. Otherwise, the control will generate a display name based on the file name. There are no invalid values. If this parameter is set to true and DisplayName is not passed, then an error will occur.
- szDisplayName: The name to be used for the file, should it be displayed. If the bUseDisplayName parameter is true, then this value will be used as the name of the file. If the bUseDisplayName parameter is FALSE, then this parameter will be ignored.
- bDeleteOnClose: Set this parameter to true if the file being viewed should be deleted after the control is finished viewing it.
- pIoRedir: If iSpecType is "IO Redirect," this parameter represents user data that will be passed to all of the Redirected IO events.

Related to

SCCVW_VIEWFILE (Outside In Viewer)

VScroll

This method is used by the developer to cause vertical scrolling. It has two parameters. dwScrollOp directs the control to scroll one line up or down, one page up or down, or to a specific position. If this parameter specifies that the view should scroll to a specific position, lPosition specifies that position, which must be within the scroll range.

Usage

```
LONG VScroll( ULONG dwScrollOp, LONG lPosition )
```

Returns

- 0: Indicates success (ERROR_SUCCESS)
- >0: Indicates failure, lPosition out of VScroll range (ERROR_INVALID_SCROLLBAR_RANGE)

Parameters

- dwScrollOp: May be one of the following values:
  - SCCSB_LINEUP: Scroll one line up
  - SCCSB_BOTTOM: Scroll to bottom of page
  - SCCSB_LINEDOWN: Scroll one line down
  - SCCSB_PAGEUP: Scroll one view window page up
  - SCCSB_PAGEDOWN: Scroll one view window page down
  - SCCSB_NEXTDOCPAGE: Scroll to the beginning of the next document page or section

- SCCSB_PREVDOCPAGE: Scroll to the beginning of the previous document page or section
- SCCSB_TOP: Scroll to top of page
- SCCSB_POSITION: Scroll to an absolute position
- lPosition: If dwScrollOp = SB_THUMBPOSITION, this parameter contains the absolute position to scroll to.

Related to

SCCVW_VSCROLL (Outside In Viewer)

Events

The following events occur during actions.

AnnotationEvent

This event informs the developer that some user action has been taken on an annotation.

Usage

```
Void AnnotationEvent(long lEvent, long lId, VARIANT varData)
```

Returns

Void

Parameters

- lEvent: Passes the action that triggered the event. See AddAnnotationHilite, or AddAnnotationPicture for valid values.
- lId: Unique positive number that indicates which annotation fired the event.
- varData: Data value associated with the annotation.

Related to

SCCVW_ANNOTATIONEVENT (Outside In Viewer)

Close

The developer needs to respond to this event by closing the file identified by the varFile parameter and cleaning up all memory associated with it. The pResult parameter should be set to IOERR_OK if the operation is successful, and IOERR_UNKNOWN if the operation is not.

Usage

```
Close( VARIANT FileSpec, VARIANT varData, VARIANT varFile, long *pResult)
```

Parameters

- FileSpec: Contains the file specification passed to the ViewFile method in the FileSpec parameter.
- varData: Contains the variable passed to the ViewFile method in the Redirect parameters.
- varFile: This contains the variant set by the developer in the Open event.
- pResult: This developer sets this variable to return the result of the operation.

ContextMenu

This event signals the container application in the event that a right-mouse click has occurred inside the view. The developer may decide to handle this interaction, or let Outside In pop up its own context menu. See the DoContextMenu property (see DoContextMenu) for information on disabling the built-in context menu.

Usage

```
ContextMenu ( int iXPos, int iYPos )
```

Parameters

- int iXPos: x coordinate of right mouse click
- int iYPos: y coordinate of right mouse click

Related to

SCCVW_CONTEXTMENU (Outside In Viewer)

DisplayEngineChange

This event is generated whenever the display engine changes. The new information for the display engine will be updated in the DisplayEngine properties at the same time.

Usage

```
DisplayEngineChange()
```

Related to

SCCVW_DISPLAYCHANGE (Outside In Viewer)

DoHelp

This event is generated whenever a help button has been pressed in one of the control's dialogs.

Usage

```
DoHelp(SHORT iDialog)
```

Parameters

- iDialog: This parameter is a number that identifies the dialog box in which the help button has been pressed. The possible values are as follows:
  - SCCID_DISPLAYOPTIONSDIALOG: Display Options Dialog
  - SCCID_DISPLAYMOREDIALOG: Display More Dialog
  - SCCID_PRINTOPTIONSDIALOG: Print Options Dialog
  - SCCID_PRINTMOREDIALOG: Print More Dialog
  - SCCID_CLIPBOARDOPTIONSDIALOG: Clipboard Options Dialog
  - SCCID_CLIPBOARDMOREDIALOG: Clipboard More Dialog
  - SCCID_PRINTDIALOG: Print Dialog
  - SCCID_PRINTSETUPDIALOG: Print Setup Dialog
  - SCCID_SELECTFONTDIALOG: Select Font Dialog

Related to

SCCVW_DOHELP (Outside In Viewer)

EnableApp

This event is generated whenever the developer application should be disabled/enabled. This message is sent before and after dialogs and printing. It allows the developer to disable any part of the application that could affect the view window.

Usage

```
EnableApp(BOOL bEnable, short *iReturn)
```

Parameters

- bEnable: If TRUE, then the developer application should be enabled. If FALSE, then the developer application should be disabled.
- iReturn: If the developer handles the enable/disable, return 24. If the default enable/disable behavior is desired, return 0.

Related to

SCCVW_ENABLEAPP (Outside In Viewer)

Error

This event is generated when an error condition has occurred either in the control or the Outside In Viewer. The ErrorCode and ErrorMsg properties (see ErrorCode, and ErrorMsg) will be filled with the appropriate values. This event is generated regardless of the state of ErrorShowMsg (see ErrorShowMsg).

Usage

```
Error()
```

Related to

SCCVW_BAILOUT (Outside In Viewer)

FileChange

This event is generated when the file being viewed is changed, such as with a ViewFile or Clear method (see ViewFile, and Clear). The FileInfo properties will be updated at the same time.

Usage

```
FileChange (BOOL bOpen, BOOL bOpenFollowing)
```

Parameters

- bOpen: If TRUE, indicates that the file is being opened. Otherwise, the file is being closed.
- bOpenFollowing: If the file is being closed because another file is being opened, then the OpenFollowing parameter will be TRUE. If this is the case, then another FileChange event should be coming immediately.

Related to

SCCVW_FILECHANGE (Outside In Viewer)

GenSecond

If a secondary file needs to be opened (for example, a .gif graphic in an HTML file), then the control will fire this event. The developer needs to assign the pFileSpec parameter the same file specification as for the FileSpec parameter of the ViewFile method (see ViewFile). The SpecType parameter also corresponds to the iSpecType parameter of the ViewFile method. If the developer wants to use IO redirect for the secondary file, then the SpecType must be IOTYPE_REDIRECT, just as with the ViewFile method. Finally, the pvarData parameter is the same as the Redirect parameter for the ViewFile method. Should everything be filled out OK, then the developer should return IOERR_OK in the pResult parameter. If the secondary file cannot be processed, then the developer should return IOERR_UNKNOWN.

Usage

```
GenSecond(VARIANT FileSpec, VARIANT varData, VARIANT varFile, BSTR FileName,
VARIANT *pFileSpec, long *SpecType, VARIANT *pvarData, long *pResult)
```

Parameters

- FileSpec: Contains the file specification passed to the ViewFile FileSpec parameter.
- varData: Contains the variable passed to the ViewFile method in the Redirect parameters.
- varFile: This contains the variant set by the developer in the Open event.
- FileName: The file name of the secondary file to be opened.
- pFileSpec: The file specification to be used to open the secondary file.

- SpecType: The type of file specification contained in pFileSpec.
- pvarData: The user data for the file as would be passed to the Redirect parameter of the ViewFile method.
- pResult: This developer sets this variable to return the result of the operation.

GetCredentials

The developer responds to this event by returning the requested information in the buffer provided. If the requested information is not available, the value SCCERR_CANCEL should be assigned to pResult. If redirected I/O is being used, this event requires that IOGETINFO_PATHNAME and IOGETINFO_PATHTYPE be handled. The information that is provided in response to these calls is what is passed back through the pSpec and dwSpecType in the FileAccess callback. Note that the pSpec that is being passed back is not the one that was given to the technology originally.

Usage

```
GetCredentials(BSTR Filename, long lRequestID, long lAttemptNumber, long
lMaxLength, VARIANT *pStrCredentials, long *pResult)
```

Parameters

- FileName: The filename of the file whose credentials are being requested.
- lRequestID: The ID of the information being requested. This can be one of the following values:
  - OIT_FILEACCESS_PASSWORD – Requesting the password of the file
  - OIT_FILEACCESS_NOTESID – Requesting the notes ID file location
- lAttemptNumber: The number of times the information has already been requested for the currently requested item.
- lMaxLength: The size of the buffer provided for the credentials information.
- pStrCredentials : Buffer to hold the information. For the 2 RequestIDs currently supported, this is a string buffer.
- pResult : The developer sets this variable to return the result of the operation.
  - SCCERR_OK: Tells Outside In that the requested information is provided.
  - SCCERR_CANCEL: Tells Outside In that the requested information is not available.

Note:

Not all formats that use passwords are supported. Only Microsoft Office binary (97-2003), Microsoft Office 2007, Microsoft Outlook PST 97-2013, Lotus NSF, PDF (with RC4 & AES 128-bit encryption), and Zip (with AES 128- & 256-bit, ZipCrypto) are currently supported.

GetInfo

Usage

```
GetInfo(VARIANT FileSpec, VARIANT varData, VARIANT varFile, long InfoId,
VARIANT *pInfo, long *pResult)
```

Parameters

- FileSpec: Contains the file specification passed to the ViewFile method in the FileSpec parameter.
- varData: Contains the variable passed to the ViewFile method in the Redirect parameters.
- varFile: This contains the variant set by the developer in the Open event.
- InfoId: A value that indicates what information needs to be returned. It indicates the action that the developer needs to take to respond to the event. The following values are valid:
  - IOGETINFO_FILENAME: The base file name (no path) of the open file has been requested. If the file name cannot be determined, then pResult should be assigned IOERR_UNKNOWN. However, if the file name can be determined; then the developer should assign pInfo the file name and pResult should be assigned IOERR_OK.
  - IOGETINFO_PATHNAME: The fully qualified file path (including the file name) of the open file has been requested. If the file path cannot be determined, then pResult should be assigned IOERR_UNKNOWN. However, if the file path can be determined; then the developer should assign pInfo the file name and pResult should be assigned IOERR_OK.
  - IOGETINFO_ISOLE2STORAGE: The value IOERR_FALSE should be assigned to pResult.

  For other values for InfoId, the value IOERR_BADINFOID should be assigned to pResult.
- pInfo: The developer needs to assign the requested information (identified by InfoId) to this parameter.
- pResult: The developer sets this variable to return the result of the operation.

GetTechPath

The developer can respond to this event if a technology path other than the one referenced in the registry's FilterPath entry is to be used. If the registry entry is to be used, SCCERR_CANCEL should be assigned to pResult.

Note:

GetTechPath does not work with VB6 applications.

Usage

```
GetTechPath(VARIANT *pStrTechPath, long * pResult)
```

Parameters

- pStrTechPath : Buffer to hold the new technology path. This is a string buffer.
- pResult: The developer sets this variable to return the result of the operation.
  - SCCERR_OK: Tells Outside In that a new technology path is to be used
  - SCCERR_CANCEL: Tells Outside In that the registry entry FilterPath is to be used.

HScrollPageSize

This event is generated when the page (thumb) size of the horizontal scroll bar changes. This allows the developer to establish correct thumb sizes of scroll bars external to the view window.

Usage

```
HScrollPageSize( int iPageSize )
```

Parameters

- int iPageSize: Value of the new thumb size

Related

SCCVW_SETHSCROLLPAGESIZE (Outside In Viewer)

HScrollPosition

This event is generated when the position of the horizontal scroll bar changes. This allows the developer to establish correct positioning of scroll bars external to the view window.

Usage

```
HScrollPosition( LONG lPosition )
```

Parameters

- LONG lPosition: New absolute position of the scroll bar.

Related to

SCCVW_SETHSCROLLPOSITION (Outside In Viewer)

HScrollRange

This event is generated when the range of the horizontal scroll bar changes. It allows the developer to establish correct scroll ranges external to the view window.

Usage

```
HScrollRange(int iMin int iMax )
```

Parameters

- int iMin: New minimum position for scroll bar
- int iMax: New maximum position for scroll bar

Related to

SCCVW_SETHSCROLLRANGE (Outside In Viewer)

HScrollState

This event is generated when the state of the horizontal scroll bar changes. It allows the developer to establish the correct state of scroll bars external to the view window.

Usage

```
HScrollState( BOOL bEnabled )
```

Parameters

- BOOL bEnabled: State of horizontal scroll bar

Related to

SCCVW_SETHSCROLLSTATE (Outside In Viewer)

InformationMessage

This event is generated during processing of the document. Each message indicates that the Viewer has identified a situation that it knows it is unable to correctly render. As re-rendering a page may cause the view window to reprocess the cause of a message, these messages should not be considered unique instances of a problem.

Usage

```
void InformationMessage(long lStatus)
```

Parameters

lStatus: Value indicating the type of problem that has been encountered. One of the following values:

- SCCVW_INFO_EQUATION: The file contained Equations.
- SCCVW_INFO_BADCOMPRESSION: A graphic used an unsupported compression method.
- SCCVW_INFO_BADCOLORSPACE: A graphic used an unsupported colorspace
- SCCVW_INFO_FORMS: The document used forms.
- SCCVW_INFO_MISSINGMAP: A PDF document did not have a toUnicode table.
- SCCVW_INFO_VERTICALTEXT: A vertical text run was present.

- SCCVW_INFO_TEXTEFFECTS: Unsupported text effects were present (such as Word Art).
- SCCVW_INFO_RTTOLFTTABLES: A Table had right-to-left columns.
- SCCVW_INFO_ALIASEDFONT: A font was missing, but a font alias was used.
- SCCVW_INFO_MISSINGFONT: A desired font was not available on the system.
- SCCVW_INFO_SUBDOCFAILED: A subdocument had an error.
- SCCVW_INFO_TYPETHREEFONT: A Type 3 Font was encountered.
- SCCVW_INFO_BADSHADING: An unsupported shading pattern was encountered.
- SCCVW_INFO_BADHTML: Non-standard HTML was encountered.

Keydown

This event is generated whenever a key is pressed while the view window has focus. This allows developers to define their own handlers for specific keys.

Usage

```
Keydown ( int iVKey )
```

Parameters

- int iVKey: Virtual key code for the key pressed

Related to

SCCVW_KEYDOWN (Outside In Viewer)

Open

This event is generated after the developer calls ViewFile (or responds to a GenSecond event). The developer needs to respond by opening the requested file and placing a meaningful value into the pvarFile variant. The developer needs to set the pResult parameter to IOERR_OK if the open operation is successful, or IOERR_UNKNOWN if the open operation fails.

Usage

```
ViewThisFile(VARIANT FileSpec, VARIANT varData, VARIANT *pvarFile, long
*pResult)
```

Parameters

- FileSpec: Contains the file specification passed to the ViewFile method in the FileSpec parameter.
- VarData: Contains the variable passed to the ViewFile method in the Redirect parameters.
- pvarFile: The developer needs to set this variable to whatever information is generated by opening the file (for example, a handle).

- pResult: This developer sets this variable to return the result of the operation.

OptionChange

This event is generated whenever an option has been changed. It can be used to monitor the current state of options.

Usage

```
OptionChange (short iOptionId)
```

Parameters

- iOptionId: The Id number of the option that changed. The possible values are included in the header files.

Related to

SCCVW_OPTIONCHANGE (Outside In Viewer)

PrinterAbort

When the USEABORTPROC flag is set in the PrintOI method (see PrintOI), this event is generated periodically during the printing process. This allows the developer to handle the abort procedure.

Usage

```
PrinterAbort(HDC hDC, int iError, bool *bContinue )
```

Parameters

- hDC: Identifies the device context for the print job.
- iError: Specifies whether an error has occurred. This parameter is zero if no error has occurred; it is SP_OUTOFDISK if Windows Print Manager is currently out of disk space and more disk space will become available if the application waits.
- bContinue: The developer must set this value to TRUE to continue the print job, or FALSE to abort the print job.

Related to

SCCVW_PRINTEX (Outside In Viewer)

RawTextEvent

This event is generated repeatedly during the initial read of a document. Each event passes the 0 based offset of the next block of raw text available from the control. The last RawTextEvent (see RawTextEvent) contains a text offset of -1 to indicate it is done extracting text from the file. Use the GetRawText method (see GetRawText) to retrieve the text buffer.

Usage

```
Void RawTextEvent(long lTextOffset)
```

Parameters

- lTextOffset: The 0-based offset of the next block of raw text available from the control. Pass this parameter to GetRawText (see GetRawText) to set the RawText properties to this new block.

Related to

SCCVW_RAWTEXTEVENT (Outside In Viewer)

Read

The developer responds to this event by reading in the requested amount of data from the current file position. After reading the requested data, the file pointer should be set to the byte after the last byte read. The data read should be returned to the control by assigning a SAFEARRAY of bytes to the pData variant. The actual count of bytes read should be assigned to the pCount parameter. If all or some of the data can be read, then IOERR_OK should be assigned to pResult. If the read failed because the file pointer was beyond the end of the file at the time of the read, then IOERR_EOF should be assigned to pResult. If the read failed for some reason other than being beyond the end of the file, then the value IOERR_UNKNOWN should be assigned to pResult.

Usage

```
Read(VARIANT FileSpec, VARIANT varData, VARIANT varFile, VARIANT *pData, long
Size, long *pCount, long *pResult)
```

Parameters

- FileSpec: Contains the file specification passed to the ViewFile method in the FileSpec parameter.
- varData: Contains the variable passed to the ViewFile method in the Redirect parameters.
- varFile: This contains the variant set by the developer in the Open event.
- pData: Variable that needs to be assigned a SAFEARRAY of bytes that contains the requested data.
- Size: The amount of data to be read.
- pCount: The actual count of bytes read.
- pResult: This developer sets this variable to return the result of the operation.

ReadAheadDone

Signals that the chunker is done reading the document. This means that the Outside In Viewer has completely read to the end of the document. This doesn't mean that rendering is done. It may also be necessary for the Outside In Viewer to access the file again while the user scrolls through the document.

Usage

```
ReadAheadDone()
```

Seek

The developer responds to this event by moving the file pointer. If the seek is successful, then the value IOERR_OK should be assigned to pResult. If the seek is not successful, then the value IOERR_UNKNOWN should be assigned to pResult.

Usage

```
Seek(VARIANT FileSpec, VARIANT varData, VARIANT varFile, short From, long
Offset, long *pResult)
```

Parameters

- FileSpec: Contains the file specification passed to the ViewFile method in the FileSpec parameter.
- varData: Contains the variable passed to the ViewFile method in the Redirect parameters.
- varFile: This contains the variant set by the developer in the Open event.
- From: This value indicates the starting point for moving the file pointer. The values are:
  - IOSEEK_TOP: Move the file position Offset bytes from the top of the file.
  - IOSEEK_BOTTOM: Move the file position Offset bytes from the bottom of the file.
  - IOSEEK_CURRENT: Move the file position Offset bytes from the current file position.
- Offset: The number of bytes to move the file pointer. A positive value moves the file pointer forward in the file and a negative value moves the file pointer backward.
- pResult: The developer sets this variable to return the result of the operation.

SelChange

This event is generated when the selection state of the contents of the view window has changed.

Usage

```
SelChange()
```

Parameters

None

Related to

SCCVW_SELCHANGE (Outside In Viewer)

Stat

This event is generated repeatedly during the read of a document. This event is periodically generated to verify that the file is still being processed. Use this with a monitoring process to help identify files that may be hung.

Usage

Void Stat(long *pResult)

Parameters

pResult - Set this variable to OIT_STATUS_WORKING to indicate continuation, or to OIT_STATUS_ABORT to interrupt viewing of the file.

Related to

VWSetStatCallback (Outside In Viewer)

Tell

The developer should respond to this event by assigning the current file position to the pOffset parameter. If successful, the developer should also assign the value IOERR_OK to the pResult parameter. If not successful, the developer should assign the value IOERR_UNKNOWN to pResult.

Usage

```
Tell(VARIANT FileSpec, VARIANT varData, VARIANT varFile, long *pOffset, long
*pResult)
```

Parameters

- FileSpec: Contains the file specification passed to the ViewFile method in the FileSpec parameter.
- varData: Contains the variable passed to the ViewFile method in the Redirect parameters.
- varFile: This contains the variant set by the developer in the Open event.
- pOffset: This parameter is assigned the current file position.
- pResult: This developer sets this variable to return the result of the operation.

ViewThisFile

This event is generated when another file should be viewed. Currently, this occurs when the user double-clicks or hits return on a file entry in an Archive view and on a hyperlink to a referenced document. The developer must return a value indicating whether the file is to be viewed or not.

Do not use the ViewFile method (see ViewFile) with the same view window when handling the ViewThisFile event (see ViewThisFile).

Usage

```
ViewThisFile(VARIANT FileSpec, SHORT iSpecType, SHORT *iReturn)
```

Parameters

- szFileSpec: Contains the file specification for the new file. This specification may be passed to the ViewFile method to view the file.
- iSpecType: The type of file spec information contained in the FileSpec variable. See the ViewFile method for more information.
- iReturn: The developer must set this value to indicate whether the file is viewed or not.

Related to

SCCVW_VIEWTHISFILE (Outside In Viewer)

VScrollPageSize

This event is generated when the page (thumb) size of the vertical scroll bar changes. This allows the developer to establish correct thumb sizes of scroll bars external to the view window.

Usage

```
VScrollPageSize( int iPageSize )
```

Parameters

- int iPageSize: New size of vertical page

Related to

SCCVW_SETVSCROLLPAGESIZE (Outside In Viewer)

VScrollPosition

This event is generated when the position of the vertical scroll bar changes. This allows the developer to establish correct positioning of scroll bars external to the view window.

Usage

```
VScrollPosition( LONG lPosition )
```

Parameters

- LONG lPosition: New absolute position of the scroll bar.

Related to

SCCVW_SETVSCROLLPOSITION (Outside In Viewer)

VScrollRange

Obsolete - replaced by VScrollRangeMin and VScrollRangeMax.

VScrollRangeMin

Usage

```
VScrollRangeMin(long lRangeMin)
```

Parameters

- long lRangeMin : New minimum position for scroll bar.

Related to

SCCVW_SETVSCROLLRANGEMIN (Outside In Viewer)

VScrollRangeMax

Usage

```
VScrollRangeMax(long lRangeMax)
```

Parameters

- long lRangeMax : New maximum position for scroll bar.

Related to

SCCVW_SETVSCROLLRANGEMAX (Outside In Viewer)

VScrollState

This event is generated when the range of the vertical scroll bar changes. It allows the developer to establish correct scroll ranges external to the view window.

Usage

```
VScrollState( BOOL bEnabled )
```

Parameters

- BOOL bEnabled: State of vertical scroll bar

Related to

SCCVW_SETVSCROLLSTATE (Outside In Viewer)

B

Copyrights and Licensing

This appendix provides a comprehensive overview of all copyright and licensing information for Outside In Viewer for ActiveX.

Outside In Viewer for ActiveX Licensing

The Programs (which include both the software and documentation) contain proprietary information; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent, and other intellectual and industrial property laws. Reverse engineering, disassembly, or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. This document is not warranted to be error-free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose.If the Programs are delivered to the United States Government or anyone licensing or using the Programs on behalf of the United States Government, the following notice is applicable:U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the Programs, including documentation and technical data, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement, and, to the extent applicable, the additional rights set forth in FAR 52.227-19, Commercial Computer Software--Restricted Rights (June 1987). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and we disclaim liability for any damages caused by such use of the Programs.Oracle, JD Edwards, PeopleSoft, and Siebel are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.The Programs may provide links to web sites and access to content, products, and services from third parties. Oracle is not responsible for the availability of, or any content provided on, third-party web sites. You bear all risks associated with the use of such content. If you choose to purchase any products or services from a third party, the relationship is directly between you and the third party. Oracle is not responsible for: (a) the quality of third-party products or services; or (b) fulfilling any of the terms of the agreement with the third party, including delivery of products or services and warranty obligations related to purchased products or services. Oracle is not responsible for any loss or damage of any sort that you may incur from dealing with any third party.

Portions relating to XServer copyright 1990, 1991 Network Computing Devices, 1987 Digital Equipment Corporation and the Massachusetts Institute of Technology.

Portions relating to PNG copyright 1999, 2000, 2001, 2002 Greg Roelofs. Portions relating to PNG Copyright 1995-1996 Jean-loup Gailly and Mark Adler Portions relating to PNG Copyright 1998, 1999 Glenn Randers-Pehrson, Tom Lane, Willem van Schaik, John Bowler, Kevin Bracey, Sam Bushell, Magnus Holmgren, Greg Roelofs, Tom Tanner, Andreas Dilger, Dave Martindale, Guy Eric Schalnat, Paul Schmidt, Tim Wegner

Portions relating to JPEG and to color quantization copyright 2000, 2001, 2002, Doug Becker and copyright (C) 1994, 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, Thomas G. Lane. This software is based in part on the work of the Independent JPEG Group. See the file README-JPEG.TXT for more information. Portions relating to WBMP copyright 2000, 2001, 2002 Maurice Szmurlo and Johan Van den Brande. Portions relating to GIF Copyright 1987, by Steven A. Bennett.

UnRAR - free utility for RAR archivesLicense for use and distribution of FREE portable versionThe source code of UnRAR utility is freeware. This means:1. All copyrights to RAR and the utility UnRAR are exclusively owned by the author - Alexander Roshal.2. The UnRAR sources may be used in any software to handle RAR archives without limitations free of charge, but cannot be used to re-create the RAR compression algorithm, which is proprietary. Distribution of modified UnRAR sources in separate form or as a part of other software is permitted, provided that it is clearly stated in the documentation and source comments that the code may not be used to develop a RAR (WinRAR) compatible archiver.3. The UnRAR utility may be freely distributed. No person or company may charge a fee for the distribution of UnRAR without written permission from the copyright holder.4. THE RAR ARCHIVER AND THE UNRAR UTILITY ARE DISTRIBUTED "AS IS". NO WARRANTY OF ANY KIND IS EXPRESSED OR IMPLIED. YOU USE AT YOUR OWN RISK. THE AUTHOR WILL NOT BE LIABLE FOR DATA LOSS, DAMAGES, LOSS OF PROFITS OR ANY OTHER KIND OF LOSS WHILE USING OR MISUSING THIS SOFTWARE.5. Installing and using the UnRAR utility signifies acceptance of these terms and conditions of the license.6. If you don't agree with terms of the license you must remove UnRAR files from your storage devices and cease to use the utility.

JasPer License Version 2.0Copyright (c) 2001-2006 Michael David AdamsCopyright (c) 1999-2000 Image Power, Inc.Copyright (c) 1999-2000 The University of British ColumbiaAll rights reserved.Permission is hereby granted, free of charge, to any person (the"User") obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:1. The above copyright notices and this permission notice (which includes the disclaimer below) shall be included in all copies or substantial portions of the Software.2. The name of a copyright holder shall not be used to endorse or promote products derived from the Software without specific prior written permission.THIS DISCLAIMER OF WARRANTY CONSTITUTES AN ESSENTIAL PART OF THIS LICENSE. NO USE OF THE SOFTWARE IS AUTHORIZED HEREUNDER EXCEPT UNDER THIS DISCLAIMER. THE SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, OR ANY SPECIAL INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS,

WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE. NO ASSURANCES ARE PROVIDED BY THE COPYRIGHT HOLDERS THAT THE SOFTWARE DOES NOT INFRINGE THE PATENT OR OTHER INTELLECTUAL PROPERTY RIGHTS OF ANY OTHER ENTITY. EACH COPYRIGHT HOLDER DISCLAIMS ANY LIABILITY TO THE USER FOR CLAIMS BROUGHT BY ANY OTHER ENTITY BASED ON INFRINGEMENT OF INTELLECTUAL PROPERTY RIGHTS OR OTHERWISE. AS A CONDITION TO EXERCISING THE RIGHTS GRANTED HEREUNDER, EACH USER HEREBY ASSUMES SOLE RESPONSIBILITY TO SECURE ANY OTHER INTELLECTUAL PROPERTY RIGHTS NEEDED, IF ANY. THE SOFTWARE IS NOT FAULT-TOLERANT AND IS NOT INTENDED FOR USE IN MISSION-CRITICAL SYSTEMS, SUCH AS THOSE USED IN THE OPERATION OF NUCLEAR FACILITIES, AIRCRAFT NAVIGATION OR COMMUNICATION SYSTEMS, AIR TRAFFIC CONTROL SYSTEMS, DIRECT LIFE SUPPORT MACHINES, OR WEAPONS SYSTEMS, IN WHICH THE FAILURE OF THE SOFTWARE OR SYSTEM COULD LEAD DIRECTLY TO DEATH, PERSONAL INJURY, OR SEVERE PHYSICAL OR ENVIRONMENTAL DAMAGE ("HIGH RISK ACTIVITIES"). THE COPYRIGHT HOLDERS SPECIFICALLY DISCLAIM ANY EXPRESS OR IMPLIED WARRANTY OF FITNESS FOR HIGH RISK ACTIVITIES.

Index

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W