Oracle® Outside In Viewer

Developer's Guide

Release 8.5
F10999-05
November 2023

Oracle Outside In Viewer Developer's Guide, Release 8.5

F10999-05

Copyright © 2010, 2023, Oracle and/or its affiliates.

Primary Author: Kalpana N

Contributing Authors: Nirmala Suryaprakasha, Promila Chitkara

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle®, Java, and MySQL are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

A Viewer Options

Preface

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

Audience

This document is intended for developers who are integrating Outside In Viewer 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

The complete Oracle Outside In Technology documentation set is available from the Oracle Help Center at http://www.oracle.com/pls/topic/lookup?ctx=oitlatest&id=homepage.

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

This chapter is an introduction to the Oracle Outside In Viewer module. The Viewer is a part of Oracle's Outside In family of OEM technologies, a powerful document viewing and conversion technology that can access the information in more than 600 file formats.

There may be references to other Outside In Technology SDKs within this manual. To obtain complete documentation for any other Outside In product, see Middleware documentation page and click Outside In Technology link below.

> **Note:**
>
> For new functionality information, see What's New guide.

This chapter includes the following sections:

1.1 What Does the Oracle Outside In Viewer Do?

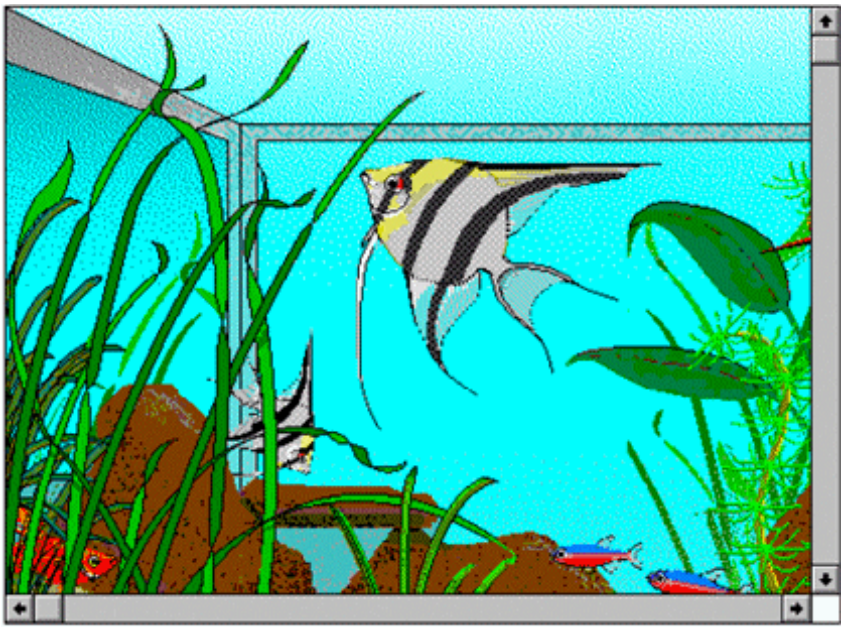
For each supported platform, the Oracle Outside In Viewer provides a way to create a rectangular view consisting of a horizontal scroll bar, a vertical scroll bar and a display area. This rectangle is referred to as the **view window**.

Figure 1-1 View Window

The view window is blank until the developer sends a SCCVW_VIEWFILE message. This message allows the developer to specify a file to be viewed. On some platforms you can specify a bitmap to display when no file is being viewed.

If SCCVW_VIEWFILE returns successfully, the display area will contain the top part of the file specified and scroll bars will be enabled, allowing the user to move around the rest of the file.

Figure 1-2 View Window in Use



Once a file is being viewed, a whole set of messages can be sent by the developer to the view window allowing the developer to print, copy, search, change the look of, and perform a number of other operations on the view.

Eventually, the developer will either close the file (SCCVW_CLOSE) and destroy the view window or make another call to SCCVW_VIEWFILE.

1.2 Architectural Overview

The basic architecture of the Oracle Outside In Viewer is noted below. There are no supported platform distinctions:

Filter/Module	Description
Input Filter	The input filters form the base of the architecture. Each one reads a specific file format or set of related formats and sends the data to the chunker module through a standard set of function calls. There are more than 150 of these filters that read more than 600 distinct file formats. Filters are loaded on demand by the data access module.
Chunker	The Chunker module is responsible for caching a certain amount of data from the filter and returning this data as a display engine. It is also responsible for running the filter to rebuild any data not already in the cache that is requested.
Display Engine	The Display Engine is responsible for reading data from the chunker and displaying it in the view window. It is also responsible for all user interface, clipboard and printing for a given data type. There are currently six display engines: Document, Spreadsheet/Database, Bitmap, Drawing, Archive and Hex. Display Engines are loaded on demand by the view window.
View Window	The View Window controls all of the modules for this product.

1.3 Definition of Terms

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

1.4 Directory Structure

Each Oracle Outside In product has an sdk directory, under which there is a subdirectory for each platform on which the product ships (for example, vw/sdk/vw_win-x86-32_sdk). Under each of these directories are the following two subdirectories:

- **redist**: Contains only the files that the customer is allowed to redistribute. These include all the compiled modules, filter support files, .xsd and .dtd files, cmmap000.bin, and third-party libraries, like freetype.
- **sdk**: Contains the other subdirectories that used to be at the root-level of an sdk (common, lib, resource, samplefiles, and samplecode (previously samples). In addition, one new subdirectory has been added, demo, that holds all of the compiled sample apps

and other files that are needed to demo the products. These are files that the customer should not redistribute (.cfg files, exportmaps, etc.).

In the root platform directory (for example, vw/sdk/vw_win-x86-32_sdk), there are two files:

- **README**: Explains the contents of the sdk, and that makedemo must be run in order to use the sample applications.
- **makedemo** (.bat platform-based): This script will copy (on Windows) the contents of …/redist into …/sdk/demo, so that sample applications can then be run out of the demo directory.

2
Windows Implementation Details

This chapter describes the implementation of the Viewer on the Windows platform. The Windows implementation of Oracle Outside In Viewer is delivered as a set of DLLs. When a developer uses the Windows LoadLibrary call to load sccvw.dll, the DLL registers a Window Class named SCCVIEWER. The developer can then create windows of this class (using the windows CreateWindow call) and use them like any other window. This class of window also accepts a number of Viewer-specific windows messages that implement all of the functions needed for viewing, printing, searching, etc.
For a list of the currently supported platforms, see Outside In Technology and click links under Certified Platforms and Supported Formats.

> **Note:**
>
> The 64-bit version of sccvw.dll will not load on an AMD-64 system without Visual C++ runtime version 12 installed. This happens because the system is missing the msvcr140.dll library, which is required. Users can download the required library from Microsoft page.

This chapter includes the following sections:

2.1 Installation

To install the demo version of the SDK, copy the contents of the ZIP archive (available on the web site) to a local directory of your choice.

This product requires the Visual C++ libraries included in the Visual C++ Redistributable Package available from Microsoft. There is a version of this package for the appropriate platform (x86 or x64) version of Windows. This can be downloaded from www.microsoft.com, by searching on the site for the following package:

- vcredist_x86.exe, or
- vcredist_x64.exe

The required download version is "Visual C++ Redistributable Packages for Visual Studio 2019."

Oracle Outside In requires the msvcr140.dll redistributable module.

The installation directory should contain the following directory structure.

Directory	Description
\redist	Contains a working copy of the Windows version of the technology.
\sdk\common	Contains the C include files needed to build or rebuild the technology.
\sdk\demo	Contains the compiled executables of the sample applications.
\sdk\lib	Contains the library (.lib) files needed for the products.
\sdk\resource	Contains localization resource files.
\sdk\samplecode	Contains a subdirectory holding the source code for a sample application.
\sdk\samplefiles	Contains sample input files authored in a variety of popular graphics, word processor, compression, spreadsheet and presentation applications.
\sdk\template	Contains template files based on which html is created.

2.1.1 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 Oracle 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, when an NSF file is embedded in another file, or with IOTYPE_UNICODEPATH. 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. On Windows, SCCID_LOTUSNOTESDIRECTORY should be set to the directory containing the nnotes.dll. NSF support is only available on the Win32 and Win x86-64 platforms.

2.2 Libraries and Structure

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
sccca.dll	Content Access module (provides organized chunker data for the developer)
sccda.dll	Data Access module

File	Description
sccfi.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.
sccta.dll	Text Access module (provides straight text data for the developer)
sccvw.dll	Viewer module (this is the DLL that your application loads, providing control of all viewer functions)

Support DLLs

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)
sccanno.dll	Annotation module
sccch.dll	Chunker (provides caching of and access to filter data for the display engine)
sccdu.dll	Display Utilities module (includes text formatting)
sccfa.dll	Filter Access module
sccfmt.dll	Formatting module (resolves numbers to formatted strings)
sccfut.dll	Filter utility module
sccind.dll	Indexing engine
scclo.dll	Localization library (all strings, menus, dialogs and dialog procedures reside here)
sccole.dll	OLE rendering module
sccut.dll	Utility functions (including IO subsystem)
sccvw.dll	The DLL that your application loads, providing control of all viewer functions
wvcore.dll	The GDI Abstraction layer

Display Engine DLLs

File	Description
debmp.dll	Bitmap (TIFF, GIF, BMP, PCX…)
dehex.dll	Hexadecimal
devect.dll	Vector, Presentation (PowerPoint, Impress, Freelance…)
dess.dll	Spreadsheet/Database (Excel, Calc, Lotus 123…)
detree.dll	Archive (ZIP, GZIP, TAR…)
dewp.dll	Document (Word, Writer, WordPerfect…)

Filter and Export Filter Libraries

File	Description
vs*.dll	Filters for specific file types (there are more than 150 of these filters, covering more than 600 file formats)
oitnsf.id	Support file for the vsnsf filter.

Premier Graphics Filters

File	Description
i*2.dll	Import filters for premier graphics formats
isgdi32.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)
cmmap000.sbc	Tables for character mapping (single-byte character sets). Located in the common directory.
cmmap000.dbc	Identical to cmmap000.Bin, but renamed for clarity (.dbc = double-byte character). This file is located in the common directory.

2.3 The Basics

All the steps outlined in this section are used in the sample applications provided with the SDK. Looking at the code for the simple sample application is recommended for those wishing to see a real-world example of this process.

For detailed information about all sample applications included with this product, see Sample Applications.

2.3.1 What You Need in Your Source Code

Any source code that uses this product should #include the file sccvw.h and #define WINDOWS and WIN32 or WIN64. For example, a Windows application might have a source file with the following lines:

```
#define WINDOWS         /* Will be automatically defined if your
                           compiler defines _WINDOWS */
#define WIN32
#include <sccvw.h>
```

2.3.2 Options and Information Storage

The technology creates the default options. In the Windows implementations, this is built by the technology as needed, usually the first time the software is run. You do not

need to ship this list with your application. The list is 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 will be created automatically by the technology. The files are automatically regenerated if corrupted or deleted.

The files are:

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

> **Note:**
>
> Some applications and services may run under a local system account for which there is no users "application data" folder. In that case, the technology checks for the following environment variables in order: OIT_DATA_PATH, APPDATA, and LOCALAPPDATA. If none of those exist or is writable, the technology attempts to write the options files into the executable path of the UT module. The technology can still run if it cannot write the options files. However, performance will be significantly impeded.

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 Oracle 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.

2.3.3 Structure Alignment

Oracle Outside In is built with 8-byte structure alignment. This is the default setting for most Windows compilers. This and other compiler options that should be used are demonstrated in the files provided with the sample applications in samples\win.

2.3.4 Loading the Viewer DLL

The DLL named sccvw.dll is loaded using a LoadLibrary call as follows;

```
g_ViewDllHnd = LoadLibrary("SCCVW.DLL");
```

In this DLL's DllEntryPoint function, it registers a window class that can then be used by the developer to create view windows.

All the Viewer's DLLs must reside in the same directory. Do NOT put them in the Windows or System directory.

All the sample applications #include a file called scclink.c (in the SDK common directory) that implements loading of the sccvw.dll from any directory. Comments in this code explain the

problems involved in loading a DLL from a different directory and how these problems are solved in the code.

2.3.5 Creating a View Window

A simple example of view window creation code is:

```
g_ViewWnd = CreateWindow("SCCVIEWER",...);
```

All the additional parameters to CreateWindow are totally up to the developer. For instance, the dwStyle parameter can be WS_CHILD, WS_POPUP, WS_OVERLAPPED or any other set of styles.

The view window checks the class of its parent when it is created. If the class of its parent is "MDICLIENT", it will call DefMDIChildProc for default message processing. If the class is something else, it will call DefWindowProc for default message processing. This allows the view window to be used directly as a MDI child window. See the sample MDIVIEW for an example of this.

2.3.6 Sending SCCVW Messages

Once the developer has a handle to a view window, any of the SCCVW messages that make up the bulk of this document may be sent to the window (using SendMessage). The most likely action at this point would be to send SCCVW_VIEWFILE to view a file.

2.3.7 Receiving SCCVW Messages

The parent of the view window receives all SCCVW notification messages as Windows messages.

2.3.8 Unloading the Viewer DLL

When the developer's application no longer needs the Viewer (usually on exit), it must free the viewer DLL as follows:

```
FreeLibrary(g_ViewDllHnd);
```

Windows does not automatically free DLLs that are loaded using LoadLibrary. This step is necessary.

2.4 Character Sets

This section provides information about supported character sets.

2.4.1 Default API Character Set

The strings passed in the Windows API are ANSI1252 by default.

2.4.2 Double-Byte Character Set Mapping

Please note that to optimize performance on systems that do not require DBCS support, a second character mapping bin file, that does not contain any of the DBCS pages, is now included. The second bin file will give additional performance benefits

for English documents, but will not be able to handle DBCS documents. To use the new bin file, replace the cmmap000.bin with the new bin file, cmmap000.sbc. For clarity, a copy of the cmmap000.bin file named cmmap000.dbc has also been included. Both the cmmap000.sbc and cmmap000.dbc files are located in the Common directory of the technology.

2.5 Runtime Considerations

The files used by this product must be in the same directory as the developer's executable.

2.6 Menus

The Viewer exposes a number of menus that developers can use at their discretion. The specifics of menu interactions under Windows are discussed here.

2.6.1 Context Menu

In keeping with the current trend in Windows of using the right mouse click to bring up a context menu, the Viewer pops up a view-window-specific context menu when the right mouse button is pressed. The context menu under Windows contains Copy, Print..., the display engine's menu, and access to the option dialogs. For example, the context menu that appears while viewing a spreadsheet appears in the following image:

Figure 2-1 Context Menu

If this meets your application's needs and user interface standards, you can ignore the interaction described in the next section and let the Viewer do all the work. If you wish to handle the right click yourself, this functionality can be overridden by handling the SCCVW_CONTEXTMENU message.

2.6.2 Menu Interaction

Each type of display engine (Document, Bitmap, Archive, etc.) has functions that are unique to the kinds of files it can view. For instance, zooming, rotation and dithering are functions associated with graphic images, but not with spreadsheets. To handle these specialized tasks, each display engine has a menu that is tailored specifically for it.

When the application receives a SCCVW_DISPLAYCHANGE message, it may call SCCVW_GETDISPLAYINFO. One of the elements returned by this message is a handle to a

pop-up menu. The application may then do whatever it likes with this menu (except destroy it), such as saving it to be popped up at another time.

2.7 Default Font Aliases

The technology includes the following default font alias map for Windows. The first value is the original font, the second is the alias.

- Chicago = Arial
- Geneva = Arial
- Helv = Arial
- Helvetica = Arial
- itc zapf dingbats = Zapfdingbats
- itc zapfdingbats = Zapfdingbats
- New York = Times New Roman
- times = Times New Roman
- Times = Times New Roman
- Tms Roman = Times New Roman

This list can be applied with SCCID_DISPLAYFONTALIAS and SCCID_PRINTFONTALIAS using the flag SCCVW_FONTALIAS_USEDEFAULTS.

2.8 File Open Modes

Files to be viewed are opened using the GENERIC_READ, FILE_SHARE_READ, and FILE_SHARE_WRITE flags under Windows. This should allow developers to launch an application and edit the file (occasionally, the application launches but the file is read-only) while still viewing the file.

However, if you wish to launch a file you are viewing in its native application, it is recommended that you send SCCVW_CLOSEFILE to the view window first.

2.9 Changing Resources

Oracle Outside In Viewer ships with the necessary files for OEMs to change any of the menus or strings in the technology as they see fit.

Strings are stored in the lodlgstr.h file found in the resource directory. The file can be edited using any text editor.

> **Note:**
>
> Do not directly edit the scclo.rc file. Strings are saved with their identifiers in lodlgstr.h. If a new scclo.rc file is saved, it will contain numeric identifiers for strings, instead of their #define'd names.
>
> When saving resource strings in character sets that require multi-byte characters, make sure to use a text editor that supports these characters, and save the file as appropriate for the character set being used.

Once the changes have been made, the updated scclo.dll file can be rebuilt using the following steps:

1. Compile the .res file:

   ```
   rc /fo ".\scclo.res" /i "<path to header (.h) files folder>" /d "NDEBUG" scclo.rc
   ```

2. Link the scclo.res file you've created with the scclo.obj file found in the resource directory to create a new scclo.dll:

   ```
   link /DLL /OUT:scclo.dll scclo.obj scclo.res
   ```

   > **Note:**
   >
   > Developers should make sure they have set up their environment variables to build the library for their specific architecture. For Windows x86_32, when compiling with VS 2013, the solution is to run vsvars32.bat (in a standard VS 2013 installation, this is found in C:\Program Files (x86)\Microsoft Visual Studio 12.0\Common7\Tools). If this works correctly, you will see the statement, "Setting environment for using Microsoft Visual Studio 2013 x86 tools." If you do not complete this step, you may have conflicts that lead to unresolved symbols due to conflicts with the Microsoft CRT.

3. Embed the manifest (which is created in the \resource directory during step 2) into the new DLL:

   ```
   mt -manifest scclo.dll.manifest -outputresource:scclo.dll;2
   ```

If you are not using Microsoft Visual Studio, substitute the appropriate development tools from your environment.

> **Note:**
>
> In previous versions of Oracle Outside In, it was possible to edit SCCLO.DLL directly in Microsoft Visual Studio. Oracle Outside In DLLs are now digitally signed. Editing the signed DLL is not advisable.

3
Using the View Window

This chapter describes the view window. The process of accessing the Viewer, creating the view window and sending/receiving messages is platform-dependent. However, once you have successfully created a view window, the message and structures for viewing files and manipulating the view window are fully portable.
This chapter includes the following sections:

3.1 Viewing

Files are viewed by sending the SCCVW_VIEWFILE message. This message allows the caller to specify the file to be viewed in a number of ways based on the operating system being used. For instance, under Windows, the developer can specify a path name in Unicode, ANSI or OEM character sets. The message also allows the developer to specify a number of other parameters, including a name to use in display situations (this is helpful if the file is a temporary file and the real file name is known), a flag indicating the file should be deleted when it is closed (again this can be helpful with temporary files), and a parameter that forces the file to be viewed as a text or hex file.

At any time after an SCCVW_VIEWFILE call, the developer may call SCCVW_GETFILEINFO. This message returns information on the file being viewed and a text string indicating the file's format (for example, WordPerfect 4.2 or Excel 5.0). This message is often called in response to the SCCVW_FILECHANGE callback message.

In certain situations the viewer may request that another file be viewed by sending an SCCVW_VIEWTHISFILE message to the developer. The developer may ignore this message, in which case nothing will happen, or the developer may create another view window and pass on the request through the SCCVW_VIEWFILE message. There are currently two cases where this happens. The first is when the user double clicks on an entry in a view of an archive file. The archive display engine decompresses that file into a temporary file and issues a SCCVW_VIEWTHISFILE message to the developer. The second case is when the user double clicks on a non-OLE graphic that is embedded or linked to a file. Again the graphic is extracted to a temporary file and an SCCVW_VIEWTHISFILE is

sent. These cases represent significant functionality to the user and it is recommended that all OEMs support the SCCVW_VIEWTHISFILE message.

- SCCVW_VIEWFILE: View a file
- SCCVW_FILECHANGE: Notifies developer that the file being viewed has changed
- SCCVW_GETFILEINFO: Gets information about the currently viewed file
- SCCVW_VIEWTHISFILE: Notifies developer that another file should be viewed

3.2 Printing

The full printing functionality of the view window can be reached by sending a single SCCVW_PRINT message. This message is totally automatic, handling all dialogs and other user interactions needed to print on any given platform. The viewer follows each operating system's guidelines for how the print UI should work.

If more control over the printing process is required, the SCCVW_PRINTEX message may be used. This message allows the developer to override any of the view window's print options and specify extra platform-specific print options while still having the view window do most of the work. For instance, under Windows, the developer may pass his or her own printer DC or specify an alternate AbortProc to use.

- SCCVW_PRINT: Prints the file currently being viewed
- SCCVW_PRINTEX: Prints the file under developer control
- SCCVW_PRINTSETUP: Invokes platform dependent print setup dialog

3.3 Copying

The Viewer activates copy/paste functionality for each supported platform via the SCCVW_COPYTOCLIP message. When the developer calls this message, the view window generates one or more renditions of the area selected in the view and transfers them to the operating system using appropriate system calls. The option SCCID_TOCLIPBOARD allows the developer to tailor what is sent to the operating system.

At any time, the developer may call SCCVW_GETCLIPINFO to see if anything is selected in the view window. The developer will receive an SCCVW_SELCHANGE message when the selection status has changed. Used in conjunction, these two messages can be used to enable or disable a Copy button or menu item.

- SCCVW_COPYTOCLIP: Copies current selection to clipboard
- SCCVW_SELCHANGE: Notifies developer that the selection state has changed
- SCCVW_GETCLIPINFO: Gets the current state of the selection

3.4 Menus, Dialogs and Options

The Viewer provides a large array of options that give the developer and the user great control over how files look in the view window, how files get printed, how files are copied to the clipboard and a number of other aspects of the technology. The developer has two avenues available when providing these options to the user.

1. The developer can code his or her own dialogs and menus and call the SCCVW_SETOPTION and SCCVW_GETOPTION messages directly.
2. The developer can use the menus and dialogs that are part of the Viewer. Menus and dialogs are only available on certain platforms. See the implementation details for your platform for more information.

These courses are not mutually exclusive and developers may mix the viewer technologies menu and dialogs with their own to get just the level of functionality he or she desires.

3.4.1 Setting Options Directly

Options can be set with the SCCVW_SETOPTION message and retrieved with the SCCVW_GETOPTION message. Both of these messages can set either the current value of an option or the default value of an option. The current value is the value that is active in the view window. Setting the current value of an option has an immediate effect on the view window. The default value is the value that a new view window will begin with, setting this value will have no effect on the current view window.

There are cases where the developer may wish to monitor changes to an option for display or user-interaction purposes. The SCCVW_OPTIONCHANGE message is sent whenever an option is changed.

3.4.2 Dialog Boxes

There are three dialogs that the Viewer makes available to the developer. These dialogs are the display options dialog, the print options dialog and the clipboard options dialog. These dialogs are accessed through the SCCVW_DODIALOG message, and are totally automatic. All the developer has to do is call SCCVW_DODIALOG with the correct dialog ID and the rest of the interaction is handled by the Viewer.

The IDs used in SCCVW_DODIALOG are as follows:

- SCCID_DISPLAYOPTIONSDIALOG: display options
- SCCID_PRINTOPTIONSDIALOG: print options
- SCCID_CLIPBOARDOPTIONSDIALOG: clipboard options

3.4.3 Help in Dialogs

Whenever one of the Viewer's dialogs is invoked, the developer may receive SCCVW_DOHELP messages along with an SCCID indicating which dialog is displayed. The developer may then invoke help information for that dialog. There is no default action for this message, so the developer should either handle the message or set the SCCVW_DIALOG_NOHELP bit in the SCCID_DIALOGFLAGS option which removes the Help button from all the dialogs.

3.4.4 Display Engine Specific Menus

Each type of display engine (Document, Bitmap, Archive, etc.) has functions that are unique to that type of file. For instance, zooming, rotation and dithering are functions normally associated with graphic images, but not with spreadsheets. To handle these specialized tasks, each display engine has a menu that is tailored just for it. This menu is made available to the developer through the SCCVW_GETDISPLAYINFO message.

Some operating systems have the concept of a context menu activated by a right mouse click. On these systems, the view window will automatically handle the right click and pop-up the display engines menu plus some other menu items that are OS specific. This feature may be overridden by handling the SCCVW_CONTEXTMENU message.

The interaction between the menu, the application and the Viewer is operating-system specific.

- SCCVW_SETOPTION: Sets the current or default value of an option.
- SCCVW_GETOPTION: Gets the current or default value of an option.
- SCCVW_DODIALOG: Invokes a built-in dialog.
- SCCVW_DISPLAYCHANGE: Notifies the developer that the display engine has changed.
- SCCVW_GETDISPLAYINFO: Gets information about the current display engine including its menu.
- SCCVW_CONTEXTMENU: Notifies the developer that a context menu is going to be invoked and allows the developer to override it.
- SCCVW_DOHELP: Notifies the developer that the Help button has been pressed in a dialog.
- SCCVW_ENABLEAPP: Notifies the developer that the application should be disabled/enabled. Sent before and after dialogs and printing.

3.5 Searching

The view window offers basic text searching of documents, spreadsheets and databases. The developer can supply the UI and call SCCVW_SEARCH or let the viewer supply the UI by calling SCCVW_SEARCHDIALOG. In either case, the SCCVW_SEARCHNEXT message may then be called to repeat the search, searching forward or backward.

- SCCVW_SEARCH: Search for a text string
- SCCVW_SEARCHDIALOG: Bring up a search dialog and search
- SCCVW_SEARCHNEXT: Search for the next occurrence of a string

3.6 Raw Text and Annotations

During processing of any document, the developer has the option of receiving SCCVW_RAWTEXTEVENT messages. This message informs the developer that additional text is available. Once text is available, the developer can use SCCVW_GETRAWTEXT to retrieve blocks of raw text for any area of the document. Any text position can be mapped into a viewer position using SCCVW_MAPPOSITION. The developer can highlight text, hide text and/or insert icons and bitmaps at defined viewer positions using SCCVW_ADDANNOTATION. Once these annotations are added, the developer may use SCCVW_GOTOANNOTATION to jump to any given annotation or set of annotations. The developer may also receive SCCVW_ANNOTATIONEVENT messages when the user transitions, clicks, or double clicks on any annotation.

The format of a position in the document is a SCCVWPOS structure. The SCCVW_MAPPOSITION message provides a bidirectional method of translating between an actual character count (ACC) and an SCCVWPOS. The SCCVW_FINDPOSITION message provides the ability to locate a position based on certain input criteria including the current selection positions. SCCVW_SETSELECTION allows developer control of the selection. SCCVW_FINDANNOTATION provides a powerful method of locating an annotation using numerous options. SCCVW_DISPLAYPOSITION can then be used to bring the annotation into view. SCCVW_CLEARANNOTATIONS supports selective clearing of previously defined annotations. SCCVW_HILITESTYLE allows the developer to define a style that can then be referenced when adding highlight annotations.

Generally, this functionality has been used to highlight and move around search hits in indexing applications or to insert hyperlinks into the document.

To receive SCCVW_RAWTEXTEVENT messages, the developer must set the SCCVW_SYSTEM_RAWTEXT flag in the SCCVW_SYSTEMFLAGS option.

The value of this option is reset to 0 for every new view window, so it must be set every time a new window is created.

- SCCVW_RAWTEXTEVENT: Notifies the developer that more text is available.
- SCCVW_GETRAWTEXT: Returns raw text blocks.
- SCCVW_ADDANNOTATION: Adds an annotation into the view.
- SCCVW_CLEARANNOTATIONS: Clears some or all annotations.
- SCCVW_GOTOANNOTATION: Go to a certain annotation.
- SCCVW_ANNOTATIONEVENT: Notifies the developer that an action has been taken on an annotation.
- SCCVW_FINDANNOTATION: Finds an annotation given certain input criteria.
- SCCVW_FINDPOSITION: Finds a position given certain input criteria.
- SCCVW_HILITESTYLE: Defines an annotation highlight style for later reference when adding annotations.
- SCCVW_MAPPOSITION: Maps between an actual character count and SCCVWPOS.
- SCCVW_DISPLAYPOSITION: Finds an annotation given certain input criteria.
- SCCVW_SETSELECTION: Sets the current selection positions.

3.7 Drawing Pages

The view window has the ability to draw pages to an arbitrary area of an arbitrary output device. The developer must provide two rectangles, the first (output) is the actual area on the display device that should be filled. The second (format) is the area in twips (Windows) (1440 twips = one inch) that will be used to determine text positioning and wrapping. For instance, a Windows developer may want a thumbnail of a standard 8.5 x 11 inch page. In this case he would provide a small output area, maybe 85 x 110 pixels, but a 8.5 x 11 inch (converted to twips) format area.

Once it is initialized, the SCCVW_DRAWPAGE message can draw any page in the document.

- SCCVW_DRAWPAGE: Draws a page to the specified device
- SCCVW_INITDRAWPAGE: Initializes the DrawPage process

- SCCVW_DEINITDRAWPAGE: De-initializes the DrawPage process

3.8 Controlling the Scroll Bars

By default, the view window provides scroll bars for moving around the document, but some developers may have special UI considerations that make this behavior undesirable. The view window provides options and messages that allow the developer to override the view window's scrolling behavior. The SCCID_SCROLLFLAGS option allows the developer to selectively remove the horizontal and vertical scroll bars from the view window. The developer may then maintain his own scroll bars or other controls by handling the following notification messages:

- SCCVW_SETHSCROLLSTATE: Notifies the developer of a new state for the horizontal scroll bar.
- SCCVW_SETVSCROLLSTATE: Notifies the developer of a new state for the vertical scroll bar.
- SCCVW_SETHSCROLLPOSITION: Notifies the developer of a new position for the horizontal scroll bar's thumb.
- SCCVW_SETVSCROLLPOSITION: Notifies the developer of a new position for the vertical scroll bar's thumb.
- SCCVW_SETHSCROLLRANGE: Notifies the developer of a new range for the horizontal scroll bar.
- SCCVW_SETVSCROLLRANGEMIN : Notifies the developer of the minimum range for the vertical scroll bar.
- SCCVW_SETVSCROLLRANGEMAX : Notifies the developer of the maximum range for the vertical scroll bar.
- SCCVW_SETHSCROLLPAGESIZE: Notifies the developer of a new size for the horizontal scroll bar's thumb.
- SCCVW_SETVSCROLLPAGESIZE: Notifies the developer of a new size for the vertical scroll bar's thumb.

3.9 Character Sets and Character Encoding

There are three distinct areas where character sets (international language support) and character encoding (1 Byte, DBCS, 2 Byte) have an important role in the Viewer.

- In the strings passed between the developer and the viewer through this API.
- In the specification of files by the developer in the SCCVWVIEWFILE40 or SCCVWVIEWFILE80 structure.
- In the actual viewing of files containing various character sets and encoding.

3.9.1 In the API

All strings sent and received by the view window are defined to be in a particular character set/encoding for any given platform.

For example, using SCCVW_SETOPTION to set the SCCID_PRINTJOBNAME option does not let the developer define the character set or encoding of the string passed. It is pre-defined by the Viewer for the platform on which the software is running.

3.9.2 In File Specification

The sole exception to the rule described in the preceding section appears in the SCCVWVIEWFILE40 or SCCVWVIEWFILE80 structure, where the developer has a number of options in the character set/encoding of the file/path information. This gives the developer more flexibility in what he/she can pass through this critical API. For example, under Windows, the developer might want to pass an OEM, ANSI or Unicode path to the Viewer.

3.9.3 In Viewing

In the current implementation of the Viewer SDK, it is the viewer's responsibility to convert all character sets coming from the filters to an appropriate character set based on the available fonts. The technology includes support for most languages, including those based on DBCS. This creates two basic scenarios concerning character sets and how they are displayed.

- Scenario 1:

  If the file was created by an application native to the platform on which it is being viewed, the characters are generally piped directly through the system with no modification. This has the advantage of perfect character mapping even in fonts where the glyphs themselves are different. For example a Word for Windows file being viewed by the Windows version of the Viewer should view with all characters displaying correctly, even characters in symbol sets such as Wingdings.

- Scenario 2:

  If the file was created on a different platform or has a private character set (WordPerfect or DCA/RFT for example) the characters are mapped into the native character set using mapping tables. This produces loss of character integrity in a number of cases. These cases include characters that do not have a corresponding character in the target character set (for example, the WordPerfect set has a large number of symbols that do not appear in ANSI or Macintosh character sets) and fonts where the glyph set is different. The technology uses a robust character/font mapping architecture that attempts to locate a font containing the required character glyph when the original font is not available.

While these two cases cover most of the character set interactions you will see, the number of permutations on this subject is huge and space prohibits us from going into the caveats for every file format in this document. If you have specific questions concerning this subject please contact Oracle Support.

3.10 API Functions

This section provides information about API functions.

3.10.1 VWSetStatCallback

This function sets up a callback that the technology will periodically call into to verify that the file is still being processed. The customer can use this with a monitoring process to help identify files that may be hung. Since this function will be called more frequently than other callbacks, it is implemented as a separate function.

Use of the Status Callback Function

An application's status callback function will be called periodically by Oracle 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 Oracle 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 Oracle Outside In to abort.

The status callback function has two return values defined:

- OIT_STATUS_CONTINUE: Tells Oracle Outside In to continue processing the current document.
- OIT_STATUS_ABORT: Tells Oracle 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);
}
```

Prototype

```
DWORD VWSetStatCallback(VWSTATCALLBACKFN pCallback,
   VTHANDLE hUnique,
   VTSYSVAL pAppData)
```

Parameters

- pCallback: Pointer to a the callback function.
- hUnique: Handle to the view window.
- pAppData: User-defined data. Oracle Outside In will never use this value other than to provide it to the callback function.

The callback function should be of type VWSTATCALLBACKFN. This function has the following signature:

```
(VTHANDLE hUnique, VTDWORD dwID, VTSYSVAL pCallbackData, VTSYSVAL pAppData)
```

- hUnique: Handle for an hExport.
- dwID: Currently only OIT_STATUS_WORKING
- pCallbackData; Currently will always be NULL
- pAppData; User-defined data provided to VWSetStatCallback

Return Values

- SCCERR_OK: If successful. Otherwise, one of the error values in sccerr.h is returned.

4
Messages

This chapter describes the use of Messages in the Viewer. Messages are the primary way in which the developer and the view window interact. The implementation of message passing is different for each operating systems supported, but the messages themselves and the structures passed are for the most part generic.
Messages marked with an asterisk (*) are sent from the developer to the view window.

Messages marked with double asterisks (**) are sent from the view window to the developer.

Many messages pass pointers to structures in their lParam. All structures passed by the SCCVW messages are C structures. The packing of these structures compiled for Windows have 8-byte boundaries, per Microsoft's requirement for Windows applications.

4.1 SCCVW_ADDANNOTATION

* This message allows the parent to add various annotations to a viewed document.

Parameters

- wParam: Type of annotation to add, which should be one of the following:
  - SCCVW_HIDEPARABREAK: Removes a paragraph break. This value is only valid when viewing word processing, spreadsheet/database, or metafile documents. Note: This flag does not just cause the paragraph break to be whited out, but actually removes it from the view.
  - SCCVW_HIDETEXT: Removes an area of text. This may be useful if the document contains tags or information that is useful to your application but you do not want the user to see. Note: This flag does not just cause the text to be whited out, but actually removes the text from the view.
  - SCCVW_HILITETEXT: Change the foreground and background color of an area of text.
  - SCCVW_INSERTBITMAP: Insert a bitmap between two characters. This value is only valid when viewing word processing documents. Note: This flag is only supported on the Windows platform.
  - SCCVW_INSERTICON: Insert an icon between two characters. This value is only valid when viewing word processing documents. Note: This flag is only supported on the Windows platform.
  - SCCVW_INSERTPARABREAK: Insert a paragraph break between two characters. This value is only valid when viewing word processing, spreadsheet/database, or metafile documents.
  - SCCVW_INSERTTEXT: Insert text between two characters. This value is only valid when viewing word processing, spreadsheet/database, or metafile documents.
- lParam: A pointer to a structure of the type SCCVWHIDEPARABREAK80, SCCVWHIDETEXT80, SCCVWHILITETEXT41, SCCVWINSERTBITMAP42, SCCVWINSERTICON41, SCCVWINSERTPARABREAK80, or SCCVWINSERTTEXT80.

Also backwards compatible with SCCVWHILITETEXT40, SCCVWINSERTICON40, or SCCVWHIDETEXT40 and SCCVWHIDETEXT41.

Return Value

- SCCVWERR_OK: The annotation was added successfully.
- SCCVWERR_ALLOCFAILED: There is not enough memory to add the annotation.

Comments

Annotations can be added in any order, but the underlying code is optimized for additions to the end of the list. It takes many hundreds of annotations for this effect to be noticeable. The insertion of icons and bitmaps is only supported on the Windows platforms. Adding annotations to unprintable characters may not always display the annotation.

Platform

Windows

4.1.1 SCCVWHIDEPARABREAK80 Structure

This structure is used by the SCCVW_ADDANNOTATION message to specify a paragraph break to be deleted.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct SCCVWHIDEPARABREAK80tag
{
   VTDWORD    dwSize;
   VTDWORD    dwUser;
   SCCVWPOS   sParaPos;
   VTDWORD    dwData;
} SCCVWHIDEPARABREAK80,  * PSCCVWHIDEPARABREAK80;
```

Parameters

- dwSize: Must be set by the OEM to sizeof(SCCVWHIDEPARABREAK80).
- dwUser: Unique ID that may be used later to remove, jump to or otherwise identify this particular annotation or a set of annotations. Always initialize this value to 0 if you are not using this functionality. The use of the high bit is reserved for internal annotation tagging.
- sParaPos: Position of first paragraph break to be hidden. Result of SCCVWMAKEPOS(id,offset) macro or result of SCCVW_MAPPOSITION message. Note: Using the SCCVWMAKEPOS macro is no longer recommended but supported for backward compatibility. See the Comments section of the SCCVW_GETRAWTEXT message section.
- dwData: Additional data associated with this annotation. It can later be retrieved using SCCVW_FINDANNOTATION.

4.1.2 SCCVWHIDETEXT80 Structure

This structure is used by the SCCVW_ADDANNOTATION message to remove an area of text. Also backward compatible with SCCVWHIDETEXT40 and SCCVWHIDETEXT41.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct SCCVWHIDETEXT80tag
{
   VTDWORD    dwSize;
   VTDWORD    dwUser;
   SCCVWPOS   sStartPos;
   SCCVWPOS   sEndPos;
   VTDWORD    dwData;
} SCCVWHIDETEXT80,  * PSCCVWHIDETEXT80;
```

Parameters

- dwSize: Must be set by the OEM to sizeof(SCCVWHIDETEXT80).
- dwUser: Unique ID that may be used later to remove, jump to or otherwise identify this particular annotation or a set of annotations. Always initialize this value to 0 if you are not using this functionality. The use of the high bit is reserved for internal annotation tagging.
- sStartPos: Position of first character to be hidden. Result of SCCVW_MAPACCTOPOS OR result of SCCVWMAKEPOS(id,offset). Note: Using the SCCVWMAKEPOS macro is no longer recommended but supported for backward compatibility. See the Comments section of the SCCVW_GETRAWTEXT message section.
- sEndPos: Position of last character to be hidden. Result of SCCVW_MAPACCTOPOS OR result of SCCVWMAKEPOS(id,offset). Note: Using the SCCVWMAKEPOS macro is no longer recommended but supported for backward compatibility. See the Comments section of the SCCVW_GETRAWTEXT messages.
- dwData: Additional data associated with this annotation. It can later be retrieved using SCCVW_FINDANNOTATION or SCCVW_GETANNOTATIONINFO.

4.1.3 SCCVWHILITETEXT41 Structure

This structure is used by the SCCVW_ADDANNOTATION message to specify an area of text to color. Also backward compatible with SCCVWHILITETEXT40.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct SCCVWHILITETEXT41tag
   {
   VTDWORD    dwSize;
   VTDWORD    dwUser;
   SCCVWPOS   sStartPos;
   SCCVWPOS   sEndPos;
   VTDWORD    dwInteraction;
   VTDWORD    dwData;
   VTDWORD    dwDummy1;
   VTDWORD    dwDummy2;
```

```
VTDWORD    dwDisplay;
} SCCVWHILITETEXT41;
```

Parameters

- dwSize: Must be set by the OEM to sizeof(SCCVWHILITETEXT41).
- dwUser: Unique ID that may be used later to remove, jump to or otherwise identify this particular annotation or a set of annotations. Always initialize this value to 0 if you are not using this functionality. The use of the high bit is reserved for internal annotation tagging.
- sStartPos: Position of first character to be highlighted. Result of SCCVWMAKEPOS(id,offset) macro or result of SCCVW_MAPPOSITION message. Note: Using the SCCVWMAKEPOS macro is no longer recommended but supported for backward compatibility. See the Comments section of the SCCVW_GETRAWTEXT message section.
- sEndPos; Position of last character to be highlighted. Result of SCCVWMAKEPOS(id,offset) macro or result of SCCVW_MAPPOSITION message. Note: Using the SCCVWMAKEPOS macro is no longer recommended but supported for backward compatibility. See the Comments section of the SCCVW_GETRAWTEXT message section.
- dwInteraction: The type of events that the OEM would like to receive for this annotation. Note: Interaction events are only supported in word processor formats. Should be one or more of the following OR-ed together:
  - SCCVW_EVENTSINGLECLICK: If set, parent window will receive a SCCVW_ANNOTATIONEVENT when the highlighted text is clicked.
  - SCCVW_EVENTDOUBLECLICK: If set, parent window will receive a SCCCVW_ANNOTATIONEVENT when the highlighted text is double-clicked.
  - SCCVW_EVENTTRANSITIONINTO: If set, parent window will receive a SCCVW_ANNOTATIONEVENT when the cursor transitions into the annotation.
  - SCCVW_EVENTTRANSITIONOUTOF: If set, parent window will receive a SCCVW_ANNOTATIONEVENT when the cursor transitions out of the annotation.
- dwData: Additional data associated with this annotation. It can later be retrieved using SCCVW_FINDANNOTATION.
- dwDummy1; Reserved
- dwDummy2; Reserved
- dwDisplay: 32 bits packed with information about how the highlight should display. If SCCVW_USESTYLE is set, then the LOWORD contains the style number to use as defined by the SCCVW_HILITESTYLE message. If styles are not used, then set using one of the values in the following table:
  - SCCVW_BDEFAULT (Do not change the background color) OR-ed with SCCVW_FDEFAULT (Do not change the foreground color)
  - SCCVW_BBLACK (Black background) OR-ed with SCCVW_FBLACK (Black foreground)
  - SCCVW_BDARKRED (Dark Red background) OR-ed with SCCVW_FDARKRED (Dark Red foreground)

- SCCVW_BDARKGREEN (Dark Green background) OR-ed with SCCVW_FDARKGREEN (Dark Green foreground)
- SCCVW_BDARKYELLOW (Dark Yellow background) OR-ed with SCCVW_FDARKYELLOW (Dark Yellow foreground)
- SCCVW_BDARKBLUE (Dark Blue background) OR-ed with SCCVW_FDARKBLUE (Dark Blue foreground)
- SCCVW_BDARKMAGENTA (Dark Magenta background) OR-ed with SCCVW_FDARKMAGENTA (Dark Magenta foreground)
- SCCVW_BDARKCYAN (Dark Cyan background) OR-ed with SCCVW_FDARKCYAN (Dark Cyan foreground)
- SCCVW_BLIGHTGRAY (Light Gray background) OR-ed with SCCVW_FLIGHTGRAY (Light Gray foreground)
- SCCVW_BGRAY (Gray background) OR-ed with SCCVW_FGRAY (Gray foreground)
- SCCVW_BRED (Red background) OR-ed with SCCVW_FRED (Red foreground)
- SCCVW_BGREEN (Green background) OR-ed with SCCVW_FGREEN (Green foreground)
- SCCVW_BYELLOW (Yellow background) OR-ed with SCCVW_FYELLOW (Yellow foreground)
- SCCVW_BBLUE (Blue background) OR-ed with SCCVW_FBLUE (Blue foreground)
- SCCVW_BMAGENTA (Magenta background) OR-ed with SCCVW_FMAGENTA (Magenta foreground)
- SCCVW_BCYAN (Cyan background) OR-ed with SCCVW_FCYAN (Cyan foreground)
- SCCVW_BWHITE (White background) OR-ed with SCCVW_FWHITE (White foreground)

4.1.4 SCCVWINSERTBITMAP42 Structure

This structure is used by the SCCVW_ADDANNOTATION message to insert a bitmap at a specific location in the text.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct
   {
   VTDWORD    dwSize;
   VTDWORD    dwUser;
   VTDWORD    dwData;
   SCCVWPOS   sBitmapPos;
   HICON      hBitmap;
   VTDWORD    dwInteraction;
   } SCCVWINSERTBITMAP42;
```

Parameters

- dwSize: Must be set by the OEM to sizeof(SCCVWINSERTBITMAP42).

- dwUser: Unique ID that may be used later to remove, jump to, or otherwise identify this particular annotation or a set of annotations. Always initialize this value to 0 if you are not using this functionality. The use of the high bit is reserved for internal annotation tagging.
- dwData: Additional data associated with this annotation. It can later be retrieved using SCCVW_FINDANNOTATION or SCCVW_GETANNOTATIONINFO.
- sBitmapPos; Position of the character before which the icon will be appear. Result of SCCVWMAKEPOS(id,offset) macro OR result of SCCVW_MAPACCTOPOS. Note: Using the SCCVWMAKEPOS macro is no longer recommended but supported for backward compatibility. See the Comments section of the SCCVW_GETRAWTEXT message section.
- hBitmap: Handle to the Windows bitmap to display.
- dwInteraction: The type of events that the OEM would like to receive for this annotation. Note: Interaction events are only supported in word processor formats. Should be one or more of the following OR-ed together:
  - SCCVW_EVENTSINGLECLICK: If set, parent window will receive an SCCVW_ANNOTATIONEVENT when the icon is clicked.
  - SCCVW_EVENTDOUBLECLICK: If set, parent window will receive an SCCCVW_ANNOTATIONEVENT when the icon is double-clicked.
  - SCCVW_EVENTTRANSITIONINTO: If set, parent window will receive an SCCVW_ANNOTATIONEVENT when the cursor transitions into the icon.
  - SCCVW_EVENTTRANSITIONOUTOF: If set, parent window will receive an SCCVW_ANNOTATIONEVENT when the cursor transitions out of the icon.

4.1.5 SCCVWINSERTICON41 Structure

This structure is used by the SCCVW_ADDANNOTATION message insert an icon at a specific location in the text. Also backward compatible with SCCVWINSERTICON40.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct
   {
   VTDWORD    dwSize;
   VTDWORD    dwUser;
   VTDWORD    dwData;
   SCCVWPOS   sIconPos;
   HICON      hIcon;
   VTDWORD    dwInteraction;
   } SCCVWINSERTICON41;
```

Parameters

- dwSize: Must be set by the OEM to sizeof(SCCVWINSERTICON41).
- dwUser: Unique ID that may be used later to remove, jump to, or otherwise identify this particular annotation or a set of annotations. Always initialize this value to 0 if you are not using this functionality. The use of the high bit is reserved for internal annotation tagging.
- dwData: Additional data associated with this annotation. It can later be retrieved using SCCVW_FINDANNOTATION or SCCVW_GETANNOTATIONINFO.

- sIconPos: Position of the character before which the icon will be appear. Result of SCCVWMAKEPOS(id,offset) macro OR result of SCCVW_MAPACCTOPOS. Note: Using the SCCVWMAKEPOS macro is no longer recommended but supported for backward compatibility. See the Comments section of the SCCVW_GETRAWTEXT message section.
- hIcon: Handle to the icon to display.
- dwInteraction: The type of events that the OEM would like to receive for this annotation. Note: Interaction events are only supported in word processor formats. Should be one or more of the following OR-ed together:
  - SCCVW_EVENTSINGLECLICK: If set, parent window will receive an SCCVW_ANNOTATIONEVENT when the icon is clicked.
  - SCCVW_EVENTDOUBLECLICK: If set, parent window will receive an SCCCVW_ANNOTATIONEVENT when the icon is double-clicked.
  - SCCVW_EVENTTRANSITIONINTO: If set, parent window will receive an SCCVW_ANNOTATIONEVENT when the cursor transitions into the icon.
  - SCCVW_EVENTTRANSITIONOUTOF: If set, parent window will receive an SCCVW_ANNOTATIONEVENT when the cursor transitions out of the icon.

4.1.6 SCCVWINSERTPARABREAK80 Structure

This structure is used by the SCCVW_ADDANNOTATION message to insert a paragraph break between two characters.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct SCCVWINSERTPARABREAK80tag
{
   VTDWORD    dwSize;
   VTDWORD    dwUser;
   SCCVWPOS   sParaPos;
   VTDWORD    dwData;
} SCCVWINSERTPARABREAK80,  * PSCCVWINSERTPARABREAK80;
```

Parameters

- dwSize: Must be set by the OEM to sizeof(SCCVWINSERTPARABREAK80).
- dwUser: Unique ID that may be used later to remove, jump to or otherwise identify this particular annotation or a set of annotations. Always initialize this value to 0 if you are not using this functionality. The use of the high bit is reserved for internal annotation tagging.
- sParaPos: Position before which the paragraph break will be inserted. Result of SCCVWMAKEPOS(id,offset) macro or result of SCCVW_MAPPOSITION message. Note: Using the SCCVWMAKEPOS macro is no longer recommended but supported for backward compatibility. See the Comments section of the SCCVW_GETRAWTEXT message section.
- dwData: Additional data associated with this annotation. It can later be retrieved using SCCVW_FINDANNOTATION.

4.1.7 SCCVWINSERTTEXT80 Structure

This structure is used by the SCCVW_ADDANNOTATION message to insert text between two characters.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct SCCVWINSERTTEXT80tag
{
   VTDWORD    dwSize;
   VTDWORD    dwUser;
   SCCVWPOS   sTextPos;
   VTDWORD    dwData;
   VTLPWORD   pText;
   VTDWORD    dwLength;
} SCCVWINSERTTEXT80,  * PSCCVWINSERTTEXT80;
```

Parameters

- dwSize: Must be set by the OEM to sizeof(SCCVWINSERTTEXT80).
- dwUser: Unique ID that may be used later to remove, jump to or otherwise identify this particular annotation or a set of annotations. Always initialize this value to 0 if you are not using this functionality. The use of the high bit is reserved for internal annotation tagging.
- sTextPos: Position before which the text will be inserted. Result of SCCVWMAKEPOS(id,offset) macro or result of SCCVW_MAPPOSITION message. Note: Using the SCCVWMAKEPOS macro is no longer recommended but supported for backward compatibility. See the Comments section of the SCCVW_GETRAWTEXT message section.
- dwData: Additional data associated with this annotation. It can later be retrieved using SCCVW_FINDANNOTATION.
- pText: The text buffer (Unicode)
- dwLength: The number of characters

4.2 SCCVW_ANNOTATIONEVENT

* * This message informs the OEM that some user action has been taken on an annotation.

Parameters

- wParam: The handle of the view window sending this message.
- lParam: Pointer to a structure of the type SCCVWANNOTATIONEVENT41. Also backward compatible with SCCVWANNOTATIONEVENT40.

Return Value

If the return value is 0, the Viewer will perform the default behavior for this event on this annotation. If the return value is anything but 0, the Viewer will not perform any action on this annotation.

Comments

The transition events may be generated due to mouse or keyboard movement of the cursor. This message is only supported for annotations added to word processor formats. The OEM will always receive a single-click event before a double-click event.

Platform

Windows

4.2.1 SCCVWANNOTATIONEVENT41 Structure

This structure is used by the SCCVW_ANNOTATIONEVENT message to inform the OEM what event has occurred. Also backward compatible with SCCVWANNOTATIONEVENT40.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct
   {
   VTDWORD      dwSize;
   VTDWORD      dwEvent;
   VTDWORD      dwUser;
   VTDWORD      dwData;
   } SCCVWANNOTATIONEVENT41;
```

Parameters

- dwSize: Will be set to sizeof(SCCVWANNOTATIONEVENT41).
- dwEvent: One of the following:
  - SCCVW_EVENTSINGLECLICK: The annotation specified in dwUser has been clicked.
  - SCCVW_EVENTDOUBLECLICK: The annotation specified in dwUser has been double clicked.
  - SCCVW_EVENTRIGHTCLICK: This message is sent when the user right-clicks on the annotation. If the host application indicates that it will not handle this event (i.e., returns zero from the SCCVW_ANNOTATIONEVENT message), then the viewer sends the SCCVW_CONTEXTMENU message to the host application. If the host application returns non-zero from the SCCVW_ANNOTATIONEVENT message, indicating that it is handling the right-click event, then the viewer will not send the SCCVW_CONTEXTMENU message.
  - SCCVW_EVENTTRANSITIONINTO: The annotation specified in dwUser has had the cursor transition into it.
  - SCCVW_EVENTTRANSITIONOUTOF: The annotation specified in dwUser has had the cursor transition out of it.
- dwUser: Identifies the annotation using the dwUser value passed in SCCVW_ADDANNOTATION. If you want to handle these events correctly, each of your annotations should have a unique dwUser.
- dwData: Returns the value passed when this annotation was created using SCCVW_ADDANNOTATION.

4.3 SCCVW_BAILOUT

* * This message informs the OEM that file viewing has stopped because of an error or some other reason. The developer may decide to handle this message himself, or allow the default action, which will display an appropriate message.

Parameters

- wParam: Should be 0.
- lParam: The value of lParam describes the reason for the bailout, outlined as follows:
  - SCCVW_BAILOUT_BADCREDENTIALS: the password suppiled is invalid.
  - SCCVW_BAILOUT_BADFILE: The file viewed is not valid for that type.
  - SCCVW_BAILOUT_CANCEL: the OEM responded to a callback function requesting that we cancel processing the file.
  - SCCVW_BAILOUT_DIVIDEBYZERO: There was a divide by zero operation attempted.
  - SCCVW_BAILOUT_DRMFILE: the file is protected by a DRM system
  - SCCVW_BAILOUT_EMPTYFILE: The file has no data to be displayed.
  - SCCVW_BAILOUT_FILECHANGED: The file has changed and is no longer valid.
  - SCCVW_BAILOUT_FILEOPENFAILED: There was a problem with the information in the header.
  - SCCVW_BAILOUT_FILTERTIMEOUT: The filter has timed out.
  - SCCVW_BAILOUT_GPFAULT: There was a general protection fault.
  - SCCVW_BAILOUT_MEMORY: There was a memory allocation failure.
  - SCCVW_BAILOUT_MISSINGELEMENT: There were other files needed that are not available.
  - SCCVW_BAILOUT_NODATATORENDER: There is no data in this format to render.
  - SCCVW_BAILOUT_NOENGINE: The display engine needed for this type is not available.
  - SCCVW_BAILOUT_NOFILTER: There is no filter installed for the files type.
  - SCCVW_BAILOUT_NOSUPPORTEDFILE: This file type is not supported.
  - SCCVW_BAILOUT_OTHEREXCEPTION: There was an unspecified system exception.
  - SCCVW_BAILOUT_PROTECTEDFILE: The file viewed is password protected.
  - SCCVW_BAILOUT_STREAMBAIL: There was a problem in the filter.
  - SCCVW_BAILOUT_SUPFILEOPENFAILS: There was a problem opening extra files.
  - SCCVW_BAILOUT_UNKNOWN: General error.

  - SCCVW_BAILOUT_UNKNOWNNOTVIEWED: This file type is unknown and the option has been set to not display unknown file types.
  - SCCVW_BAILOUT_WRITEERROR: There was a write error during viewing.

Return Value

If the return value is 0, the Viewer will perform the default behavior for this event which is to display an appropriate message. If the return value is anything but 0, the Viewer will only clear the display window.

Platform

Windows

4.4 SCCVW_CLEARANNOTATIONS

* This message allows the developer to selectively remove annotations.

Parameters

- wParam: If set to 0 then all annotations are cleared. Otherwise, set to one of the following:
  - SCCVW_ABSOLUTE: lParam represents an absolute dwUser value as set by the OEM in a SCCVW_ADDANNOTATION message.
  - SCCVW_MASK: lParam represents a mask to be applied to the dwUser values of the annotations (lParam is described in the next table cell).
- lParam: If wParam contains SCCVW_ABSOLUTE, the set of annotations that are cleared is limited to those annotations where dwUser == lParam.

  If wParam contains SCCVW_MASK, the set of annotations that are cleared is limited to those annotations where dwUser & lParam == lParam.

Platforms

Windows

4.5 SCCVW_CLOSEFILE

* Causes the viewer to stop viewing the current file and return to its idle state.

Parameters

- wParam: Should be set to 0.
- lParam: Should be set to 0.

Return Value

none

Platform

Windows

4.6 SCCVW_COMPPOSITIONS

* This message allows the parent to compare two SCCVWPOSs. OEMs should only compare SCCVWPOS variables using this message since the structure size of SCCVWPOS is subject to change.

Parameters

- wParam: Should be set to 0.
- lParam: Pointer to a structure of the type SCCVWCOMPPOSITIONS41.

Return Value

One of the following values:

- -1: The position identified by pPosA is less than the position defined by pPosB
- 0: The two positions identified by pPosA and pPosB are equal
- 1: The position identified by pPosA is greater than the position defined by pPosB

Platforms

Windows

4.6.1 SCCVWCOMPPOSITIONS41 Structure

This structure is passed by the OEM through the SCCVW_COMPPOSITIONS message.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct
   {
   PSCCVWPOS      pPosA;
   PSCCVWPOS      pPosB;
   } SCCVWCOMPPOSITIONS41;
```

Parameters

- pPosA: Points to an SCCVWPOS structure that identifies the first of two positions being compared.
- pPosB: Points to an SCCVWPOS structure that identifies the second of two positions being compared.

4.7 SCCVW_CONTEXTMENU

* * This message is sent to the parent of the view window when a right mouse click occurs inside the view window. The developer may decide to handle that interaction, or allow the Viewer to pop up its context menu.

Parameters

- wParam: The handle of the view window that is sending this message
- lParam(Windows): LOWORD(lParam) contains the x coordinate of the cursor. HIWORD(lParam) contains the y coordinate of the cursor.

Return Value

If the return value is 0, the Viewer will pop up its own context menu. If the return value is anything but 0, the Viewer does nothing.

Platform

Windows

4.8 SCCVW_COPY

* This message allows the parent to copy the area defined by two positions into memory in a given format.

Parameters

- wParam: Should be set to 0.
- lParam: Pointer to a structure of the type SCCVWCOPY41.

Return Value

One of the following values:

- SCCVWERR_OK: The copy was successful.
- SCCVWERR_BADPARAM: One of the values in the SCCVWCOPY41 structure was invalid.

Platforms

Windows

4.8.1 SCCVWCOPY41 Structure

This structure is passed by the OEM through the SCCVW_COPY message.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct
   {
   VTDWORD      dwSize;
   VTDWORD      dwOptions;
   VTDWORD      dwFormatId;
   SCCVWPOS     sStartPos;
   SCCVWPOS     sEndPos;
   VTDWORD      dwDataHandle;
   VTDWORD      dwDataSize;
   VTDWORD      dwMessageId;
   } SCCVWCOPY41;
```

Parameters

- dwSize: Must be set by the OEM to sizeof(SCCVWCOPY41).
- dwOptions: None defined, must be set to 0.
- dwFormatId: Indicates the output format. Must be one of the following:
  - SCCVW_CLIPFORMAT_TEXT: Text in whatever character set is appropriate for the operating system
  - SCCVW_CLIPFORMAT_RTF: Rich Text Format
- sStartPos: Identifies the start position of the area to be copied.
- sEndPos: Identifies the end position of the area to be copied.
- dwDataHandle: Returns an operating system handle to the copied data.
- dwDataSize: The size of the data produced in dwDataHandle.
- dwMessageId: Identifies the copy result. 0 if the copy was successful, otherwise one of the following:
  - SCCID_VWCLIP_NOMEMORY: Not enough memory for data.
  - SCCID_VWCLIP_NOEMBEDDED: An informational ID indicating the selected area contains an embedding but the embedding is not provided in the data. Note: This will only copy data to the text formats. It is not supported for bitmap, vector or archive formats.

4.9 SCCVW_COPYTOCLIP

* Causes the viewer to copy the current selection (if any) to the clipboard.

Parameters

- wParam: Should be set to 0.
- lParam: Should be set to 0.

Return Value

none

Platforms

Windows

4.10 SCCVW_DEINITDRAWPAGE

* This message de-initializes the page drawing routines. This message must be called after all calls to SCCVW_DRAWPAGE.

Parameters

- wParam: Should be set to 0.
- lParam: Should be set to 0.

Platforms

Windows

4.11 SCCVW_DISPLAYCHANGE

* *This message is sent to the developer whenever the display engine changes.

Parameters

- wParam: The handle of the view window that is sending this message
- lParam: Not used.

Comments

A reasonable response to this message might be to call SCCVW_GETDISPLAYINFO, if you need this kind of information.

Platforms

Windows

4.12 SCCVW_DISPLAYPOSITION

* This message allows the parent to bring any position into view and provides some control over the placement of the position in the view.

Parameters

- wParam: Should be set to 0.
- lParam: Pointer to a structure of the type SCCVWDISPLAYPOSITION41

Return Value

One of the following values:

- SCCVWERR_OK: The mapping was successful.
- SCCVWERR_BADPARAM: One of the values in the SCCVWDISPLAYPOSITION41 structure was invalid.

Comments

This message does not cause the cursor to be located at the defined position. If cursor placement is required then use SCCVW_SETSELECTION.

Platforms

Windows

4.12.1 SCCVWDISPLAYPOSITION41 Structure

This structure is passed by the OEM through the SCCVW_DISPLAYPOSITION message.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct
   {
   VTDWORD      dwSize;
   VTDWORD      dwDisplayOptions
   SCCVWPOS     sPos;
   } SCCVWDISPLAYPOSITION41;
```

Parameters

- dwSize: Must be set by the OEM to sizeof (SCCVWDISPLAYPOSITION41).
- dwDisplayOptions: Flags that allow control over the location of the position in the view. If none of the following are set, then the default action is to display the position with as little scrolling of the screen as possible. Otherwise, one of the following:
  - SCCVW_DISPLAYNEARTOP: The position will be displayed near the top of the view. Formatting considerations will determine exact location.
  - SCCVW_DISPLAYNEARMIDDLE: The position will be displayed near the middle of the view. Formatting considerations will determine exact location.
  - SCCVW_DISPLAYNEARBOTTOM: The position will be displayed near the bottom of the view. Formatting considerations will determine exact location.
- sPos: Identifies the position to go to.

4.13 SCCVW_DODIALOG

* This message invokes one of the Viewer's built in dialogs.

Parameters

- wParam: Should be set to 0.
- lParam: Id of the dialog the developer wishes to invoke. Should be one of the following:
  - SCCID_DISPLAYOPTIONSDIALOG: A dialog that lets the user set a number of aspects of how files are viewed.
  - SCCID_PRINTOPTIONSDIALOG: A dialog that lets the user set a number of aspects of how files are printed.
  - SCCID_CLIPBOARDOPTIONSDIALOG: A dialog that lets the user set a number of aspects of how files are copied to the clipboard.

Return Value

One of the following values:

- SCCVWERR_OK: The dialog was successful.
- SCCVWERR_CANCEL: The user canceled the dialog.
- SCCVWERR_BADPARAM: The ID in lParam was invalid.

Platform

Windows

4.14 SCCVW_DOHELP

* * This message is sent to the developer whenever a help button has been pressed in one of the Viewer's dialogs.

Parameters

- wParam: The handle of the view window that is sending this message.
- lParam: ID of the dialog in which the help button was pressed, it can be one of the following:
  - SCCID_DISPLAYOPTIONSDIALOG
  - SCCID_DISPLAYMOREDIALOG
  - SCCID_PRINTOPTIONSDIALOG
  - SCCID_PRINTMOREDIALOG
  - SCCID_CLIPBOARDOPTIONSDIALOG
  - SCCID_CLIPBOARDMOREDIALOG
  - SCCID_PRINTDIALOG
  - SCCID_PRINTSETUPDIALOG
  - SCCID_SELECTFONTDIALOG

Platform

Windows

4.15 SCCVW_DOMENUITEM

* This message should be called when the developer's application detects user interaction with a menu having an ID greater than the value set by SCCVW_SETMENUMAX or greater than SCCVW_DEFAULTMENUMAX if SCCVW_SETMENUMAX was not called.

Parameters

- wParam: Menu ID received in WM_COMMAND.
- lParam: Not used.

Platform

Windows

4.16 SCCVW_DRAWPAGE

* This message draws a page of the currently viewed file to a rectangle and device of the OEM's choice.

Parameters

- wParam: Should be set to 0.
- lParam: Pointer to a structure of the type SCCVWDRAWPAGE40 or SCCVWDRAWPAGE41.

Return Value

One of the following values:

- SCCVWERR_OK: Draw page was successful.
- SCCVWERR_LASTPAGE: The page just drawn is the last page in the file.
- SCCVWERR_NOPAGE: The requested page does not exist or is not yet available.
- SCCVWERR_NOFILE: No file is being viewed.
- SCCVWERR_NOINIT: SCCVW_INITDRAWPAGE has not been called.
- SCCVWERR_ALLOCFAILED: Allocation failed.
- SCCVWERR_FEATURENOTAVAIL: Draw page is not supported by this display engine.
- SCCVWERR_UNKNOWN: Some other error has occurred.

Platforms

Windows

4.16.1 SCCVWDRAWPAGE41 Structure

This structure is passed by the developer through the SCCVW_DRAWPAGE message.

Structure

A C data structure defined in sccvw_w.h, as follows:

Windows Structure

```
typedef struct SCCVWDRAWPAGE41tag
   {
   VTDWORD     dwSize;
   VTDWORD     dwPageToDraw;
   VTDWORD     dwReserved;
   VTDWORD     dwFlags;
   VTLONG      lUnitsPerInch;
   VTLONG      lFormatWidth;
   VTLONG      lFormatHeight;
   VTLONG      lTop;
   VTLONG      lLeft;
   VTLONG      lBottom;
   VTLONG      lRight;
   VTLONG      lResultTop;
   VTLONG      lResultLeft;
   VTLONG      lResultBottom;
   VTLONG      lResultRight;
   HDC         hOutputDC;
   HDC         hFormatDC;
```

```
HPALETTE    hPalette;
} SCCVWDRAWPAGE41,  * PSCCVWDRAWPAGE41;
```

Parameters

- dwSize: Must be set by the developer to sizeof (SCCVWDRAWPAGE41).
- dwPageToDraw: The page of the file that should be drawn. Page numbers start at 0. In order to draw a page, all pages before it must have been drawn, but once a page has been drawn it may be drawn again at any time. For example, the following sequence is OK:

  ```
  0 - 1 - 2 - 3 - 4 - 1 - 3 - 2 - 5 - 0 - …
  ```

  But in this sequence, the request for page 5 will return SCCVWERR_NOPAGE:

  ```
  0 - 1 - 2 - 5
  ```

- dwReserved: Reserved for future use, must be set to 0.
- dwFlags: Flags indicating the options for draw page. One of the following:
  - SCCVW_DPFLAG_RETURNPALETTE: The hPalette member will be filled by the view window.
  - SCCVW_DPFLAG_NOTMETAFILE: hOutputDC will be assumed not to be 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 creating a metafile.
- lUnitsPerInch: Number of units per inch that lFormatWidth and lFormatHeight are in. For instance, if lFormatHeight and lFormatWidth are in twips, lUnitsPerInch should be 1440.
- 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.
- lTop, lLeft, lBottom, lRight: Rectangle in device units that is equivalent to lFormatHeight and lFormatWidth. This rectangle must be in device units. On the Windows platform, SCCVW_DRAWPAGE uses the Windows mapping modes to draw into this rectangle.
- lResultTop, lResultLeft, lResultBottom, lResultRight: Rectangle in device units that was actually filled with output. These elements are filled by the SCCVW_DRAWPAGE call, they may be different than the requested rectangle for the following reasons:
  - For Documents: lResultBottom may be less than lBottom because lines and table rows are never split when printing.
  - For Spreadsheets and Databases: lResultBottom and lResultLeft may be less because columns and rows are never split when printing.
  - For Bitmaps and Vectors: The result rectangle may be different because the viewer may be maintaining the aspect ratio of the image.
- hOutputDC: The device context or the id pointing to the NSView to draw into.

- hFormatDC: The device or information context or information contet or the id pointing to the NSView used to wrap text and retrieve other formatting information.

  Note: hOutputDC and hFormatDC may not be the same DC.

- hPalette: Palette that should be used when rendering the page. If the SCCVW_DPFLAG_RETURNPALETTE flag is not set, this member will always be NULL. If this flag is set, it may be a handle to a palette or it may still be NULL if no palette is required. It is the developer's responsibility to DeleteObject the palette, if one is returned.

Comments

How the SCCVW_DRAWPAGE message interprets the contents of the SCCVWDRAWPAGE41 structure on a Windows system depends somewhat on the nature of the hOutputDC. If hOutputDC is NOT a metafile DC or the SCCVW_DPFLAG_NOTMETAFILE flag is set, draw page sets the DC to MM_ANIOSTROPIC mode, sets the window extents/origin and sets the viewport extents/origin so the page will be drawn into the rectangle defined by lLeft/lTop/lRight/lBottom. If hOutputDC is a metafile DC, draw page still sets the mode to MM_ANISOTROPIC and sets the window extents/origin, but does not set the viewport. This produces a scalable metafile allowing the developer to later play back the metafile to any rectangle by setting the viewport extents and origin before playing the metafile.

4.17 SCCVW_ENABLEAPP

* *This message is sent to the developer 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 which could affect the view window.

Parameters

- wParam: The handle of the view window that is sending this message.
- lParam: One of the following:
  - FALSE: The application should be disabled
  - TRUE: The application should be enabled.

Return Value

Must be SCCVWERR_MESSAGEHANDLED, which tells the view window the message has been handled. Any other return value will cause the view window to try the default disabling method for that platform. For example, under Windows if the developer does not process this message and return SCCVWERR_MESSAGEHANDLED, the view window will use EnableWindow to disable/enable its parent. If this default behavior is acceptable, simply return 0.

Platform

Windows

4.18 SCCVW_FILECHANGE

* * This message is sent to the developer whenever the file being viewed is changed, such as on a SCCVW_VIEWFILE or SCCVW_CLOSEFILE message. A reasonable response to this message might be to call SCCVW_GETFILEINFO to get information about the new file to display within your UI.

Parameters

- wParam: The handle of the view window that is sending this message
- lParam: Flags indicating the change that is occurring. One or more of the following values OR-ed together:
  - SCCVW_FC_CLOSED: File was closed.
  - SCCVW_FC_OPENED: File was opened.
  - SCCVW_FC_INOPEN: When OR-ed with SCCVW_FC_CLOSED, indicates that the close is in response to a SCCVW_VIEWFILE message and that another SCCVW_FILECHANGE notification can be expected. This flag should be used to avoid any flashing within your UI.

Platform

Windows

4.19 SCCVW_FINDANNOTATION

* This message allows the parent to find an annotation and retrieve information associated with the annotation. A variety of options are provided for locating an annotation.

Parameters

- wParam: Should be set to 0.
- lParam: Pointer to a structure of the type SCCVWFINDANNOTATION41

Return Value

One of the following values:

- SCCVWERR_NONEFOUND: No annotation that meets the mask or absolute criteria was found.
- SCCVWERR_OK: An annotation that meets the mask or absolute criteria was found and the structure passed through lParam is updated.

Platform

Windows

4.19.1 SCCVWFINDANNOTATION41 Structure

This structure is passed by the OEM through the SCCVW_FINDANNOTATION message.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct
   {
   VTDWORD      dwSize;
   VTDWORD      dwFindOptions;
   VTDWORD      dwUserTest;
   SCCVWPOS     sSearchPos;
   VTDWORD      dwUserResult;
   SCCVWPOS     sStartPos;
   SCCVWPOS     sEndPos;
   VTDWORD      dwData;
   } SCCVWFINDANNOTATION41;
```

Parameters

- dwSize: Must be set by the OEM to sizeof (SCCVWFINDANNOTATION41).
- dwFindOptions: Flags indicating the starting position and the meaning of the value in dwUserTest. One of the following values:
  - SCCVW_FINDFIRST: Find the first or only annotation in the document that meets the mask or absolute criteria.
  - SCCVW_FINDLAST: Find the last or only annotation in the document that meets the mask or absolute criteria.
  - SCCVW_FINDFIRSTFROMPOS: Find the next annotation in the document that appears after the position identified by the sSearchPos value and that meets the mask or absolute criteria.
  - SCCVW_FINDPREV: Find the first annotation that appears before the position identified by the sSearchPos and meets the mask or absolute criteria.
  - SCCVW_FINDNEXT: Find the first annotation that appears after the position identified by the sSearchPos and meets the mask or absolute criteria.

  OR-ed with one of the following values:

  - SCCVW_MASK: dwUserTest represents a mask to be applied to the dwUser values of the annotations.
  - SCCVW_ABSOLUTE: dwUserTest represents an absolute dwUser value as set by the OEM in a SCCVW_ADDANNOTATION message.

    If dwFindOptions contains SCCVW_ABSOLUTE, the set of annotations that can be moved to is limited to those annotations where dwUser == dwUserTest. If dwFindOptions contains SCCVW_MASK, the set of annotations that can be moved to is limited to those annotations where dwUser & dwUserTest == dwUserTest. For example, if five annotations exist with the following dwUser values:

    * Annotation 1 (dwUser = 0x001)
    * Annotation 2 (dwUser = 0x0010)
    * Annotation 3 (dwUser = 0x0011)
    * Annotation 4 (dwUser = 0x0100)
    * Annotation 5 (dwUser = 0x1000)

then the following are true:

* a dwUserTest of 0x0000 will use all the annotations
* a dwUserTest of 0x0001 will use annotations 1 and 3
* a dwUserTest of 0x0011 will use annotation 3 only
* a dwUserTest of 0x1001 will not use any annotations.

SCCVW_MASK can be used to move around within a particular set of annotations, where each annotation has a unique dwUser but they all share some bits in common.

- dwUserTest: Used to indicate the class of annotations to be included in the search based on the rules outlined in the preceding list.
- sSearchPos: Indicates the position to begin searching from when SCCVW_FINDNEXT or SCCVW_FINDPREV are set in the dwFindOptions.
- dwUserResult: If the annotation is found, returns the value of dwUser as set by the OEM in a SCCVW_ADDANNOTATION message.
- sStartPos: If the annotation is found:
  - If the annotation type is SCCVW_INSERTTEXT or SCCVW_INSERTPARABREAK, sStartPos will indicate the position of the beginning of the inserted text.
  - For all other annotation types, sStartPos will be as set by the OEM in the SCCVW_ADDANNOTATION message.
- sEndPos: If the annotation is found:
  - If the annotation type is SCCVW_INSERTTEXT or SCCVW_INSERTPARABREAK, sEndPos will indicate the position of the insertion point of the inserted text
  - For all other annotation types, sEndPos will be as set by the OEM in the SCCVW_ADDANNOTATION message
- dwData: If the annotation is found, returns the value of dwData as set by the OEM in a SCCVW_ADDANNOTATION message.

4.20 SCCVW_FINDPOSITION

* This message allows the parent to find a position in terms of a SCCVWPOS with numerous options. The position returned can then be used by any message which requires an SCCVWPOS as input. All of the find options are relative to the currently viewed section.

Parameters

- wParam: Should be set to 0.
- lParam: Pointer to a structure of the type SCCVWFINDPOSITION41.

Return Value

One of the following values:

- SCCVWERR_NONEFOUND: No position that meets the find criteria was found.
- SCCVWERR_OK: A position that meets the find criteria was found and the structure passed through lParam is updated.

Platforms

Windows

4.20.1 SCCVWFINDPOSITION41 Structure

This structure is passed by the OEM through the SCCVW_FINDPOSITION message. Many of the find options are only supported for word processing file types.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct
{
VTDWORD      dwSize;
VTDWORD      dwFindOptions;
SCCVWPOS     sSearchPos;
SCCVWPOS     sResultPos;
} SCCVWFINDPOSITION41;
```

Parameters

- dwSize: Must be set by the OEM to sizeof (SCCVWFINDPOSITION41).
- dwFindOptions: Flags indicating the starting position and the meaning of the value in dwUserIn. Refer to the following table. Note: These options support the currently viewed section. Sections are used to delimit multiple-sheet spreadsheets, multi-page presentation files, and multi-page bitmap graphics.
  - SCCVW_FINDFIRSTPOS: Find the first position in the current section.
  - SCCVW_FINDLASTPOS: Find the last position in the current section.
  - SCCVW_FINDPREVPOS: Find the position in the document that appears before the position identified by the sSearchPos.
  - SCCVW_FINDNEXTPOS: Find the position in the document that appears after the position identified by the sSearchPos.
  - SCCVW_FINDPREVSECTION: Find the previous section in a document.
  - SCCVW_FINDNEXTSECTION: Find the next section in a document.
  - SCCVW_FINDANCHORSELECTPOS: Find the position in the document that identifies the start of selection.
  - SCCVW_FINDNEXTSELECTPOS: Archive files only, find the position in the document that identifies the next selected node when there are multiple selections.
  - SCCVW_FINDPREVSELECTPOS: Archive files only, find the position in the document that identifies the previous selected node when there are multiple selections.
  - SCCVW_FINDENDSELECTPOS: Find the position in the document that identifies the end of selection.
  - SCCVW_FINDSTARTDISPLAYPOS: For word processing file types, find the position in the document that identifies the top/left most text position in the currently displayed area. For presentation file types, return the viewer position of the beginning of the currently displayed slide.

  - SCCVW_FINDENDDISPLAYPOS: For word processing file types, find the position in the document that identifies the bottom/right most text position in the currently displayed area. For presentation file types, return the viewer position of the end of the currently displayed slide.
  - SCCVW_FINDSTARTLINEPOS: Find the position in the document that identifies the start of the line which contains sSearchPos.
  - SCCVW_FINDENDLINEPOS: Find the position in the document that identifies the end of the line which contains sSearchPos.
  - SCCVW_FINDPREVLINEPOS: Find the position in the document that identifies the start of the line previous to the line which contains sSearchPos.
  - SCCVW_FINDNEXTLINEPOS: Find the position in the document that identifies the start of the line following the line which contains sSearchPos.
  - SCCVW_FINDPREVWORDPOS: Find the position in the document that identifies the start of the word previous to sSearchPos.
  - SCCVW_FINDNEXTWORDPOS: Find the position in the document that identifies the start of the word which follows sSearchPos.
- sSearchPos: Identifies the position to begin searching for specific dwFindOptions as outlined in the preceding list.
- sResultPos: Returns the resulting position if it was found.

Comments

Many of the find options are only supported for word processing file types. These include SCCVW_STARTDISPLAYPOS, SCCVW_ENDDISPLAYPOS, SCCVW_FINDPREVLINEPOS, SCCVW_NEXTLINEPOS, SCCVW_PREVWORDPOS, and SCCVW_NEXTWORDPOS.

4.21 SCCVW_FREEFILEINFO

Deallocates the file name buffer allocated during the SCCVW_GETFILEINFO message.

Parameters

- wParam: Should be set to 0.
- lParam: A pointer to a SCCVWFILEINFO84 structure which was filled by previously calling SCCVW_GETFILEINFO.

Return Value

SCCVWERR_OK: call was successful

Platform

Windows

4.22 SCCVW_GETANNOTATIONDATA

* This message is provided to allow the developer to retrieve extended data associated with annotations that are internally created by the Viewer. The technology currently creates annotations to support hyperlinks and bookmarks defined in the original file.

When the technology adds hyperlink annotations, the dwUser member is set to SCCVW_USERHYPERTAG OR-ed with a zero-based count of the current hyperlink. When the technology adds bookmark annotations, the dwUser member is set to SCCVW_USERBOOKMARKTAG OR-ed with a zero-based count of the current bookmark. This allows OEMs to use all of the related annotation messages on these internal annotations. Extended data is available for internal annotations and can be retrieved using this message.

Parameters

- wParam: Should be set to 0.
- lParam: A pointer to a structure of the type SCCVWGETANNOTATIONDATA.

Return Value

One of the following values:

- SCCVWERR_OK: The annotation data was retrieved successful.
- SCCVWERR_BADPARAM: One of the values in the SCCVWGETANNOTATIONDATA structure was invalid.

Platform

Windows

4.22.1 SCCVWGETANNOTATIONDATA Structure

This structure is passed by the OEM through the SCCVW_GETANNOTATIONDATA message.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct
   {
   VTDWORD   dwSize;
   VTDWORD   dwUser;
   VTDWORD   dwData;
   VTDWORD   dwResultDataSize;
   VTVOID    pData;
   VTDWORD   dwDataType;
   } SCCVWGETANNOTATIONDATA;
```

Parameters

- dwSize: Must be set by the OEM to sizeof (SCCVWGETANNOTATIONDATA).
- dwUser: Indicates the annotation for which extended data is being retrieved. This value should be the absolute value of the dwUser member that was used when initially adding the annotation.
- dwData: Indicates the value of the value of the dwData member that was used when initially adding the annotation.
- dwResultDataSize: On input, indicates the size of the data pointed to by pData. After returning, indicates the size of the extended data associated with this

annotation. This element will be updated, even if pData is not of sufficient size to retrieve the data.

- pData: A pointer to data where the extended data should be copied. If the size of the data pointed to by pData is not sufficient to retrieve the extended data then no data will be copied.
- dwDataType: Set by the technology, indicates the format of the data retrieved. One of the following:
  - ADTYPE_URL: pData points to a NULL-terminated Internet Uniform Resource Locator. This data type can be returned for annotations created from hyperlinks in formats which allow referencing URLs.
  - ADTYPE_BOOKMARK: pData points to a NULL-terminated string that identifies the bookmark name. This can be returned for annotations created from bookmarks.
  - ADTYPE_NODATA: This data type can be returned for annotations that do not have any extended data.

4.23 SCCVW_GETCLIPINFO

* This message returns flags that specify the readiness of the view window to manipulate clipboard information.

Parameters

- wParam: Should be set to 0.
- lParam: Should be set to 0.

Return Value

Zero or more of the following flags OR-ed together:

- SCCVW_CANCOPYTOCLIP: Information is selected and can be copied to the clipboard.
- SCCVW_CANCUTTOCLIP: Information is selected and can be cut to the clipboard.
- SCCVW_CANPASTEFROMCLIP: Information on the clipboard is available for pasting.

Platform

Windows

4.24 SCCVW_GETCURRENTPAGENUMBER

For this message, both wParam and lParam are ignored (but should be set to zero anyway). When the viewer receives this message, it returns either a number or the value 0 to indicate that a current page number isn't 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.

Parameters

- wParam: Should be set to 0.

- lParam: Should be set to 0.

Return Value

Zero or the current page number

4.25 SCCVW_GETDISPLAYINFO

* Returns information about the current display engine, if any.

Parameters

- wParam: Should be set to 0.
- lParam: A pointer to a SCCVWDISPLAYINFO40 or SCCVWDISPLAYINFO80 structure to be filled.

Return Value

One of the following values:

- SCCVWERR_OK: pFileInfo was filled successful.
- SCCVWERR_BADPARAM: Structure size was incorrect or lParam was null.

Platform

Windows

4.25.1 SCCVWDISPLAYINFO40 and SCCVWDISPLAYINFO80 Structures

These structures are passed by the developer to the SCCVW_GETDISPLAYINFO message handler. Note that while both of these structures can accept Unicode, only the SCCVWDISPLAYINFO80 structure is truly intended to accept Unicode.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct
{
VTDWORD      dwSize;
VTTCHAR      szName[MAX_DISPLAYINFO_SZNAME_SIZE];
VTDWORD      hMenu;
VTDWORD      dwFunctions;
VTDWORD      dwType
} SCCVWDISPLAYINFO40;

typedef struct
{
VTDWORD      dwSize;
VTWORD       szName[MAX_DISPLAYINFO_SZNAME_SIZE];
VTDWORD      hMenu;
VTDWORD      dwFunctions;
VTDWORD      dwType
} SCCVWDISPLAYINFO80;
```

Parameters

- dwSize: Must be set by the developer to sizeof(SCCVWDISPLAYINFO40) or sizeof(SCCVWDISPLAYINFO80).
- szName: String used when a human readable form of the display engine is needed. For example Bitmap, Hex, Document, Archive, Spreadsheet, etc.
- hMenu: Handle to a popup menu specific to the display engine, or NULL if the display engine has no popup menu. Under Windows, hMenu should be cast into the type HMENU.
- dwFunctions: Reserved by Oracle.
- dwType: One of the following:
  - 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 graphics display engine
  - SCCVWTYPE_HTML - HTML document display engine
  - SCCVWTYPE_EMAIL - email display engine

Comments

The popup menus available for each display engine allow the developer to place powerful, file-type-specific options/events in the hand of the user without designing his/her own menus and without dealing with the API for setting options/events. The basic procedure for doing this is as follows.

When you get the SCCVW_DISPLAYCHANGE message sent by the view window, call SCCVW_GETDISPLAYINFO. If hMenu is not 0, remove any previous popup you added and use the szName and hMenu parameters of the SCCVWDISPLAYINFO40 or SCCVWDISPLAYINFO80 structure to add the display engine's popup menu to your menu bar. If hMenu is 0, just remove any previous menu.

4.26 SCCVW_GETDRAWPAGEINFO

* This message will get page information about the requested page. It is to be used within a SCCVW_INITDRAWPAGE and a SCCVW_DEINITDRAWPAGE before the SCCVW_DRAWPAGE to get the best page size information for that page.

Parameters

- wParam: Should be set to 0.
- lParam: Pointer to a structure of the type SCCVWGETDRAWPAGEINFO.

Return Value

One of the following values:

- SCCVWERR_OK: GETDRAWPAGEINFO was successful.
- SCCVWERR_NONEFOUND: There is no page information available in this file.
- SCCVWERR_LASTPAGE: The page just drawn is the last page in the file.
- SCCVWERR_NOPAGE: The requested page does not exist or is not yet available.
- SCCVWERR_NOFILE: No file is being viewed.
- SCCVWERR_NOINIT: SCCVW_INITDRAWPAGE has not been called
- SCCVW_ALLOCFAILED: Allocation failed.
- SCCVW_FEATURENOTAVAIL: Draw page is not supported by this display engine.
- SCCVWERR_UNKNOWN: Some other error has occurred.

Platform

Windows

4.26.1 SCCVWDRAWPAGEINFO Structure

This structure is passed by the developer through the SCCVW_GETDRAWPAGEINFO message.

Structure

A C data structure defined in sccvw_w.h (Windows), as follows:

Windows Structure

```
typedef struct SCCVWDRAWPAGEINFOtag
   {
   VTDWORD      dwSize;
   VTDWORD      dwPageToDraw;
   VTDWORD      dwReserved;
   VTDWORD      dwFlags;
   VTLONG       lUnitsPerInch;
   VTLONG       lFormatWidth;
   VTLONG       lFormatHeight;
   VTLONG       lTableWidth;
   HDC          hOutputDC;
   HDC          hFormatDC;
   } SCCVWDRAWPAGEINFO,  * PSCCVWDRAWPAGEINFO;
```

Parameters

- dwSize: Must be set by the developer to sizeof (SCCVWDRAWPAGEINFO).
- dwPageToDraw: The page of the file about which information is needed. Page numbers start at 0. In order to get this information, all pages before it must have been drawn through SCCVW_DRAWPAGE. If the previous page was not already drawn, this message will return SCCVWERR_NOPAGE.
- dwReserved: Reserved for future use, must be set to 0.
- dwFlags: (Default) 0

By default, this returns the page height and page width using the lUnitsPerInch value.

(optional) SCCVW_DPFLAG_DETERMINEIMAGEPIXEL: This returns the size in Pixels (recommended only for Image Types--if the user wants to get the pixel dimensions instead of page height and page width using the lUnitsPerInch value) The Pixel dimensions are filled in lFormatWidth and lFormatHeight.

- lUnitsPerInch: Number of units per inch that lFormatWidth and lFormatHeight are in. For instance, if lFormatHeight and lFormatWidth are in twips (as on Windows systems), lUnitsPerInch should be 1440.
- lFormatHeight: This will get filled with the page height using the lUnitsPerInch value. If there is no page information in the file, the return code will be SCCVWERR_NONEFOUND and this value will be unchanged.
- lFormatWidth: This will get filled with the page width using the lUnitsPerInch value. If there is no page information in the file, the return code will be SCCVWERR_NONEFOUND and this value will be unchanged.
- lTableWidth: This will get filled with the width of the widest table in a word processing document, using the lUnitsPerInch value. If there are no tables, or the document is not a Word Processing document, this value will be set to 0. If there is no page information in the file, the return code will be SCCVWERR_NONEFOUND and this value will be unchanged.
- hOutputDC: Device context to draw into.
- hFormatDC: Device context (or information context) used to wrap text and retrieve other formatting information.

  Note: hOutputDC and hFormatDC may not be the same DC.

4.27 SCCVW_GETFILEACCESSDATA

** This message is sent from the view window to the developer when the technology encounters a file that requires additional information to access its contents.

Parameters

- wParam: ID of information requested.
- lParam: Pointer to a SCCVWFILEACCESSDATA structure.

```
typedef struct {
VTDWORD   dwSize;       /* size of this structure */
VTBOOL    bProcessed;   /* flag if this request was processed */
VTDWORD   dwRequestId;  /* id of information requested */
VTWORD    wFIId;        /* FI id of reference file */
VTDWORD   dwSpecType;   /* file spec type */
VTVOID   *pSpec;            /* pointer to a file spec */
VTSYSVAL  pReturnData;      /* pointer to structure to place requested */
                            /*  information in to return to OIT */
VTDWORD   dwReturnDataSize; /* size of the return data structure */
VTDWORD   dwAttemptNumber;  /* The number of times the callback has */
                            /* already been called for the currently */
                            /* requested item of information */
} SCCVWFILEACCESSDATA, * PSCCVWFILEACCESSDATA;
```

- dwRequestID: One of the following values:
  - OIT_FILEACCESS_PASSWORD – Requesting the password of the file

  - OIT_FILEACCESS_NOTESID – Requesting the notes ID file location
- pReturnData: For OIT_FILEACCESS_PASSWORD and OIT_FILEACCESS_NOTESID, the buffer is an array of WORD characters.
- dwReturnDataSize: Size of the pReturnData buffer.

Return Value

- SCCERR_OK: Tells Oracle Outside In that the requested information is provided.
- SCCERR_CANCEL: Tells Oracle Outside In that the requested information is not available.

This message will be repeatedly sent if the information provided is not valid (such as the wrong password). It is the responsibility of the application to provide the correct information or return SCCERR_CANCEL.

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.

4.28 SCCVW_GETFILEINFO

* Returns information about the file currently being viewed.

Parameters

- wParam: Should be set to 0.
- lParam: A pointer to a SCCVWFILEINFO40, SCCVWFILEINFO80 or SCCVWFILEINFO84 structure to be filled.

  The new structure is:

```
typedef struct
{
  VTDWORD       dwSize;
  VTWORD        szDisplayName[SCCVW_DISPLAYNAMEMAX];
  VTDWORD       dwFileId;
  VTWORD        szFileIdName[SCCVW_FILEIDNAMEMAX];
  VTWORD       *szFileName;
} SCCVWFILEINFO84;
```

- dwSize: Must be set by the developer to sizeof(SCCVWFILEINFO40), sizeof(SCCVWFILEINFO80) or sizeof(SCCVWFILEINFO84).

  When using the SCCVWFILEINFO84 structure, the viewer will allocate the buffer used by szFileName to avoid the buffer size limitation of the other versions of the structure. When the caller has finished using szFileName, the buffer must be released using the SCCVW_FREEFILEINFO message. For example:

```
HWND hViewWnd = CreateWindow(…);
SCCVWFILEINFO84 fileInfo;
fileInfo.dwSize = sizeof(SCCVWFILEINFO84);
SendMessage(hViewWnd, SCCVW_GETFILEINFO, 0, (LPARAM)&fileInfo);
…
SendMessage(hViewWnd, SCCVW_FREEFILEINFO, 0, (LPARAM)&fileInfo);
```

Return Value

One of the following values:

- SCCVWERR_OK: pFileInfo was filled successfully
- SCCVWERR_NOFILE: The viewer is not currently viewing a file
- SCCVWERR_BADPARAM: Structure size was incorrect

Platform

Windows

4.28.1 SCCVWFILEINFO40 and SCCVWFILEINFO80 Structures

These structures can be passed by the developer to the SCCVW_GETFILEINFO message. Note that while both of these structures can accept Unicode, only the SCCVWFILEINFO80 structure is truly intended to accept Unicode.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct
   {
   VTDWORD   dwSize;
   VTTCHAR   szDisplayName[SCCVW_DISPLAYNAMEMAX];
   VTDWORD   dwFileId;
   VTTCHAR   szFileIdName[SCCVW_FILEIDNAMEMAX];
   VTTCHAR   szFileName[SCCVW_FILENAMEMAX];
   } SCCVWFILEINFO40;

typedef struct
   {
   VTDWORD   dwSize;
   VTWORD    szDisplayName[SCCVW_DISPLAYNAMEMAX];
   VTDWORD   dwFileId;
   VTWORD    szFileIdName[SCCVW_FILEIDNAMEMAX];
   VTWORD    szFileName[SCCVW_FILENAMEMAX];
   } SCCVWFILEINFO80;
```

Parameters

- dwSize: Must be set by the developer to sizeof(SCCVWFILEINFO40) or sizeof(SCCVWFILEINFO80).
- szDisplayName: String used when a human readable form of the file name is needed. This string is either set specifically by the developer in the structure or it is generated by the viewer based on the real name of the file.
- dwFileId: The ID number associated with this type of file. The numbers are defined in the include file sccfi.
- szFileIdName: A human readable representation of the FI ID number, for example WordPerfect 5.0.
- szFileName: A path to the actual file being viewed.

4.29 SCCVW_GETIDEALWINDOWSIZE

* This message allows the developer to determine the ideal window size to use for the loaded document. This is especially useful when viewing graphic file formats.

Parameters

- wParam: The handle of the view window sending this message.
- lParam: Pointer to a structure of the type SCCVWIDEALSIZE50.

Return Value

One of the following values:

- SCCVWERR_OK: The ideal window size was successfully calculated.
- SCCVWERR_BADPARAM: One of the values in the SCCVWIDEALSIZE50 structure was invalid.
- SCCVWERR_NONEFOUND: This file does not have an ideal size. In this case the Result values are set to the Maximum values.

Comments

Since this message cannot determine the ideal window size until the document is already loaded, it is recommended that the initial view window be hidden when created, and shown after resizing using the results of this message. Also, if the OEM is taking over control of the scroll bars, it should be done prior to using this message.

Platform

Windows

4.29.1 SCCVWIDEALSIZE50 Structure

This structure is passed by the OEM through the SCCVW_GETIDEALWINDOWSIZE message.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct SCCVWIDEALSIZE50tag
{
   VTDWORD   dwSize;
   VTDWORD   dwFlags;
   VTDWORD   dwMaxWidth;
   VTDWORD   dwMaxHeight;
   VTDWORD   dwResultWidth;
   VTDWORD   dwResultHeight;
} SCCVWIDEALSIZE50,* PSCCVWIDEALSIZE50;
```

Parameters

- dwSize: Must be set by the OEM to sizeof(SCCVWIDEALSIZE50).
- dwFlags: Set by the OEM to a value of 0. Future support may allow for special flag settings.

- dwMaxWidth: Set by the OEM to the maximum value, in screen coordinates, to which the dwResultWidth should be set.
- dwMaxHeight: Set by the OEM to the maximum value, in screen coordinates, to which the dwResultHeight should be set.
- dwResultWidth: Returned value which indicates the ideal width, in screen coordinates, to use as the window width of the viewer window. This value is calculated based on the dimensions of the document being viewed and takes the existence of scroll bars into account.
- dwResultHeight: Returned value which indicates the ideal height, in screen coordinates, to use as the window height of the viewer window. This value is calculated based on the dimensions of the document being viewed and takes the existence of scroll bars into account.

4.30 SCCVW_GETOPTION

* This message allows the developer to get the current or default value of an option.

Parameters

- wParam: Should be set to 0.
- lParam: A pointer to a SCCVWOPTIONSPEC40 structure.

Return Value

One of the following values:

- SCCVWERR_OK: pData contains the value of the option.
- SCCVWERR_BADPARAM: Structure size was incorrect or the option ID was bad.

Platform

Windows

4.30.1 SCCVWOPTIONSPEC40 Structure

This structure is passed by the developer through the SCCVW_GETOPTION or SCCVW_SETOPTION message.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct
   {
   VTDWORD   dwSize
   VTDWORD   dwId;
   VTDWORD   dwFlags;
   VTVOID    pData;
   } SCCVWOPTIONSPEC40;
```

Parameters

- dwSize: Must be set by the developer to sizeof(SCCVWOPTIONSPEC40).
- dwId: The ID of the option the developer wants to set or get.

- dwFlags: Any of the following OR-ed together:
  - SCCVWOPTION_CURRENT: Set/Get the current value of the option.
  - SCCVWOPTION_DEFAULT: Set/Get the default value of the option.

  Note: These two flags may only be used together when passing the structure to SCCVW_SETOPTION. In SCCVW_GETOPTION they are mutually exclusive.
- pData: A pointer to a data area containing the option's value on a Set or receiving the option's value on a Get. This area must be as large as the data type specified for the option ID the developer is getting or setting.

4.31 SCCVW_GETPROPERTY

This message returns any file property information generated by the input filter.

Parameters

- wParam: 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 previous property generated by the filter.
  - SCCVW_GETPROP_BYID: Retrieves the information (if any) for a property specified in the structure. Values are specified in sccca.h, and range from SCCCA_DOCCOMMENT (1) to SCCCA_APPVERSION (162). User defined properties can not be specified.
  - SCCVW_GETPROP_BYINDEX: Retrieves a property specified by a zero-based index from the property array.
- lParam: Pointer to a structure of the type SCCVWGETPROP.

Return Value

One of the following values:

- SCCERR_OK: Property information was successfully retrieved, or data size members were populated (if NULL pointers are passed for data).
- SCCERR_BADPARAM: lParam is NULL/0 or an invalid value was passed in wParam, or a buffer size of less than 2 was specified in the SCCVWGETPROP structure.
- SCCERR_ALLOCFAILED: An internal allocation failed.
- SCCERR_INDEXOUTOFBOUNDS: Request property does not exist. This can happen with SCCVW_GETPROP_NEXT if the last property was already retrieved, or with SCCVW_GETPROP_PREV if the first property was just retrieved, or with SCCVW_GETPROP_FIRST if the are no properties available for the file, or with the SCCVW_GETPROP_BYID if the specified property ID is not available.
- SCCERR_INSUFFICIENTBUFFER: User allocated buffer was not sufficient to hold property information. Data will not be placed in insufficient buffers, and the initial character will be set to NULL. Because there are two buffers, one may be

filled even if the other is insufficient. The required buffer size is placed in the SCCVWGETPROP structure.

4.31.1 SCCVWGETPROP Structure

This structure is passed by the developer through the SCCVW_GETPROPERTY message.

Structure

A C data structure defined in sccvw.h, as follows:

```
typedef struct SCCVWGETPROPtag
   {
   VTDWORD   dwSize;
   VTDWORD   dwPropId;
   VTLPVOID  pName;
   VTDWORD   dwNameSize;
   VTLPVOID  pData;
   VTDWORD   dwDataSize;
   VTDWORD   dwDataType;
   } SCCVWGETRAWTEXT50;
```

Parameters

- dwSize: [IN]: Must be set by the developer to sizeof(SCCVWGETPROP).
- dwPropId: [IN or OUT]: ID of property name. Must be set by the developer for SCCVW_GETPROP_BYID.
- pName: [IN] pointer to a buffer supplied by the developer to receive property name. May be NULL.
- dwNameSize: [IN/OUT] IN: Size in bytes of buffer pointed to by pName. [OUT]: Bytes copied into pName, or size of buffer necessary to receive property name.
- pData: [IN] pointer to a buffer supplied by the developer to receive property value. May be NULL.
- dwDataSize: [IN/OUT] IN: Size in bytes of buffer pointed to by pData. [OUT]: Bytes copied into pData, or size of buffer necessary to received property value.
- dwDataType: [OUT] Type of data copied into pData. For this version, SCCVW_PROPTYPE_STRING is the only valid type, indicating Unicode string data.

Comments

All strings (Property names, as well as property value for this version) are Unicode. If pName or pData are NULL, the dwNameSize and dwDataSize members will be filled with required buffer size to hold the indicated property.

4.32 SCCVW_GETRAWTEXT

* This message allows the parent to get the raw text buffer which contains the character identified by its actual character count (ACC) from the beginning of the document. The buffer of text returned is of variable size.

Parameters

- wParam: Should be set to 0.

- lParam: Pointer to a structure of the type SCCVWGETRAWTEXT50.

Return Value

One of the following values:

- SCCVWERR_BADPARAM: There was a problem with the parameters passed into this call.
- SCCVWERR_ALLOCFAILED: A memory allocation has failed. The machine has possibly run out of memory.
- SCCVWERR_UNKNOWNFAILURE: There was a failure opening this document and getting the text.
- SCCVWERR_NONEFOUND: The document does not contain the character (the number of characters in the document is less than the ACC requested).
- SCCVWERR_OK: The text buffer containing the requested character was found and the structure has been updated.

Comments

To produce offsets used in SCCVW_ADDANNOTATION, use the SCCVWGETRAWTEXT50 structure as follows:

1. Fill the dwACC member of the structure with a character count which identifies the offset of the character to be annotated.
2. Fill dwSize and dwOptions and call SCCVW_GETRAWTEXT.
3. Locate the start and end character count offsets of the text to annotate and call SCCVW_MAPPOSITION to turn each character count into a SCCVWPOS. The end position of an annotation is NOT inclusive, so the value used for an end position should be the offset of the next character after the last character to be modified by the annotation. If an annotation includes the last character of the document, the end position must be set to the offset value of the last character plus 1.
4. Use the resulting SCCVWPOS values as the sStartPos and sEndPos elements when calling SCCVW_ADDANNOTATION.

   NULL characters are included in the text buffer to act as fillers for text which was in the original file but is not part of the document body (revision deletions and document properties).

   If Unicode output is required, then use the option SCCID_SYSTEMFLAGS to enable Unicode raw text output.

   It is the OEM's responsibility to free the hText memory blocks. This allows the OEM to perform cross chunk searches.

> **Note:**
>
> Developers who need just the text of a file (for indexing, for example) should contact an Oracle Account Representative for information about the Oracle Outside In Content Access product. This technology uses the same filters as the Viewer to rapidly read text and other content. Oracle guarantees that the text retrieved from the Content Access product will be identical (character for character) with the text returned by the SCCVW_GETRAWTEXT. However, Content Access can retrieve more data than is available from the raw text buffer, including non-visible text (such as document properties and hidden text), and text from embedded documents.

Platform

Windows

4.32.1 SCCVWGETRAWTEXT50 Structure

This structure is passed by the OEM through the SCCVW_GETRAWTEXT message.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct
   {
   VTDWORD   dwSize;
   VTDWORD   dwOptions;
   VTDWORD   dwACC;
   VTHANDLE  hText;
   VTDWORD   dwCharCount;
   VTDWORD   dwCharSet;
   VTDWORD   dwTextBufSize;
   VTDWORD   dwTextOffset;
   } SCCVWGETRAWTEXT50;
```

Parameters

- dwSize: Must be set by the OEM to sizeof(SCCVWGETRAWTEXT50).
- dwOptions: Flags indicating options for getting the raw text (none defined, set to 0).
- dwACC: Set this to the actual character count, 0 based from the beginning of the document, which defines the raw text buffer being requested.
- hText: Returns the handle to an array of BYTEs representing characters. It is the developer's responsibility to free the hText handle when finished accessing the text.
- dwCharCount: Returns the number of characters in hText.
- dwCharSet: Returns the SO_* (see sodefs) character set of the characters in hText. The output character set is either the default native environment character set or Unicode if the SCCID_SYSTEMFLAGS is set to SCCVW_SYSTEM_UNICODE.
- dwTextBufSize: Returns the size in bytes in hText. This is not necessary the same as the number of characters for DBCS and Unicode character sets.

- dwTextOffset: Returns the number of characters from the beginning of the document to the first character in hText.

4.33 SCCVW_GETTREECOUNT

* This message allows the parent to get the number of nodes in an archive file. A node is defined as either a folder or a file. Node 0 is the root node for the archive, and contains the other nodes.

Parameters

- wParam: Should be set to 0.
- LParam: Pointer to a VTDWORD to be filled in with the number of nodes in the archive.

Return Value

One of the following values:

- SCCVWERR_BADPARAM: There was a problem with the parameters passed into this call. This can include the file being viewed not being an archive file.
- SCCVWERR_OK: The VTDWORD has been filled in with the number of nodes in the archive.

Platform

Windows

4.34 SCCVW_GETTREENODE

* This message allows the parent to get information regarding a node such as the name or size of the node. A node is defined as either a folder or a file. Node 0 is the root node for the archive, and contains the other nodes.

Parameters

- wParam: Should be set to 0.
- lParam: Pointer to a SCCDATREENODE structure.

Return Value

One of the following values:

- SCCVWERR_BADPARAM: There was a problem with the parameters passed to this message. This can include the file being viewed not being an archive file.
- SCCVWERR_OK: The SCCDATREENODE structure has been filled in with information about the node in the archive.

Platform

Windows

Comments

A node can be either a folder, or an archived file. For archives that store path information, the path is broken down into separate nodes. Node zero is always a folder that contains the contents of the archive file.

4.34.1 SCCDATREENODE Structure

This structure is passed by the OEM through the SCCVW_GETTREENODE message.

Structure

A C data structure defined in sccda as follows:

```
typedef struct
   {
   VTDWORD   dwSize;
   VTDWORD   dwNode;
   VTTCHAR   szName[1024];
   VTDWORD   dwFileSize;
   VTDWORD   dwTime;
   VTDWORD   dwFlags;
   VTDWORD   dwCharSet;
   } SCCDATREENODE,  *PSCCDATREENODE;
```

Parameters

- dwSize: Must be set by the OEM to sizeof(SCCDATREENODE).
- dwNode: The number of the node to retrieve information about (zero-based). The first node is 0.
- szName: A buffer to hold the name of the node.
- dwFileSize: Returns the file size (in bytes) of the requested node. Folder nodes have size 0.
- dwTime: Returns the timestamp of the requested node, in MS-DOS time.
- dwFlags: Returns additional information about the node. It can be a combination of the following:
  - SCCDA_TREENODEFLAG_FOLDER: Indicating that the selected node is a folder and not a file.
  - SCCDA_TREENODEFLAG_SELECTED: Indicating that the node is selected.
  - SCCDA_TREENODEFLAG_FOCUS: Indicating that node has focus.
- dwCharSet: Returns the SO_* (see sodefs) character set of the characters in szName. The output character set is either the default native environment character set or Unicode if the SCCID_SYSTEMFLAGS is set to SCCVW_SYSTEM_UNICODE.

4.35 SCCVW_GOTOANNOTATION

* This message allows the parent to move the view/selection from annotation to annotation.

Parameters

- wParam: Flags indicating the starting position and the meaning of the value in lParam. One of the following:
  - SCCVW_GOTOFIRST: Go to the first or only annotation in the document that meets the mask or absolute criteria.
  - SCCVW_GOTOLAST: Go to the last or only annotation in the document that meets the mask or absolute criteria.
  - SCCVW_GOTOPREV: Go to the first annotation that appears before the current caret/selection in the document and meets the mask or absolute criteria.
  - SCCVW_GOTONEXT: Go to the first annotation that appears after the current caret/selection in the document and meets the mask or absolute criteria.

  OR-ed with one of the following:

  - SCCVW_ABSOLUTE: lParam represents an absolute dwUser value as set by the developer in a SCCVW_ADDANNOTATION message.
  - SCCVW_MASK: lParam represents a mask to be applied to the dwUser values of the annotations.
- lParam: If wParam contains SCCVW_ABSOLUTE, the set of annotations that can be moved to is limited to those annotations where dwUser == lParam. If wParam contains SCCVW_MASK, the set of annotations that can be moved to is limited to those annotations where dwUser & lParam == lParam. For example, if five annotations exist with the following dwUser values:
  - Annotation 1 (dwUser = 0x001)
  - Annotation 2 (dwUser = 0x0010)
  - Annotation 3 (dwUser = 0x0011)
  - Annotation 4 (dwUser = 0x0100)
  - Annotation 5 (dwUser = 0x1000)

  ... then the following are true:

  - an lParam of 0x0000 will use all the annotations
  - an lParam of 0x0001 will use annotations 1 and 3
  - an lParam of 0x0011 will use annotation 3 only
  - an lParam of 0x1001 will not use any annotations

  SCCVW_MASK can be used to move around within a particular class of annotation that each has a unique dwUser but share some bits in common.

Return Value

One of the following values:

- SCCVWERR_NONEFOUND: No annotation that meets the mask or absolute criteria was found.
- SCCVWERR_OK: An annotation that meets the mask or absolute criteria was found and is selected
- SCCVWERR_BADPARAM: One of the values in wParam or lParam is bad.

Platform

Windows

4.36 SCCVW_HILITESTYLE

* This message allows the parent to define a formatting style for highlighted text. The style identifier can be used in conjunction with SCCVW_ADDANNOTATION.

Parameters

- wParam: Should be set to 0.
- wCharAttr: Refers to the value set for the wAttr variable used in the SCCVWFONTSPEC structure of the SCCID_DEFAULTPRINTFONT option to determine font attributes.
- lParam: Pointer to a structure of the type SCCVWHILITESTYLE41.

Return Value

One of the following values:

- SCCVWERR_OK: The style definition has been stored.
- SCCVWERR_BADPARAM: One value in SCCVWCOPY41 structure was invalid.

Platform

Windows

4.36.1 SCCVWHILITESTYLE41 Structure

This structure is passed by the OEM through the SCCVW_HILITESTYLE message.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct
   {
   VTDWORD         dwSize;
   VTDWORD         dwStyleId;
   VTDWORD         dwOptions;
   SCCVWCOLORREF   sForeground;
   SCCVWCOLORREF   sBackground;
   VTWORD          wCharAttr;
   } SCCVWHILITESTYLE41;
```

Parameters

- dwSize: Must be set by the OEM to sizeof(SCCVWHILITESTYLE41).
- dwStyleId: A unique ID for this style which can be used when adding highlight annotations with the SCCVW_ADDANNOTATION message.
- dwOptions: Flags which provide highlight options:
  - SCCVW_USEFOREGROUND: Indicates that sForeground defines the foreground text color to apply to highlights with this style.

  - SCCVW_USEBACKGROUND: Indicates that sBackground defines the background color to apply to highlights with this style.
  - SCCVW_USECHARATTR: Indicates that wCharAttr defines attributes to apply to highlights with this style.
- sForeground: Defines the color to be used for the highlight text if SCCVW_USEFOREGROUND was set in dwOptions. Set this value using the SCCVWRGB(red,green,blue) macro. The red, green, and blue are percentages of the color from 0-255 (with 255 being 100%).
- sBackground: Defines the color to be used for the background of the highlight if SCCVW_USEBACKGROUND was set in dwOptions. Set this value using the SCCVWRGB(red,green,blue) macro. The red, green, and blue are percentages of the color from 0-255 (with 255 being 100%).
- wCharAttr: Defines the font attribute to be used for the highlight text if SCCVW_USECHARATTR was set in dwOptions.

4.36.2 SCCVWHILITESTYLE81 Structure

This structure is passed by the OEM through the SCCVW_HILITESTYLE message.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct SCCANNOHILITESTYLE81tag
   {
   VTDWORD         dwSize;
   VTDWORD         dwStyleId;
   VTDWORD         dwOptions;
   SCCVWCOLORREF   sForeground;
   SCCVWCOLORREF   sBackground;
   VTWORD          wCharAttr;
   VTWORD          wCharAttrMask;
   SCCVWCOLORREF   sUnderline;
}  SCCVWHILITESTYLE81, * PSCCVWHILITESTYLE81;
```

The behavior of most of the fields is identical with their behavior in the SCCVWHILITESTYLE41 data structure. The only differences are as follows:

- The addition of wCharAttrMask. This field uses the same flags as the wCharAttr field, and is used to indicate which bits of wCharAttr should be used in defining this style. If a particular attribute is set in wCharAttrMask, then the corresponding value from wCharAttr will be used in the style. In this way, the combination of wCharAttr and wCharAttrMask may be used to turn on or off any of the available character attributes. Any flag set in wCharAttr without the corresponding flag set in wCharAttrMask will be ignored.
- The addition of sUnderline. This value is a color reference that allows a highlight to be specified in which the text is underlined in a color that is different from the text color. For this value to take effect, the SCCVW_USEUNDERLINECOLOR flag must be set in the dwOptions field.

Parameters

- dwSize: Must be set by the OEM to sizeof(SCCVWHILITESTYLE81).

- dwStyleId: A unique ID for this style which can be used when adding highlight annotations with the SCCVW_ADDANNOTATION message.
- dwOptions: Flags which provide highlight options:
  - SCCVW_USEFOREGROUND: Indicates that sForeground defines the foreground text color to apply to highlights with this style.
  - SCCVW_USEBACKGROUND: Indicates that sBackground defines the background color to apply to highlights with this style.
  - SCCVW_USECHARATTR: Indicates that wCharAttr defines attributes to apply to highlights with this style.
  - SCCVW_USETHINUNDERLINE: If the user specifies this flag in the dwOptions field of this data structure, and the style defines an underline character attribute, then the underline will be drawn with a thin line. This overrides the normal case in which the thickness of an underline scales to the size of the font.
  - SCCVW_USEUNDERLINECOLOR: If the user specifies this flag in the dwOptions field of this data structure, and the style defines an underline character attribute, then the sUnderline field will be used to specify the color of underline attributes applied with this style. If this flag is not specified, the sUnderline field will be ignored, and the text color will be used for any underline attributes that may be defined.
- sForeground: Defines the color to be used for the highlight text if SCCVW_USEFOREGROUND was set in dwOptions. Set this value using the SCCVWRGB(red,green,blue) macro. The red, green, and blue are percentages of the color from 0-255 (with 255 being 100%).
- sBackground: Defines the color to be used for the background of the highlight if SCCVW_USEBACKGROUND was set in dwOptions. Set this value using the SCCVWRGB(red,green,blue) macro. The red, green, and blue are percentages of the color from 0-255 (with 255 being 100%).
- wCharAttr: Defines the font attribute to be used for the highlight text if SCCVW_USECHARATTR was set in dwOptions. The following character attribute values are supported:
  - SCCVW_CHARATTR_UNDERLINE
  - SCCVW_CHARATTR_ITALIC
  - SCCVW_CHARATTR_BOLD
  - SCCVW_CHARATTR_STRIKEOUT
  - SCCVW_CHARATTR_CAPS
  - SCCVW_CHARATTR_SUBSCRIPT
  - SCCVW_CHARATTR_SUPERSCRIPT
  - SCCVW_CHARATTR_DUNDERLINE
  - SCCVW_CHARATTR_WORDUNDERLINE
  - SCCVW_CHARATTR_DASHUNDERLINE
  - SCCVW_CHARATTR_DOTUNDERLINE

  Note that though there are multiple underline attributes defined independently of each other, only one at a time may be applied as an annotation style. If more than one underline style is specified in the character attribute flags, it is not guaranteed which underline style will take effect.

4.37 SCCVW_HSCROLL

* This message allows the OEM to cause horizontal scrolling behavior.

Parameters

- wParam: Code describing how to scroll. One of the following values:
  - SCCSB_LINELEFT: Scroll left one line as if the left arrow on the scroll bar had been pressed.
  - SCCSB_LINERIGHT: Scroll right one line as if the right arrow on the scroll bar had been pressed.
  - SCCSB_PAGELEFT: Scroll left one page.
  - SCCSB_PAGERIGHT: Scroll right one page.
  - SCCSB_POSITION: Sets the scroll bar to the position given in lParam.
- lParam: Position used if wParam = SCCSB_POSITION. Its value must be in the range given by the last SCCVW_SETSCROLLRANGE message sent to the OEM.

Return Value

One of the following values:

- SCCVWERR_OK: View window was scrolled.
- SCCVWERR_BADPARAM: One of the values in wParam or lParam is bad.

Platform

Windows,

4.38 SCCVW_IDLE

* This message should be sent repeatedly by the developer to the viewer during the viewing process. This message allows the viewer to blink the cursor and read ahead in the file being viewed. Under Windows, this message is only necessary if the developer has set the SCCVW_SYSTEM_NOTIMER message in the SCCID_SYSTEMFLAGS option.

Parameters

- wParam: Should be set to 0.
- lParam: Should be set to 0.

Platform

Windows

4.39 SCCVW_INFOMESSAGE

* * This message is sent to the developer 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.

Parameters

- wParam: Handle of the view window sending this message.
- lParam: A VTDWORD with 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_UNSUPPORTEDSHADING - An unsupported shading pattern was encountered.
  - SCCVW_INFO_INVALIDHTML - An HTML parse error, as defined by the W3C, was encountered.

Platform

Windows

4.40 SCCVW_INITDRAWPAGE

* This message initializes the page drawing routines. This message must be called before any calls to SCCVW_DRAWPAGE.

Parameters

- wParam: Should be set to 0.
- lParam: Should be set to 0.

Return Value

One of the following values:

- SCCVWERR_OK: Ready of SCCVW_DRAWPAGE messages.
- SCCVWERR_ALLOCFAILED: Not enough memory to do page drawing.

Platform

Windows

4.41 SCCVW_KEYDOWN

* * This message is sent to the developer when the view window receives a key press notification from the operating system. This allows the developer to define their own function for any key.

Parameters

- wParam: Not used, will be set to 0.
- lParam: Operating system specific key identifier.

  For Windows, lParam represents the Windows virtual key code (for example, VK_DOWN).

Return Value

One of the following values:

- SCCVW_DOKEY: If the view window should process the key as usual.
- SCCVW_IGNOREKEY: If the view window should ignore the key.

Platform

Windows

4.42 SCCVW_MAPPOSITION

* This message allows the parent to map between a position and an actual character count.

Parameters

- wParam: Should be set to 0.
- lParam: Pointer to a structure of the type SCCVWMAPPOSITION41.

Return Value

One of the following values:

- SCCVWERR_OK: The mapping was successful
- SCCVWERR_BADPARAM: One of the values in the SCCVWMAPPOSITION41 structure was invalid.
- SCCVWERR_NONEFOUND

Platform

Windows

4.42.1 SCCVWMAPPOSITION41 Structure

This structure is passed by the OEM through the SCCVW_MAPPOSITION message.

Structure

A C data structure defined in sccvw as follows.

```
typedef struct
   {
   VTDWORD      dwSize;
   VTDWORD      dwMapOptions;
   SCCVWPOS     sPos;
   VTDWORD      dwACC;
   } SCCVWMAPPOSITION41;
```

Parameters

- dwSize: Must be set by the OEM to sizeof(SCCVWMAPPOSITION41).
- dwMapOptions: Flags indicating mapping options. One of the following:
  - SCCVW_MAPACCTOPOS: Map the actual character count provided in dwACC into an SCCVWPOS and store the result in sPos. An actual character count is 0 based count of characters from the beginning of the document. See section 6.28.
  - SCCVW_MAPPOSTOACC: Map the SCCVWPOS identified by sPos into an actual character count and store the result in dwACC.
- sPos: Identifies the position to be mapped or to map into according to the dwMapOptions.
- dwACC: Identifies the actual character count to be mapped or to map into according to the dwMapOptions.

4.43 SCCVW_MAPPROB

* * This message informs the OEM that the page of a PDF file being read contains a font or fonts that are more likely to contain unmappable or incorrectly mapped characters.

The following fonts have non-standard encodings and do not have ToUnicode mappings. When used by PDF, they are considered standard:

- STANDARD
- MACROMAN
- MACEXPERT
- WIN_ANSI
- SYMBOL
- ZAPF_DINGBATS
- PDFDOC

Fonts with encodings not on this list will trigger this message if they do not have ToUnicode mappings.

This message will be sent once per page. Multiple non-standard fonts in a single page will not trigger repeated messages, but the use of a font across multiple pages will trigger one message for each such page.

Parameters

- wParam: The handle of the view window that is sending this message.
- lParam: 0

Return Value

None

4.44 SCCVW_MAPTREEPOSITION

* This message allows the parent to map between a position and an archive node numbers. The values can be used with the SCCVW_GETREENODE, SCCVW_FINDPOSITION and SCCVW_SETSELECTION messages.

Parameters

- wParam: Should be set to 0.
- lParam: Pointer to a structure of the type SCCVWMAPTREEPOSITION82.

Return Value

One of the following values:

- SCCVWERR_OK: The mapping was successful
- SCCVWERR_BADPARAM: One of the values in the SCCVWMAPTREEPOSITION82 structure was invalid.
- SCCVWERR_NONEFOUND: No node was found.

Platform

Windows

4.44.1 SCCVWMAPTREEPOSITION82 Structure

This structure is passed by the OEM through the SCCVW_MAPTREEPOSITION message.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct
   {
   VTDWORD   dwSize;
   VTDWORD   dwMapOptions;
   SCCVWPOS  sPos;
   VTDWORD   dwNode;
   VTDWORD   dwRecordNum;
   } SCCVWMAPTREEPOSITION82;
```

Parameters

- dwSize: Must be set by the OEM to sizeof(SCCVWMAPTREEPOSITION82).

- dwMapOptions: Flags indicating mapping options. One of the following:
  - SCCVW_MAPNODETOPOS: Map the tree node value provided in dwNode into an SCCVWPOS and store the result in sPos.
  - SCCVW_MAPPOSTONODE: Map the SCCVWPOS identified by sPos into an actual node value and store the result in dwNode.
- sPos: Identifies the position to be mapped or to map into according to the dwMapOptions. This value corresponds to the values used in SCCVW_GETSELCTION and SCCVWSETSELECTION.
- dwNode: Identifies the actual node value to be mapped or to map into according to the dwMapOptions. This value corresponds to the values used in SCCVW_GETTREENODE.
- dwRecordNum: Identifies the record number of the node in the archive. This value corresponds to the record number values used in the Content Access technology.

4.45 SCCVW_OPTIONCHANGE

* * This message is sent to the developer whenever an option has been changed. It can be used to monitor the current state of options.

Parameters

- wParam: The handle of the view window that is sending this message.
- lParam: Id of the option that has changed.

Platform

Windows

4.46 SCCVW_PRINT

* Causes the viewer to print the file currently being viewed.

Parameters

- wParam: Should be set to 0.
- lParam: Should be set to 0.

Return Value

One of the following values:

- SCCVWERR_OK: The file was printed.
- SCCVWERR_CANCEL: The user pressed Cancel in the dialog.

Comment

This call implements the entire printing process, including setup, option and printer selection dialogs. These dialogs give the user control over the full range of printing functionality available in this technology. If the developer needs more control over the printing process, the SCCVW_PRINTEX message should be used.

Platform

Windows

4.47 SCCVW_PRINTEX

* Causes the viewer to print the file currently being viewed based on information provided by the developer in the SCCVWPRINTEX structure.

Parameters

- wParam: Should be set to 0.
- lParam: A pointer to a SCCVWPRINTEX40 structure.

Return Value

One of the following values:

- SCCVWERR_OK: The user OK'd the operation.
- SCCVWERR_CANCEL: The user canceled the operation.

Platforms

Windows

4.47.1 SCCVWPRINTEX40 Structure

This structure is used by the SCCVW_PRINTEX messages to print a file.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct
   {
   VTDWORD   dwSize;
   VTDWORD   dwFlags;
   HWND      hParentWnd;
   HDC       hPrinterDC;
   VTTCHAR   szPrinter[128];
   VTTCHAR   szPort[128];
   VTTCHAR   szDriver[128];
   VTBOOL    bPrintSelectionOnly;
   VTBOOL    bDoSetupDialog;
   VTBOOL    bDoAbortDialog;
   VTBOOL    bPrintHeader;
   VTBOOL    bStartDocAlreadyDone;
   VTTCHAR   szJobName[40];
   VTDWORD   dwTopMargin;
   VTDWORD   dwBottomMargin;
   VTDWORD   dwLeftMargin;
   VTDWORD   dwRightMargin;
   VTTCHAR   szDefaultFont[32];
   VTWORD    wDefaultFontSize;
   FARPROC   pAbortProc;
   VTBOOL    bCollate;
   VTDWORD   dwCopies;
```

```
VTHANDLE  hDevMode;
} SCCVWPRINTEX40;
```

Parameters

- dwSize: Must be set to sizeof(SCCVWPRINTEX40).
- dwFlags: Any of the following OR-ed together:
  - SCCVW_USEPRINTERDC: Use hPrinterDC as the output device instead of the default printer.
  - SCCVW_USEPRINTERNAME: If SCCVW_USEPRINTERDC is NOT set, use szPrinter, szPort and szDriver as the output device instead of the default printer. If SCCVW_USEPRINTERDC is set, use szPrinter, szPort and szDriver for display purposes in the Abort dialog.
  - SCCVW_USEPRINTSELECTIONONLY: Use the value of bPrintSelectionOnly instead of the value of the SCCID_WHATTOPRINT option.
  - SCCVW_USEJOBNAME: Use the value of szJobName instead of the value of the SCCID_PRINTJOBNAME option.
  - SCCVW_USEMARGINS: Use the values of dwTopMargin through dwRightMargin as the margins instead of the value of the SCCID_DEFAULTPRINTMARGINS option.
  - SCCVW_USEPRINTHEADER: Use the value of bPrintHeader instead of the value of the SCCID_PRINTHEADER option.
  - SCCVW_USEDEFAULTFONT: Use the values of szDefaultFont and wDefaultFontSize instead of the value of the SCCID_DEFAULTPRINTFONT option.
  - SCCVW_USEABORTPROC: Use the values of pAbortProc as the Windows printing abort procedure instead of the default abort procedure.
  - SCCVW_USECOLLATE: Use the values of bCollate instead of the value of the SCCID_PRINTCOLLATE option.
  - SCCVW_USECOPIES: Use the values of dwCopies instead of the value of the SCCID_PRINTCOPIES option.
  - SCCVW_SENDPROGRESS: The view window will send SCCVW_PRINTPROGRESS messages during the print operation. Currently, this flag has not been implemented.
  - SCCVW_USEDEVMODE: Use the value of hDevMode instead of the default device setup information for hPrinterDC
- hParentWnd: Parent window for the setup and abort printing dialogs.
- hPrinterDC: Output DC created by the developer.
- szPrinter: String passed to CreateDC to create the output DC and/or used in the Abort dialog.
- szPort: String passed to CreateDC to create the output DC and/or used in the Abort dialog.
- szDriver: String passed to CreateDC to create the output DC and/or used in the Abort dialog.
- bPrintSelectionOnly: If set to TRUE, only the currently selected area of the file will be printed.
- bDoSetupDialog: If set to TRUE, the dialog shown in the SCCVW_PRINT message will be displayed before printing.

- bDoAbortDialog: If set to TRUE, an abort dialog with page count will be displayed as the file is printing.
- bPrintHeader: If set to TRUE, a header with the display name of the file and the page number will appear on every page.
- bStartDocAlreadyDone: This flag need only be set if you are passing in an hPrinterDC. If set to TRUE, the viewer will assume that a StartDoc has already been done on the DC and will not do a StartDoc or EndDoc itself. This flag can be used to print multiple files in a single print job.
- szJobName: The text passed to StartDoc as the job name. This text generally appears on banner pages for network printing.
- dwTopMargin: Distance in twips from the top edge of the page to the top edge of the area where output should appear.
- dwBottomMargin: Distance in twips from the bottom edge of the page to the bottom edge of the area where output should appear.
- dwLeftMargin: Distance in twips from the left edge of the page to the left edge of the area where output should appear.
- dwRightMargin: Distance in twips from the right edge of the page to the right edge of the area where output should appear.
- szDefaultFont: Name of the font to use when no font is specified in the document (such as ASCII documents).
- wDefaultFontSize: Height in half points of the font to use when no font is specified in the document.
- pAbortProc: A pointer to a valid Windows print abort procedure.
- bCollate: If TRUE the viewer will collate multiple copies.
- dwCopies: The number of copies that should be printed.
- hDevMode: Device mode associated with hPrinterDC.

4.48 SCCVW_PRINTSETUP

* Brings up a platform dependent dialog that allows the user to pick from various platform-printing options.

Parameters

- wParam: Should be set to 0.
- lParam: Should be set to 0.

Return Value

One of the following values:

- SCCVWERR_OK: The user OK-ed the operation.
- SCCVWERR_CANCEL: The user canceled the operation.

Comment

The dialog allows the user to pick from the printers on the system. Additionally, Windows users can change their printer's settings.

Platform

Windows

4.49 SCCVW_RAWTEXTEVENT

* * This message is sent repeatedly to the developer during the initial read of a document. Each message identifies the 0-based offset of the next block of raw text available from the technology. The last RawTextEvent contains a text offset of -1 to indicate it is done extracting text from the file. SCCVW_GETRAWTEXT can be used to retrieve the text buffer.

Parameters

- wParam: Handle of the view window sending this message.
- lParam: Defines the character offset of the next available raw text buffer.

Platform

Windows

4.50 SCCVW_READAHEADDONE

* * Signals that the chunker is done reading the document. This means that the Oracle 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 Oracle Outside In Viewer to access the file again while the user scrolls through the document.

Parameters

- wParam: The handle of the view window that is sending this message.
- lParam: Should be set to 0

Platform

Windows

4.51 SCCVW_SAVEOPTIONS

* Causes the view window to save its current options in the default option set. Generally this happens automatically when the window is destroyed, but if the developer disables this default behavior by setting the SCCVW_SYSTEM_NOOPTIONSSAVE flag in the SCCID_SYSTEMFLAGS option, this message may be used to manually save the current options.

Parameters

- wParam: Should be set to 0.
- lParam: Should be set to 0.

Platform

Windows

4.52 SCCVW_SAVETREENODE

* This message instructs the viewer to save a given tree node from an archive file to a new file.

Parameters

- wParam: Should be set to 0.
- lParam: Should be a pointer to a SCCVWSAVENODE structure.

Return Value

One of the following values:

- SCCVWERR_BADPARAM: There was a problem with the parameters passed into this call. This can include the file being viewed not being an archive file, or an invalid path being specified.
- SCCVWERR_OK: The node was saved successfully.

Platform

Windows

Comments

A node can be either a folder, or a file. If a file is saved, just that file will be saved. If a node is saved, it will save all the contents of that node as well. Saving node zero will save the entire contents of the archive file.

4.52.1 SCCVWSAVENODE Structure

This structure is passed by the OEM through the SCCVW_SAVETREENODE message.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct
   {
   VTDWORD   dwSize;
   VTDWORD   dwNode;
   VTDWORD   dwFlags;
   VTTCHAR   szPath[256];
   VTTCHAR   szName[256];
   } SCCVWSAVENODE;
```

Parameters

- dwSize: Must be set by the OEM to sizeof(SCCVWSAVENODE).
- dwNode: The number of the node to be saved (zero-based).
- dwFlags: Sets additional information about how to handle the node. Currently the only supported flag is SCCVW_SAVENODEFLAG_RESTORE_PATH indicating

that the full path for a selected node is to be expanded. If this is not set, the node will be saved into the path exactly specified by szPath.

- szName: The name to save the node as, if the node is a file. If the node is a folder, this field is ignored. The name specified in the archive file will be used if this is a null string.
- szPath: The path specifying where to save the node. If this is a null string, the path will be obtained from the option SCCID_ARCOUTPUTPATH. If this path does not exist and cannot be created, the save message will fail.

4.53 SCCVW_SEARCH

* Causes the view window to scan the document for a text string.

Parameters

- wParam: Should be set to 0.
- lParam: Pointer to a structure of type SCCVWSEARCHINFO40 or SCCVWSEARCHINFO80.

Return Value

One of the following:

- 0: The search string was found.
- -1: An error occurred in searching.
- 1: The search reached either end of the file before finding the search string.

Platform

Windows

4.53.1 SCCVWSEARCHINFO40 and SCCVWSEARCHINFO80 Structures

These structures are used by the SCCVW_SEARCH and SCCVW_SEARCHDIALOG messages to specify what to search for in a file. Note that while both of these structures can accept Unicode, only the SCCVWSEARCHINFO80 structure is truly intended to accept Unicode.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct
   {
   VTDWORD   dwSize;
   VTTCHAR   siText[80];
   VTWORD    siTextLen;
   VTWORD    siType;
   VTWORD    siFrom;
   VTWORD    siDirection;
   } SCCVWSEARCHINFO40;

typedef struct
   {
   VTDWORD   dwSize;
   VTWORD    siText[80];
```

```
VTWORD    siTextLen;
VTWORD    siType;
VTWORD    siFrom;
VTWORD    siDirection;
} SCCVWSEARCHINFO80;
```

Parameters

- dwSize: Must be set by the developer to sizeof(SCCVWSEARCHINFO40) or sizeof(SCCVWSEARCHINFO80).
- siText: String to search for.
- siTextLen: Length of siText. This should be in bytes for the SCCVWSEARCHINFO40 structure and words for the SCCVWSEARCHINFO80 structure.
- siType: One of the following:
  - SCCVW_SEARCHCASE: Case sensitive search.
  - SCCVW_SEARCHNOCASE: Case insensitive search.
- siFrom: One of the following:
  - SCCVW_SEARCHTOP: Start the search at the top of the file.
  - SCCVW_SEARCHBOTTOM: Start the search at the bottom of the file.
  - SCCVW_SEARCHCURRENT: Start the search at the current caret position.
- siDirection: One of the following:
  - SCCVW_SEARCHFORWARD: Search towards the bottom of the file.
  - SCCVW_SEARCHBACK: Search towards the top of the file.

4.54 SCCVW_SEARCHDIALOG

* Causes the view window to display a search dialog.

Parameters

- wParam: Should be set to 0.
- lParam: Pointer to a structure of type SCCVWSEARCHINFO40 or SCCVWSEARCHINFO80, this value may be NULL if the developer does not want to set or retrieve search information. See the "Comments" for this message for details.

Return Value

One of the following:

- 0: The search string was found.
- -1: An error occurred in searching.
- 1: The search reached either end of the file before finding the search string.

Comments

The search dialog is initialized to the values in SCCVWSEARCHINFO40 or SCCVWSEARCHINFO80 passed in lParam. If the user clicks OK, the search is

executed and the SCCVWSEARCHINFO40 or SCCVWSEARCHINFO80 passed in lParam is set to the values the user entered.

Platform

Windows

4.55 SCCVW_SEARCHNEXT

* Causes the view window to scan the file for the same text string specified in a previous SCCVW_SEARCH or SCCVW_SEARCHDIALOG message.

Parameters

- wParam: One of the following:
  - SCCVW_SEARCHFORWARD: Search towards the bottom of the file.
  - SCCVW_SEARCHBACK: Search towards the top of the file.
  - SCCVW_SEARCHSAME: Continue searching in the same direction as the original search.
- lParam: Should be set to 0.

Return Value

One of the following:

- 0: The search string was found.
- -1: An error occurred in searching.
- 1: The search reached either end of the file before finding the search string.

Platform

Windows

4.56 SCCVW_SELCHANGE

* * This message is sent to the parent of the view window when the selection state has changed. The parent may then call SCCVW_GETCLIPINFO to determine the view window's readiness to copy to the clipboard.

Parameters

- wParam: The handle of the view window that is sending this message.
- lParam: Should be set to 0.

Return Value

none

Platform

Windows

4.57 SCCVW_SELECTALL

* Causes the viewer window to select all data in the view window.

Parameters

- wParam: Should be set to 0.
- lParam: Should be set to 0.

Return Value

none

Platform

Windows

4.58 SCCVW_SETDISPLAYNAME

* This call sets the string used when a human readable form of the file name needs to be displayed (like in a dialog). This is useful if the file is a temporary copy of another file (like attachments in mail programs).

Parameters

- wParam: Should be set to 0.
- lParam: Pointer to a NULL-terminated string with a length less than SCCVW_DISPLAYNAMEMAX.

Return Value

One of the following values:

- SCCVWERR_OK: The display name was set.
- SCCVWERR_UNKNOWN: There was a problem setting the display name.

Platform

Windows

4.59 SCCVW_SETHSCROLLPAGESIZE

* * This message is sent from the view window when the page size (thumb size) of the horizontal scroll bar changes. It allows the OEM to establish correct thumb sizes of scroll bars external to the view window.

Parameters

- wParam: The handle of the view window that is sending this message.
- lParam: Contains the new page size.

Platform

Windows

4.60 SCCVW_SETHSCROLLPOSITION

* * This message is sent from the view window when the position of the horizontal scroll bar changes. It allows the OEM to establish correct positioning of scroll bars external to the view window.

Parameters

- wParam: The handle of the view window that is sending this message.
- lParam: Contains the new position.

Platform

Windows

4.61 SCCVW_SETHSCROLLRANGE

* * This message is sent from the view window when the range of the horizontal scroll bar changes. It allows the OEM to establish correct positioning of scroll bars external to the view window.

Parameters

- wParam: The handle of the view window that is sending this message.
- lParam: LOWORD(lParam) contains the new minimum position. HIWORD(lParam) contains the new maximum position.

Platform

Windows

4.62 SCCVW_SETHSCROLLSTATE

* * This message is sent from the view window when the state of the horizontal scroll bar changes. It allows the OEM to establish correct positioning of scroll bars external to the view window.

Parameters

- wParam: The handle of the view window that is sending this message.
- lParam: One of the following:
  - SCCSB_ENABLED: Scroll bar should be enabled.
  - SCCSB_DISABLED: Scroll bar should be disabled.

Platform

Windows

4.63 SCCVW_SETIDLEBITMAP

* Sets the bitmap to be centered in the view window when no file is being viewed.

Parameters

- wParam: An instance handle of the EXE or DLL that contains the bitmap.
- lParam: LOWORD(lParam) should contain the resource ID of the bitmap. HIWORD(lParam) Should be set to 0.

Return Value

None.

Platforms

Windows

4.64 SCCVW_SETMENUMAX

* This message informs the view window of the maximum value of the developer's menu IDs. This allows the view window to use IDs above this value when creating menus or menu items for its menus. The default value if the message is not sent is #defined as SCCVW_DEFAULTMENUMAX.

Parameters

- wParam: Highest menu ID used by the caller.
- lParam: Not used.

Platforms

Windows

4.65 SCCVW_SETOPTION

* This message allows the developer to set the current or default value of one of the many options available in the Viewer. The term option is a little misleading because some options act much more like properties or events in that they have an immediate impact on the display. In some cases they are very much like events in that they are only triggers for an operation (such as zooming in or out of a bitmap).

Parameters

- wParam: Should be set to 0.
- lParam: A pointer to a SCCVWOPTIONSPEC40 structure.

Return Value

One of the following values:

- SCCVWERR_OK: Option has been set.
- SCCVWERR_BADPARAM: Structure size was incorrect or the option ID was bad.

Comment

The current option is the value the view is using. The default option is the initial value which a new view will have.

Platform

Windows

4.65.1 SCCVWOPTIONSPEC40 Structure

This structure is passed by the developer through the SCCVW_GETOPTION or SCCVW_SETOPTION message.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct
   {
   VTDWORD   dwSize
   VTDWORD   dwId;
   VTDWORD   dwFlags;
   VTVOID    pData;
   } SCCVWOPTIONSPEC40;
```

Parameters

- dwSize: Must be set by the developer to sizeof(SCCVWOPTIONSPEC40).
- dwId: The ID of the option the developer wants to set or get. These IDs, their data types and their functions are listed in the Options chapter of this book.
- dwFlags: Any of the following OR-ed together:
  - SCCVWOPTION_CURRENT: Set/Get the current value of the option.
  - SCCVWOPTION_DEFAULT: Set/Get the default value of the option.

  These two flags may only be used together when passing the structure to SCCVW_SETOPTION. In SCCVW_GETOPTION they are mutually exclusive.
- pData: A pointer to a data area containing the option's value on a Set or receiving the option's value on a Get. This area must be as large as the data type specified for the option ID the developer is getting or setting.

4.66 SCCVW_SETSELECTION

* This message allows the parent to set the anchor and end of selection given two positions. Setting the caret position can be achieved by setting the anchor and end of selection to the same position.

Parameters

- wParam: Should be set to 0.
- lParam: Pointer to a structure of the type SCCVWSETSELECTION41.

Return Value

One of the following values:

- SCCVWERR_OK: The mapping was successful.
- SCCVWERR_BADPARAM: One of the values in the SCCVWSETSELECTION41 structure was invalid.

Platform

Windows

4.66.1 SCCVWSETSELECTION41 Structure

This structure is passed by the OEM through the SCCVW_SETSELECTION message.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct
   {
   VTDWORD    dwSize;
   SCCVWPOS   sAnchorPos;
   SCCVWPOS   sEndPos;
   } SCCVWSETSELECTION41;
```

Parameters

- dwSize: Must be set by the OEM to sizeof(SCCVWSETSELECTION41).
- sAnchorPos: Identifies the start selection position.
- sEndPos: Identifies the end selection position.

4.67 SCCVW_SETVSCROLLPAGESIZE

* * This message is sent from the view window when the page size (thumb size) of the vertical scroll bar changes. It allows the OEM to establish correct thumb sizes of scroll bars external to the view window.

Parameters

- wParam: The handle of the view window that is sending this message.
- lParam: Contains the new page size.

Platform

Windows

4.68 SCCVW_SETVSCROLLPOSITION

* * This message is sent from the view window when the position of the vertical scroll bar changes. It allows the OEM to establish correct positioning of scroll bars external to the view window.

Parameters

- wParam: The handle of the view window that is sending this message.
- lParam: Contains the new position.

Platform

Windows

4.69 SCCVW_SETVSCROLLRANGEMIN

* * This message is sent from the view window when the range of the vertical scroll bar changes. It allows the OEM to establish correct positioning of scroll bars external to the view window.

Parameters

- wParam: The handle of the view window that is sending this message.
- lParam: The new value sent from the view window for the new minimum position.

Platform

Windows

4.70 SCCVW_SETVSCROLLRANGEMAX

* * This message is sent from the view window when the range of the vertical scroll bar changes. It allows the OEM to establish correct positioning of scroll bars external to the view window.

Parameters

- wParam: The handle of the view window that is sending this message.
- lParam: The new value sent from the view window for the new maximum position.

Platform

Window

4.71 SCCVW_SETVSCROLLSTATE

* * This message is sent from the view window when the state of the vertical scroll bar changes. It allows the OEM to establish correct positioning of scroll bars external to the view window.

Parameters

- wParam: The handle of the view window that is sending this message.
- lParam: One of the following:
  - SCCSB_ENABLED: Scroll bar should be enabled.
  - SCCSB_DISABLED: Scroll bar should be disabled.

Platform

Windows

4.72 SCCVW_VIEWAS

* This call instructs the viewer to display files identified as unknown in a specific format.

Parameters

- wParam: Set to 0 to allow the view window to decide how to view the file.
- lParam: Should be set to 0.

Platform

Windows

4.73 SCCVW_VIEWFILE

* Causes the viewer to open and display the contents of a file. If another file is already open, it will be closed automatically. No intervening SCCVW_CLOSEFILE is necessary.

Parameters

- wParam: Should be set to 0.
- lParam: A pointer to a SCCVWVIEWFILE40 or SCCVWVIEWFILE80 structure.

Return Value

One of the following values:

- SCCVWERR_OK: Open was successful.
- SCCVWERR_INVALIDID: File ID passed or generated is invalid.
- SCCVWERR_FILTERLOADFAILED: The filter DLL for this file type could not be loaded.
- SCCVWERR_FILTERALLOCFAILED: The filter DLL could not allocate enough memory.
- SCCVWERR_NOFILTER: No filter available for this file type.
- SCCVWERR_DISPLAYINITFAILED: The display window could not be initialized.
- SCCVWERR_CHUNKERINITFAILED: The chunker (SCCCH.DLL) could not be initialized.
- SCCVWERR_FILEOPENFAILED: The file could not be opened.
- SCCVWERR_UNKNOWNFAILURE: Unknown type of failure.
- SCCVWERR_BADFILE: The file is corrupt.
- SCCVWERR_EMPTYFILE: The file is empty.
- SCCVWERR_PROTECTEDFILE: The file is password protected or encrypted.

- SCCVWERR_SUPFILEOPENFAILED: The filter could not open additional files needed to view the specified file.
- SCCVWERR_BADPARAM: One of the parameters was invalid.
- SCCVWERR_DRMFILE - The file is DRM protected
- SCCVWERR_UNSUPPORTEDCOMPRESSION - The file is compressed with an unsupported compression algorithm
- SCCVWERR_NODISPLAYENGINE - The appropriate display engine DLL for this file could not be loaded
- SCCVWERR_FILTERTIMEOUT - The input filter timed out while processing this file
- SCCVWERR_MACINITFAILED - The SCCVW_MACINIT message process failed
- SCCVWERR_RAWTEXTDISABLED - Raw Text support has been disabled
- SCCVWERR_OTHERPRINTING - Another print process is already in progress
- SCCVWERR_NODATATORENDER - Graphical rendering of this format is unsupported
- SCCVWERR_BADCREDENTIALS - Invalid credentials were provided for opening this file
- SCCVWERR_OUTPUT_SOL_MISSING - No valid output solution DLL was found
- SCCVWERR_USEROFFSET - This error is unused/deprecated

Platforms

Windows

4.73.1 SCCVWVIEWFILE40 and SCCVWVIEWFILE80 Structures

These structures are used by the SCCVW_VIEWFILE and SCCVW_VIEWTHISFILE messages to pass the description of a file to view. Note that while both of these structures can accept Unicode, only the SCCVWVIEWFILE80 structure is truly intended to accept Unicode.

Structure

C data structures, defined in sccvw as follows:

```
typedef struct
   {
   VTDWORD        dwSize;
   VTDWORD        dwSpecType;
   VTVOID       * pSpec;
   VTDWORD        dwViewAs;
   VTBOOL         bUseDisplayName;
   VTTCHAR        szDisplayName[SCCVW_DISPLAYNAMEMAX];
   VTBOOL         bDeleteOnClose;
   VTDWORD        dwFlags;
   VTDWORD        dwReserved1;
   VTDWORD        dwReserved2;
   } SCCVWVIEWFILE40,  * PSCCVWVIEWFILE40;

typedef struct
   {
   VTDWORD        dwSize;
   VTDWORD        dwSpecType;
   VTVOID       * pSpec;
   VTDWORD        dwViewAs;
```

```
VTBOOL          bUseDisplayName;
VTWORD          szDisplayName[SCCVW_DISPLAYNAMEMAX];
VTBOOL          bDeleteOnClose;
VTDWORD         dwFlags;
VTDWORD         dwReserved1;
VTDWORD         dwReserved2;
} SCCVWVIEWFILE80,  * PSCCVWVIEWFILE80;
```

Parameters

- dwSize: Must be set by the developer to sizeof(SCCVWVIEWFILE40) or sizeof(SCCVWVIEWFILE80), depending on which structure is in use.
- dwSpecType: Describes the contents of pSpec, must be one of the following values:
  - IOTYPE_ANSIPATH: pSpec points to a NULL-terminated full path name using the ANSI character set and FAT 8.3 (Win16) or NTFS (Win32 and Win64) file name conventions.
  - IOTYPE_UNICODEPATH: pSpec points to a NULL-terminated full path name using the Unicode character set and NTFS (Win32 and Win64) file name conventions.
  - IOTYPE_REDIRECT: This allows the developer to redirect the IO routines used to read the file.
- pSpec: See the descriptions for individual dwSpecType values in the preceding list.
- dwViewAs: The HIWORD should be set to 0. The LOWORD should be 0 (to allow the view window to decide how to view the file).
- bUseDisplayName: If TRUE, szDisplayName is used when a human readable representation of the file needs to be displayed. If FALSE, the viewer will generate a display name from the file name.
- szDisplayName: A NULL-terminated string used when a human readable form of the file name needs to be displayed (like in a dialog), this is useful if the file is a temporary copy of another file (like attachments in mail programs). The string should be in the viewer's current character set. SCCDISPLAYNAME_MAX will be long enough to hold the longest file name for a given operating environment and sometimes longer. For instance, under FAT 8.3 (DOS), file names are 12 bytes long, but the Viewer uses 40 bytes so a more readable name can be set by the developer.
- bDeleteOnClose: If set to TRUE, the view window will delete the file when the view of it is closed by SCCVW_CLOSEFILE, SCCVW_VIEWFILE or destroying the view window.
- dwFlags: The only currently valid value for this parameter is 0.
- dwReserved1, dwReserved2: Both are currently unused and should be set to 0.

4.74 SCCVW_VIEWTHISFILE

* * This message is sent from the view window to the developer when another file should be viewed. Currently, this occurs is when the user double-clicks or hits return on a file entry in an Archive view and on a hyperlink to a referenced document. In the case of the archive formats, the display engine decompresses the file then sends a SCCVW_VIEWTHISFILE message to the developer. In the case of a hyperlink, if the

annotation is not overridden by the developer, this message is generated. The parent can ignore this message or create a new view to view the decompressed file. The parent should use SCCVW_VIEWFILE to open the decompressed file.

Parameters

- wParam: The handle of the view window that is sending this message.
- lParam: A pointer to a SCCVWVIEWTHISFILE40 or SCCVWVIEWTHISFILE80 structure which contains a SCCVWVIEWFILE40 or SCCVWVIEWFILE80 structure that can be passed unmodified through a SCCVW_VIEWFILE message.

Return Value

Must be SCCVWERR_MESSAGEHANDLED if you have viewed the file. Any other return value indicates that you have no interest in the file and the view window should clean up if necessary. Clean up usually entails deleting the file if it is temporary.

> **Note:**
>
> The SCCVWVIEWFILE40 or SCCVWVIEWFILE80 structure inside the SCCVWVIEWTHISFILE40 or SCCVWVIEWTHISFILE80 structure contains a pointer to data (pSpec) that is only valid until you return from this message. The pSpec for a SCCVWVIEWTHISFILE40 or SCCVWVIEWTHISFILE80 structure copied into a local copy cannot be expected to still be valid at a later time
>
> Do not send the SCCVW_VIEWFILE message to the same view window that sent you the SCCVW_VIEWTHISFILE message.

Platforms

Windows

4.74.1 SCCVWVIEWTHISFILE40 and SCCVEVIEWTHISFILE80 Structures

These structures are passed to the developer through the SCCVW_VIEWTHISFILE message. Note that while both of these structures can accept Unicode, only the SCCVWVIEWTHISFILE80 structure is truly intended to accept Unicode.

Structure

A C data structure defined in sccvw as follows:

```
typedef struct
   {
   VTDWORD           dwSize;
   SCCVWVIEWFILE40   sViewFile;
   VTWORD            wFileTime;
   VTWORD            wFileDate;
   VTDWORD           dwFileSize;
   } SCCVWVIEWTHISFILE40;

typedef struct
   {
```

```
VTDWORD           dwSize;
SCCVWVIEWFILE80   sViewFile;
VTWORD            wFileTime;
VTWORD            wFileDate;
VTDWORD           dwFileSize;
} SCCVWVIEWTHISFILE80;
```

Parameters

- dwSize: Will be set to sizeof(SCCVWVIEWTHISFILE40) or sizeof(SCCVWVIEWTHISFILE80).
- sViewFile: A SCCVWVIEWFILE40 or SCCVWVIEWFILE80 structure that may be a passed unmodified to another view window through the SCCVW_VIEWFILE message.
- wFileTime, wFileDate, dwFileSize: Reserved by Oracle.

4.75 SCCVW_VSCROLL

* This message allows the OEM to cause vertical scrolling behavior.

Parameters

- wParam: Code describing how to scroll. One of the following values:
  - SCCSB_TOP: Scroll to top.
  - SCCSB_BOTTOM: Scroll to bottom.
  - SCCSB_LINEUP: Scroll up one line as if the up arrow on the scroll bar had been pressed.
  - SCCSB_LINEDOWN: Scroll down one line as if the down arrow on the scroll bar had been pressed.
  - SCCSB_PAGEUP: Scroll up one view window page.
  - SCCSB_PAGEDOWN: Scroll down one view window page.
  - 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_POSITION: Sets the scroll bar to the position given in lParam.
- lParam: Position used if wParam = SCCSB_POSITION. Its value must be in the range given by the last SCCVW_SETSCROLLRANGE message sent to the OEM.

Return Value

One of the following values:

- SCCVWERR_OK: View window was scrolled.
- SCCVWERR_BADPARAM: One of the values in wParam or lParam is bad.

Platform

Windows

5
Redirected IO

This chapter describes the use of Redirected IO in the Viewer. Many developers using the earlier versions of this technology expressed a need to read file data from non-file system based sources. For instance, the developer might want to read the file from a database on a server. Perhaps the developer is downloading the file over a slow link, and wants to see the first screen of a document before the download is completed, or only wants to download enough to view the first screen. To address these requests, developers now have total control over access to a file via Oracle Outside In's redirected IO mechanism.
This chapter includes the following sections:

5.1 Using Redirected IO

A developer can redirect the IO for an input or output file by providing a data structure that contains pointers to custom IO routines for reading and writing. This data structure is passed in place of a typical file specification. The developer must set the dwSpecType parameter of the SCCVWVIEWFILE40 or SCCVWVIEWFILE80 structure to IOTYPE_REDIRECT when the SCCVW_VIEWFILE message is sent.

When dwSpecType is set this way, the pSpec element must contain a pointer to a developer-defined data structure that begins with a BASEIO structure (defined in baseio.h). The BASEIO structure contains pointers to the basic IO functions for the view window's IO system such as Read, Seek, Tell, etc. The developer must initialize these function pointers to their own functions that perform IO tasks. Beyond the BASEIO element, the developer may place any other data. For instance, a developer's structure might look like the following:

```
typedef struct MYFILEtag
{
   BASEIO    sBaseIO;       /* must be the first element */
   VTDWORD   dwMyInfo1;
   VTDWORD   dwMyInfo2;
   .
   .
   .
} MYFILE;
```

Since the pSpec passed is essentially the file handle that the view window uses, the developer can redirect the IO on a file-by-file basis while still viewing regular disk-based files.

The BASEIO structure is defined as follows:

```
typedef struct BASEIOtag
{
    IOCLOSEPROC pClose;
    IOREADPROC pRead;
    IOWRITEPROC pWrite;
    IOSEEKPROC pSeek;
    IOTELLPROC pTell;
    IOGETINFOPROC pGetInfo;
    IOOPENPROC pOpen; /* pOpen *MUST* be set to NULL. */
#ifndef NLM
    IOSEEK64PROC pSeek64;
    IOTELL64PROC pTell64;
#endif
    VTVOID *aDummy[3];
} BASEIO, * PBASEIO;
```

The developer must implement the Close, Read, Seek, Tell and GetInfo routines. The Write routine can be a dummy routine and the Open routine must be set to NULL. The first parameter to each of these routines is called hFile and is of the type HIOFILE. HIOFILE is simply the VTLPVOID to your data structure that was passed in the pSpec parameter of the SCCVWVIEWFILE40 or SCCVWVIEWFILE80 structure.

The sample source code for a simple implementation of Redirected IO is in the directory SAMPLES/REDIRect. This sample redirects the technology's IO through the fopen, fgetc, fseek, ftell and fclose run-time library routines.

> **Note:**
>
> Redirected IO does not cache the whole file! Seeks can and will occur throughout the file during the course of viewing. If the developer is implementing redirected IO on a slow or sequential link, it is the developer's responsibility to cache the file locally.

5.2 IOClose

Closes the file identified by hFile and cleans up all memory associated with the file.

Prototype

```
IOERR IOClose(
   HIOFILE   hFile);

Parameters
```

- hFile: Identifies the file to be closed. Should be cast into a pointer to your data structure (MYFILE in the preceding discussion).

Return Values

- IOERR_OK: Close was successful.
- IOERR_UNKNOWN: Some error occurred on close.

5.3 IORead

Reads data from the current file position forward and resets the position to the byte after the last byte read.

Prototype

```
IOERR IORead(
   HIOFILE         hFile,
   VTBYTE       *  pData,
   VTDWORD         dwSize,
   VTDWORD      * pCount);
```

Parameters

- hFile: Identifies the file to be read. Should be cast into a pointer to your data structure (MYFILE in the preceding discussion).
- pData: Points to the buffer into which the bytes should be read. Will be at least dwSize bytes big.
- dwSize: Number of bytes to read.
- pCount: Points to the number of bytes actually read by the function. This value is only valid if the return value is IOERR_OK.

Return Values

- IOERR_OK: Read was successful. pCount contains the number of bytes read and pData contains the bytes themselves.
- IOERR_EOF: Read failed because the file pointer was beyond the end of the file at the time of the read.
- IOERR_UNKNOWN; Read failed for some other reason.

5.4 IOWrite

Writes data from the current file position forward and resets the position to the byte after the last byte written.

> **Note:**
>
> This function has been fully documented only for completeness. OEMs who use redirected IO do not need to implement writing and the IOWrite function should do nothing but return IOERR_UNKNOWN.

Prototype

```
IOERR IOWrite(
   HIOFILE         hFile,
   VTBYTE       *  pData,
   VTDWORD         dwSize,
   VTDWORD      * pCount);
```

Parameters

- hFile: Identifies the file where the data is to be written. Should be cast into a pointer to your data structure (MYFILE in the preceding discussion).
- pData: Points to the buffer from which the bytes should be written. It must be at least dwSize bytes big.
- dwSize: Number of bytes to write.
- pCount: Points to the number of bytes actually written by the function. This value is only valid if the return value is IOERR_OK.

Return Values

- IOERR_OK: Write was successful, pCount contains the number of bytes written.
- IOERR_UNKNOWN: Write failed for some reason.

5.5 IOSeek

Moves the current file position.

Prototype

```
IOERR IOSeek(
   HIOFILE   hFile,
   VTWORD    wFrom,
   VTLONG    lOffset);
```

Parameters

- hFile: Identifies the file to be read. Should be cast into a pointer to your data structure (MYFILE in the preceding discussion).
- wFrom: One of the following values:
  - IOSEEK_TOP: Move the file position lOffset bytes from the top (beginning) of the file.
  - IOSEEK_BOTTOM: Move the file position lOffset bytes from the bottom (end) of the file.
  - IOSEEK_CURRENT: Move the file position lOffset bytes from the current file position.
- lOffset: Number of bytes to move the file pointer. A positive value moves the file pointer forward in the file and a negative value moves it backward. If a requested seek value would move the file pointer before the beginning of the file, the file pointer should remain unchanged and IOERR_UNKNOWN should be returned. Seeking past EOF is allowed. In that case IOERR_OK should be returned. IOTell would return the requested seek position and IORead should return IOERR_EOF and 0 bytes read.

Return Values

- IOERR_OK: Seek was successful.
- IOERR_UNKNOWN: Seek failed for some reason.

5.6 IOTell

Returns the current file position.

Prototype

```
IOERR IOTell(
   HIOFILE         hFile,
   VTDWORD       * pOffset);
```

Parameters

- hFile: Identifies the file to be read. Should be cast into a pointer to your data structure (MYFILE in the preceding discussion).
- pOffset: Points to the current file position returned by the function.

Return Values

- IOERR_OK: Tell was successful.
- IOERR_UNKNOWN: Tell failed for some reason.

5.7 IOGetInfo

Returns information about an open file.

Prototype

```
IOERR IOGetInfo(
   HIOFILE        hFile,
   VTDWORD        dwInfoId,
   VTVOID       * pInfo);
```

Parameters

- hFile: Identifies the file to be read. Should be cast into a pointer to your data structure (MYFILE in the previous discussion).
- dwInfoId: One of the following values:
  - IOGETINFO_FILENAME: pInfo points to a string that should be filled with the base file name (no path) of the open file (for example TEST.DOC). If you do not know the file name, return IOERR_UNKNOWN. Certain file types (such as DataEase) must know the original file name in order to open secondary files required to correctly view the original file. If you return IOERR_UNKNOWN, these file types will not convert. See the description of IOGETINFO_GENSECONDARY below.
  - IOGETINFO_PATHNAME: pInfo points to a string that should be filled with the fully qualified path name (including the file name) of the open file. For example, C:\MYDIR\TEST.DOC. If you do not know the path name, return IOERR_UNKNOWN.
  - IOGETINFO_PATHTYPE: pInfo points to a DWORD that should be filled with the IOTYPE of the path returned by IOGETINFO_PATHNAME. For instance, if you return a DOS path name in the Unicode character set, you should return IOTYPE_UNICODEPATH.
  - IOGETINFO_ISOLE2STORAGE: Must return IOERR_FALSE. pInfo is not used.

- IOGETINFO_GENSECONDARY: pInfo points to a structure of type IOGENSECONDARY. Some file types require supporting files to be opened. These supporting files may contain formatting information or extra data. Also, templates may link to other templates, and the paths to those templates must be resolved. Correct handling of IOGETINFO_GENSECONDARY is critical to the operation of the Oracle Outside In technology.

  Since the developer is in total control of the IO for the primary file, the technology does not know how to generate a path to these secondary files or even if the secondary files are accessible through the regular file system. The IOGETINFO_GENSECONDARY call gives the developer a chance to resolve this problem by generating a new IO specification for the secondary file in question. The developer gets just the base file name (often embedded in the original document or generated from the primary file's name) of the secondary file.

  The developer may either use one of the standard Oracle Outside In IO types or totally redirect the IO for the secondary file, as well.

- IOGETINFO_64BITIO: For redirected I/O that wishes to use 64-bit seek/tell functions, your IOGetInfo function must respond IOERR_TRUE to this dwInfoId. In addition, the pSeek64/pTell64 items in the baseio structure must be valid pointers to the proper function types.

- IOGETINFO_DPATHNAME: pInfo points to a structure of type DPATHNAME, which should be filled with the fully qualified path name (including the file name) of the open file, for example, C:\MYDIR\TEST.DOC. If you do not know the path name, return IOERR_UNKNOWN. The dwPathLen element contains the size of the buffer pointed to by the pPath element. If the buffer size is too small to contain the full path, modify dwPathLen to be the correct size of the buffer required to hold the path name in its IOTYPE character width including the NULL terminator and return IOERR_INSUFFICIENTBUFFER.

  The following is a C data structure defined in SCCIO.H:

```
typedef struct DPATHNAMEtag
{
    VTDWORD  dwPathLen;
    VTVOID  *pPath;
} DPATHNAME, * PDPATHNAME;
```

  Parameters

  dwPathLen: Will be set to the number of bytes in the buffer pointed to by pPath. If the size of the buffer is insufficient, reset this element to the number of bytes required and return IOERR_INSUFFICIENTBUFFER.

  pPath: Points to the buffer to be filled with the path name.

- IOGETINFO_GENSECONDARYDP: pInfo points to a structure of type IOGENSECONDARYDP. The dwSpecLen element contains the size of the buffer pointed to by the pSpec element. If the buffer size is too small to contain the spec, modify dwSpecLen to be the correct size of the buffer required to hold the path in its IOTYPE character width including the NULL terminator and return IOERR_INSUFFICIENTBUFFER.

  The following is a C data structure defined in SCCIO.H:

```
typedef struct IOGENSECONDARYDPtag
{
    VTDWORD         dwSize;
```

```
    VTVOID *        pFileName;
    VTDWORD         dwSpecType;
    VTVOID *        pSpec;
    VTDWORD         dwSpecLen;
    VTDWORD         dwOpenFlags;
} IOGENSECONDARYDP, * PIOGENSECONDARYDP;
```

Parameters

dwSize: Will be set to sizeof (IOGENSECONDARYDP)

pFileName: A pointer to a string representing the file name of the secondary file that the technology requires. It is usually a name stored in the primary file (such as MYSTYLE.STY for a Word for DOS file) or a name generated from the primary file name. The primary file for a DataEase database has a .dba extension. The secondary name is the same file name but with a .dbm extension.

dwSpecType: The developer must fill this with the IOSPEC for the secondary file.

pSpec: On entry, this pointer points to an array of bytes or may be NULL (see dwSpecLen below). If the dwSpecType is set a regular IOTYPE such as IOTYPE_ANSIPATH, the developer may fill this array with the path name or structure required for that IOTYPE. If the developer is redirecting access to the secondary file, then dwSpecType will be IOTYPE_REDIRECT and the developer should replace pSpec with a pointer to a developer-defined structure that begins with the BASEIO structure (see Using Redirected IO).

The file is supposed to be opened by the OEM's redirected IO code by the time they return the BASEIO struct. This is because the pOpen routine in the BASEIO struct is supposed to be NULL.

dwSpecLen: On entry, this is set to the size of the pSpec buffer. If the size of the buffer is insufficient, replace the value with the number of bytes required and return IOERR_INSUFFICIENTBUFFER.

dwOpenFlags: Set by the technology. A set of bit flags describing how the secondary file should be opened. Multiple flags may be used by bitwise OR-ing them together. The following flags are currently used:

- IOOPEN_READ: The secondary file should be opened for read.

- IOOPEN_WRITE: The secondary file should be opened for write. If the specified file already exists, its contents are erased when this flag is set.

- IOOPEN_CREATE: The secondary file should be created (if it does not already exist) and opened for write.

Any other value should return IOERR_BADINFOID.

- pInfo: The size of the pInfo buffer depends on the dwInfoId selected. For IOGETINFO_FILENAME and IOGETINFO_PATHNAME, the buffer is of size MAX_PATH characters (each character is either one byte or two, depending on PATHTYPE). The IOGETINFO_PATHTYPE buffer is the size of a VTDWORD.

Return Values

- IOERR_OK: GetInfo was successful.
- IOERR_TRUE: Affirmative response from a true or false GetInfo.
- IOERR_FALSE: Negative response from a true or false GetInfo.
- IOERR_BADINFOID: dwInfoId can not be handled by this file type.

- IOERR_INVALIDSPEC: The file spec is bad for this type.
- IOERR_UNKNOWN: GetInfo failed for some other reason.

5.7.1 IOGENSECONDARY and IOGENSECONDARYW Structures

These structures are passed to the developer through the IOGetInfo function. They allow the developer to tell the technology where a secondary file, needed to view the primary file, is located.

The SpecType of the original file determines which of these two structures is used. If the SpecType is IOTYPE_UNICODEPATH, IOGENSECONDARYW is used. pFileName will point to a Unicode string terminated with a NULL WORD. For all other SpecTypes, IOGENSECONDARY is used and pFileName will point to a string terminated with a NULL BYTE.

The following is a C data structure defined in SCCIO.H:

```
typedef struct
{
   VTDWORD     dwSize;
   VTLPBYTE    pFileName;
   VTDWORD     dwSpecType;
   VTLPVOID    pSpec;
   VTDWORD     dwOpenFlags
} IOGENSECONDARY, * PIOGENSECONDARY;


typedef struct
{
   VTDWORD     dwSize;
   VTLPWORD    pFileName;
   VTDWORD     dwSpecType;
   VTLPVOID    pSpec;
   VTDWORD     dwOpenFlags
} IOGENSECONDARYW, * PIOGENSECONDARYW;
```

- dwSize: Will be set to sizeof (IOGENSECONDARY) or sizeof (IOGENSECONDARYW) (both of these values are the same).
- pFileName: A pointer to a string representing the file name of the secondary file that the technology requires. It will generally be a name that is stored in the primary file somewhere (such as MYSTYLE.STY for a Word for DOS file) or a name generated from the primary file name (the primary file for a DataEase database will always have a .dba extension, the secondary name would be the same file name but with a .dbm extension).
- dwSpecType: The developer must fill this with the IOSPEC for the secondary file.
- pSpec: On entry, this pointer points to an array of 1024 bytes. If the dwSpecType is set a regular IOTYPE such as IOTYPE_ANSIPATH, the developer may fill this array with the path name or structure required for that IOTYPE. If the developer is redirecting access to the secondary file, then dwSpecType will be IOTYPE_REDIRECT and the developer should replace pSpec with a pointer to a developer-defined structure that begins with the BASEIO structure.
- Note the file is supposed to be opened by the OEM's redirected IO code by the time they return the BASEIO struct. This is because the pOpen routine in the BASEIO struct is supposed to be NULL.

- dwOpenFlags: Set by the technology. A set of bit flags describing how the secondary file should be opened. Multiple flags may be used by bitwise OR-ing them together. The following flags are currently used:
  - IOOPEN_READ: The secondary file should be opened for read.
  - IOOPEN_WRITE: The secondary file should be opened for write. Please note that if the specified file already exists, it's contents will be erased when this flag is set.
  - IOOPEN_CREATE: The secondary file should be created (if it does not already exist) and opened for write.

5.7.2 File Types That Cause IOGETINFO_GENSECONDARY

This section provides information about applicable file types.

- Microsoft Word for DOS Versions 4, 5 and 6: Used to open and read the style sheet file associated with the document. The filter will successfully degrade if the style sheet is not present.
- Harvard Graphics DOS 3.x: Used to open and read the individual slides within ScreenShow and palette files. Files with the extension .ch3 are individual graphics or slides that can be opened using no secondary files. Files with the extension .sy3 are ScreenShows that reference a list of .ch3 files via the secondary file mechanism. There is also an optional palette file that can be referenced from a .ch3 file, but the filter will successfully degrade if the palette file is not present.
- R:Base: Used to open and read required schema file. The R:Base data files are named ????2.rbf but the data is useless without the schema file named ????1.rbf. There is also a ????3.rbf file associated with each database, but it is not used.
- Paradox 4.0 and Above: Used to open and read memo field data file. Paradox uses a separate file for all memo field data larger than 32 bytes.
- DataEase: Used to open and read the data file. DataEase databases include a .dba file that contains the schema (the file that the technology can identify as DataEase) and a .dbm file that contains the actual data.

5.8 IOSEEK64PROC / IOTELL64PROC

These functions are for seek/tell using 64-bit offsets. These functions are not used by default. Rather, they are used if the IOGETINFO_64BITIO message returns IOERR_TRUE. This is so redirected I/O using strictly 32-bit I/O is unaffected.

5.8.1 IOSeek64

Moves the current file position.

Prototype

```
IOERR IOSeek64(
HIOFILE hFile,
VTWORD wFrom,
VTOFF_T offset);
```

Parameters

The parameter information is the same as for IOSeek(). However, the size of the VTOFF_T offset for IOSeek64() is 64-bit unlike the 32-bit offset in IOSeek().

5.8.2 IOTell64

Returns the current file position.

Prototype

```
IOERR IOTell64(
HIOFILE hFile,
VTOFF_T * pOffset);
```

Parameters

The parameter information is the same as for IOTell(). The only change is the use of a pointer to a 64-bit parameter for returning the offset.

6
Implementation Issues

This chapter discusses issues when using the Oracle Outside In Viewer.

6.1 Running in 24x7 Environments

To ensure robust 24x7 performance in server applications embedding the Viewer, it is strongly recommended that the technology be run in a process separate from the server's primary process.

The file filtering technology underlying the software represents almost a quarter of a million lines of code. This code is expected to robustly deal with any stream of bytes, of any length (any file), in all cases. Oracle has dedicated, and continues to dedicate, significant effort into making this technology extremely robust. However, in real world situations, expect that some small number of malformed files may force the filters into unstable states. This generally results in either a memory exception (which can be trapped and recovered from gracefully), infinite loop or a wild pointer that causes the filter to write into memory that is part of the same process but does not belong to the filter. In the latter situation, this wild pointer condition cannot be trapped.

On the desktop this is not a significant problem since the number of files being dealt with is relatively small. In a 24x7 server environment, however, a wild pointer can be extremely disruptive to the server process and produce serious problems. The best solution for dealing with this problem is to run any application that reads complex file formats, including the Viewer, in a separate process. This solution protects the application from the susceptibility of filtering technology to the unknown quality of input files.

It must be stressed that files that lead to wild pointers or infinite loops occur very infrequently, usually as a result of a third-party conversion process or beta versions of applications. Oracle is committed to addressing these issues and to updating and expanding its testing tools and corpus of documents to proactively minimize this garbage in-garbage out problem.

7
Sample Applications

This chapter describes the samples applications shipped with the Viewer. Each of the sample applications included in this SDK is designed to highlight a specific aspect of the technology's functionality. We ship built versions of these sample applications. The compiled executables should be in the root directory where the product is installed.
This chapter includes the following sections:

7.1 Building the Samples on a Windows System

Microsoft Visual Studio project files are provided for building each of the sample applications. For 32-bit versions of Windows, versions of the project files are provided for Visual Studio 2013 (.vcxproj files).

> **Note:**
>
> Because .vcxproj files may not pick up the right compiler on their own, you need to make sure that you are building with the Win64 configuration in Visual Studio 2013. For 64-bit versions of Windows, only the Visual Studio 2013 versions are available.

The project files for the sample applications can be found in the \sdk\samplecode\win subdirectory of the Oracle Outside In SDK.

7.2 An Overview of the Sample Applications

This section provides an overview of the sample applications provided with this product. Please note that not all of the sample applications are provided for the Windows platform. See the heading of each application's subsection for clarification.

This section includes the following sample applications:

7.2.1 annoex

This sample demonstrates the view window's advanced annotation and positioning capabilities. Messages used include:

- SCCVW_ADDANNOTATION
- SCCVW_ANNOTATIONEVENT
- SCCVW_BAILOUT
- SCCVW_CLEARANNOTATIONS

- SCCVW_COMPPOSITIONS
- SCCVW_COPYTOCLIP
- SCCVW_DISPLAYPOSITION
- SCCVW_FINDANNOTATION
- SCCVW_FINDPOSITION
- SCCVW_GETANNOTATIONDATA
- SCCVW_GETFILEINFO
- SCCVW_GETRAWTEXT
- SCCVW_HILITESTYLE
- SCCVW_MAPPOSITION
- SCCVW_SETSELECTION

It also demonstrates the following options:

- SCCID_SYSTEMFLAGS
- SCCID_FALLBACKFORMAT

7.2.2 annotate

This sample is essentially the same as SIMPLE except that it uses the Oracle Outside In's Raw Text and Annotation ability to annotate all occurrences of the text the in the document. This sample demonstrates the following messages:

- SCCVW_ADDANNOTATION
- SCCVW_ANNOTATIONEVENT
- SCCVW_GOTOANNOTATION
- SCCVW_CLOSEFILE
- SCCVW_COPYTOCLIP
- SCCVW_FILECHANGE
- SCCVW_GETFILEINFO
- SCCVW_PRINT
- SCCVW_VIEWFILE

Messages used include:

- SCCVW_ADDANNOTATION
- SCCVW_CLEARANNOTATIONS
- SCCVW_COMPPOSITIONS
- SCCVW_DISPLAYPOSITION
- SCCVW_FINDANNOTATION
- SCCVW_FINDPOSITION
- SCCVW_GETFILEINFO
- SCCVW_GETRAWTEXT

- SCCVW_GOTOANNOTATION
- SCCVW_MAPPOSITION
- SCCVW_SETSELECTION
- SCCVW_VIEWFILE

7.2.3 drawpage

This application demonstrates the DrawPage feature that can be used to display thumbnails and write pages to bitmaps. Messages used include:

- SCCVW_CLOSEFILE
- SCCVW_COPYTOCLIP
- SCCVW_DEINITDRAWPAGE
- SCCVW_DRAWPAGE
- SCCVW_FILECHANGE
- SCCVW_GETDISPLAYINFO
- SCCVW_GETDRAWPAGEINFO
- SCCVW_GETFILEINFO
- SCCVW_INITDRAWPAGE
- SCCVW_OPTIONCHANGE
- SCCVW_VIEWFILE

It also demonstrates the SCCID_VECPRINTBACKGROUND option.

7.2.4 ExtractArchive

ExtractArchive demonstrates using the DATree API to extract all nodes in an archive.

The application is executed from the command line and takes two parameters, the name of the input file and the name of an output directory for the extracted files:

```
ExtractArchive input_file output_directory
```

7.2.5 mdiview

This application shows how multiple view windows can be supported in an multiple document interface setting. Messages used include:

- SCCVW_FINDPOSITION
- SCCVW_GETIDEALWINDOWSIZE
- SCCVW_GETTREENODE
- SCCVW_MAPTREEPOSITION
- SCCVW_SEARCHDIALOG
- SCCVW_SEARCHNEXT
- SCCVW_SELCHANGE
- SCCVW_SELECTALL

- SCCVW_SETOPTION
- SCCVW_VIEWAS
- SCCVW_VIEWTHISFILE

This sample also demonstrates the following options:

- SCCID_ARCSAVEEVENT
- SCCID_DIALOGFLAGS

7.2.6 mfcview

This application shows a simple viewer implementation using Microsoft Foundation Class. Messages used include:

- SCCVW_GETCLIPINFO

7.2.7 options

This sample app demonstrates a broad set of messages, including the following:

- SCCVW_DODIALOG
- SCCVW_DOHELP
- SCCVW_GETCURRENTPAGENUMBER
- SCCVW_GETOPTION
- SCCVW_HSCROLL
- SCCVW_INFOMESSAGE
- SCCVW_READAHEADDONE
- SCCVW_SETOPTION
- SCCVW_VSCROLL
- SCCVW_VIEWTHISFILE

Additionally, the application demonstrates the following options:

- SCCID_ARCOUTPUTPATH
- SCCID_BMPZOOM
- SCCID_DEFAULTPRINTMARGINS
- SCCID_DISPLAYFONTALIAS
- SCCID_FIFLAGS
- SCCID_FILTERJPG
- SCCID_FILTERLZW
- SCCID_FONTSCALINGFACTOR
- SCCID_HTML_COND_COMMENT_MODE
- SCCID_MAILHEADERHIDDEN
- SCCID_MAILHEADERVISIBLE
- SCCID_MAINTAINZOOM

- SCCID_PRINTFONTALIAS
- SCCID_REORDERMETHOD
- SCCID_RESOURCELIBRARYID
- SCCID_SCROLLFLAGS
- SCCID_SYSTEMFLAGS
- SCCID_TEMPDIR
- SCCID_UNMAPPABLECHAR
- SCCID_VECZOOM
- SCCID_WPDISPLAYMODE

7.2.8 print

This application shows various print options. Messages used include:

- SCCVW_GETFILEINFO
- SCCVW_PRINT
- SCCVW_PRINTEX
- SCCVW_PRINTSETUP

This application also demonstrates the following options:

- SCCID_WHATTOPRINT
- SCCID_PRINTSTARTPAGE
- SCCID_PRINTENDPAGE
- SCCID_PRINTCOLLATE
- SCCID_PRINTCOPIES

7.2.9 redirect

This sample is essentially the same as SIMPLE except that it redirects the Oracle Outside In input routines (Read, Seek, Tell, …) through its own routines. This sample uses the API messages SCCVW_VIEWFILE and SCCVW_GETFILEINFO.

7.2.10 search

This sample is essentially the same as SIMPLE except that it adds the ability to search for strings in the file being viewed. Messages used include:

- SCCVW_GETFILEINFO
- SCCVW_SEARCH
- SCCVW_SEARCHNEXT
- SCCVW_VIEWFILE

7.2.11 simple

This sample demonstrates the simplest implementation of the Viewer. It shows the basics of how to create a view window, view a file, print the viewed file and copy to the clipboard. Messages used include:

- SCCVW_CLOSEFILE
- SCCVW_COPYTOCLIP
- SCCVW_FILECHANGE
- SCCVW_GETFILEINFO
- SCCVW_PRINT
- SCCVW_VIEWFILE

7.2.12 welcome

This application displays the welcome.doc file. Messages used include:

- SCCVW_GETIDEALWINDOWSIZE
- SCCVW_VIEWTHISFILE

A
Viewer Options

Options are parameters affecting the behavior of the Viewer. These options are available to the developer when using Viewer.

All SCCID values are VTDWORDs (4 bytes long).

A.1 Character Mapping

The following character mappings are used.

A.1.1 SCCID_DEFAULTINPUTCHARSET

This option is used in cases where Oracle Outside In 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, Oracle 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. The possible character sets are listed in charsets.h.

When "extended test for text" is enabled (see SCCID_FIFLAGS), this option will still apply to plain-text input files that are not identified as EBCDIC or Unicode.

This option supersedes the SCCOPT_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 SCCOPT_FALLBACKFORMAT, though internally such values will be translated into equivalent values for the SCCOPT_DEFAULTINPUTCHARSET. As a result, if an application were to set both options, the last such value set for either option will be the value that takes effect.

Scope

Global

Data Type

VTDWORD

Default

- CS_SYSTEMDEFAULT: Query the operating system.

Data

The data types are listed in charsets.h.

A.1.2 SCCID_UNMAPPABLECHAR

When the Oracle Outside In Viewer Technology is displaying a character and cannot find that character in any font on the system, it will display a replacement character. This value is the Unicode value for this character. The default value for this character is 0x002a "*".

Scope

Local

Data Type

VTWORD

Data

The Unicode value for the character to use.

Default

- 0x002a = "*"

A.2 Input Handling

The following sections pertain to input handling.

A.2.1 SCCID_FALLBACKFORMAT

This option controls how files are handled when their specific application type cannot be determined. This normally affects all plain-text files, because plain-text files are generally identified by process of elimination, for example, when a file isn't identified as having been created by a known application, it is treated as a plain-text file.

> **Note:**
>
> A number of values that were formerly allowed for this option have been deprecated. Specifically, the values that selected specific plain-text character sets are no longer to be used. Instead, applications should use the SCCID_DEFAULTINPUTCHARSET option for such functionality.

Scope

Global

Data Type

VTDWORD

Data

The high `VTWORD` of this value is reserved and should be set to 0, and the low `VTWORD` must have one of the following values:

- FI_TEXT: Unidentified file types will be treated as text files.
- FI_NONE: Oracle Outside In will not attempt to process files whose type cannot be identified. This will include text files. When this option is selected, an attempt to process a file of unidentified type will cause Oracle Outside In to return an error value of DAERR_FILTERNOTAVAIL (or SCCERR_NOFILTER).

Default

- FI_TEXT

A.2.2 SCCID_FIFLAGS

This option affects how an input file's internal format (application type) is identified when the file is first opened by the Oracle Outside In technology. When the extended test flag is in effect, and an input file is identified as being either 7-bit ASCII, EBCDIC, or Unicode, the file's contents will be interpreted as such by the viewing process.

The extended test is optional because it requires extra processing and cannot guarantee complete accuracy (which would require the inspection of every single byte in a file to eliminate false positives.)

Scope

Global

Data Type

VTDWORD

Data

One of the following values:

- SCCVW_FI_NORMAL: This is the default value. When this is set, standard file identification behavior occurs.
- SCCVW_FI_EXTENDEDTEST: If set, the File Identification code will run an extended test on all files that are not identified.

Default

- SCCUT_FI_NORMAL

A.2.3 SCCID_FORMATFLAGS

This option allows the developer to set flags that span multiple Oracle Outside In products.

Scope

Local

Data Type

VTDWORD

Data

- SCCOPT_FLAGS_ISODATETIMES: 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

Default

0: All flags turned off

A.2.4 SCCID_SYSTEMFLAGS

This option controls a number of miscellaneous interactions between the developer and the Outside In Technology.

Handle Type

VTHDOC

Scope

Local

Data Type

VTDWORD

Data

- SCCVW_SYSTEM_UNICODE: This flag causes the strings in SCCDATREENODE to be returned in Unicode.

Default

0

A.2.5 SCCID_LOTUSNOTESDIRECTORY

This option 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.

Handle Types

NULL

Scope

Global

Data Type

VTLPBYTE

Data

A path to the Lotus Notes directory.

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.

A.2.6 SCCID_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 enables parsing of the XMP data 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).

Scope

Local

Data Type

VTBOOL

Data

- TRUE: This setting enables parsing XMP.
- FALSE: This setting disables parsing XMP.

Default

FALSE

A.2.7 SCCID_PDF_FILTER_MAX_EMBEDDED_OBJECTS

PDF files sometimes have a very large number of embedded objects. This option allows the user to limit the number of embedded objects that are produced in a PDF file. Setting this option to 0 means to produce an unlimited amount of embedded objects.

Scope

Local

Data Type

VTDWORD

Data

The maximum number of embedded objects to produce in PDF output.

Default

0

A.2.8 SCCID_PDF_FILTER_MAX_VECTOR_PATHS

PDF files sometimes have a very large number of vector paths. This option allows the user to limit the number of vector paths that are produced in a PDF file. Setting this option to 0 means to produce an unlimited amount of vector paths.

Scope

Local

Data Type

VTDWORD

Data

The maximum number of vector paths to produce in PDF output.

Default

0

A.2.9 SCCID_PDF_FILTER_REORDER_BIDI

This option controls whether or not the PDF filter will attempt to reorder bidirectional text runs so that the output is in standard logical order as used by the Unicode 2.0 and later specification. This additional processing will result in slower filter performance according to the amount of bidirectional data in the file.

Scope

Global

Data Type

VTDWORD

Data

- SCCUT_FILTER_STANDARD_BIDI
- SCCUT_FILTER_REORDERED_BIDI

Default

SCCUT_FILTER_STANDARD_BIDI

A.2.10 SCCID_PDF_FILTER_WORD_DELIM_FRACTION

This option controls the spacing threshold in PDF input documents. Most PDF documents do not have an explicit character denoting a word break. The PDF filter calculates the distance between two character to determine if they are part of the

same word or if there should be a word break inserted. The space between characters is compared to the length of the space character in the current font multiplied by this fraction. If the space between characters is larger, then a word break character is inserted into the text stream. Otherwise, the characters are considered to be part of the same word and no word break is inserted.

Scope

Local

Data Type

VTFLOAT

Data

A fraction representing the percentage of the space character used to trigger a word break. Valid values are 0<value<=2.

Default

0.85

A.2.11 SCCID_TIMEZONE

This option allows the user to define an offset to GMT that will be applied during date formatting, allowing date values to be displayed in a selectable time zone. 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

Data Type

VTLONG

Data

Integer parameter from -96 to 96, representing 15-minute offsets from GMT. To query the operating system for the time zone set on the machine, specify SCC_TIMEZONE_USENATIVE.

Default

- 0: GMT time

A.2.12 SCCID_HTML_COND_COMMENT_MODE

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

Handle Type

VTHDOC

Scope

Local

Data Type

VTDWORD

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

A.3 Compression

The following information pertains to compression.

A.3.1 SCCID_FILTERJPG

This option can disable access to any files using JPEG compression, such as JPG graphic files or TIFF files using JPEG compression, or files with embedded JPEG graphics. Attempts to read such files when this option is enabled will fail and return the error SCCERR_UNSUPPORTEDCOMPRESSION if the entire file is JPEG compressed, and grey boxes for embedded JPEG-compressed graphics Unlike many other options, this option must be set programmatically, as it is not stored or read on startup.

The following is a list of file types affected when this option is disabled:

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

Scope

Global

Data Type

VTDWORD

Data

- SCCVW_FILTER_JPG_ENABLED: Allow access to files that use JPEG compression
- SCCVW_FILTER_JPG_DISABLED: Do not allow access to files that use JPEG compression

Default

SCCVW_FILTER_JPG_ENABLED

A.3.2 SCCID_FILTERLZW

This option can disable access to any files using Lempel-Ziv-Welch (LZW) compression, such as .GIF files, .ZIP files or self-extracting archive (.EXE) files containing "shrunk" files. Attempts to read such files when this option is enabled will fail and return the error SCCERR_UNSUPPORTEDCOMPRESSION if the entire file is LZW compressed, and grey boxes for embedded LZW-compressed graphics. Unlike many other options, this option must be set programmatically, as it is not stored or read on startup.

The following is a list of file types affected when this option is disabled:

- GIF files
- TIF files using LZW compression
- PDF files that use internal LZW compression
- ZIP and self-extracting archive (.EXE) files containing "shrunk" files
- Postscript files using LZW compression

> **Note:**
>
> 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.

Scope

Global

Data Type

VTDWORD

Data

- SCCVW_FILTER_LZW_ENABLED: LZW compressed files will be read normally.
- SCCVW_FILTER_LZW_DISABLED: LZW compressed files will not be read.

Default

SCCVW_FILTER_LZW_ENABLED

A.4 Spreadsheet and Database File Rendering

The following information pertains to spreadsheets and database file rendering.

A.4.1 SCCID_DBCLIPBOARD

This option controls the format the database data takes when copied to the clipboard.

Data Type

VTDWORD

Data

One of the following values:

- SCCVW_CLIPSUBFORMAT_TABLE: In clipboard formats that support it (RTF and Ami), the database selection will be copied as a table.
- SCCVW_CLIPSUBFORMAT_TABS: The database selection will be copied using a tab stop for each field.
- SCCVW_CLIPSUBFORMAT_OPTIMIZEDTABS: The database selection will be copied using a tab stop for each field, except when a field is empty.

A.4.2 SCCID_DBDRAFTMODE

If this option is TRUE, the display of a database includes a limited set of formatting information. If FALSE, the display includes all supported formatting.

Data Type

VTBOOL

Default

FALSE

A.4.3 SCCID_DBFIELDNAMESTOCLIP

If this option is TRUE, field headings will be copied to the clipboard along with the data.

Data Type

VTBOOL

Default

TRUE

A.4.4 SCCID_DBPRINTFITTOPAGE

This option scales a spreadsheet file to a certain percent or to a page width or height. However, in an effort to preserve readability after scaling, the Viewer Technology will not shrink a database document to under approximately one-third of its original size.

It should be noted that when this option is set to SCCVW_DBPRINTFITMODE_NOMAP, the pages of the database file are printed down first and then across.

> **Note:**
>
> Any margins applied as a result of settings for the SCCOPT_DEFAULTPRINTMARGINS option will be included in any scaling that is applied to the viewed image as a result of settings for this option.

Data Type

VTDWORD

Data

One of the following values:

- SCCVW_DBPRINTFITMODE_NOMAP: This will not do any scaling of the database image. It will render in its original size onto as many pages as are required to fit the data.
- SCCVW_DBPRINTFITMODE_FITTOPAGES: This will fit the database to one page, scaling to the image width or height depending on the page size and database size.
- SCCVW_DBPRINTFITMODE_FITTOWIDTH: This will scale the database on the rendered image so it is no larger than one page wide.
- SCCVW_DBPRINTFITMODE_FITTOHEIGHT: This will scale the database on the rendered image so it is no larger than one page high.

Default

SCCVW_DBPRINTFITMODE_FITTOPAGES

A.4.5 SCCID_DBPRINTGRIDLINES

If this option is TRUE, a dotted grid is printed.

Data Type

VTBOOL

A.4.6 SCCID_DBPRINTHEADINGS

If this option is TRUE, field headings will be printed along with the data.

Data Type

VTBOOL

A.4.7 SCCID_DBSHOWGRIDLINES

If this option is `TRUE`, a dotted grid is displayed between fields.

Data Type

VTBOOL

A.4.8 SCCID_SSCLIPBOARD

This option controls the format the spreadsheet data takes when copied to the clipboard.

Data Type

VTDWORD

Data

One of the following values:

- SCCVW_CLIPSUBFORMAT_TABLE: In clipboard formats that support it (RTF & Ami), the spreadsheet selection will be copied as a table.
- SCCVW_CLIPSUBFORMAT_TABS: The spreadsheet selection will be copied using a tab stop for each cell.
- SCCVW_CLIPSUBFORMAT_OPTIMIZEDTABS: The spreadsheet selection will be copied using a tab stop for each cell, except when a cell is empty.

A.4.9 SCCID_SSDRAFTMODE

If this option is TRUE, the display of a spreadsheet includes a limited set of formatting information. If FALSE, the display includes all supported formatting.

Data Type

VTBOOL

A.4.10 SCCID_SSPRINTDIRECTION

This option controls the pattern in which the pages are printed, either across first and then down, or down first and then across.

This option is overridden when the SCCID_USEDOCPAGESETTINGS option is set to TRUE and print direction is specified in the input document.

Data Type

VTDWORD

Data

One of the following values:

- SCCVW_SSPRINTDIRECTION_ACROSS: Will specify that pages are printed across first and then down.
- SCCVW_SSPRINTDIRECTION_DOWN: Will specify that pages are printed down first and then across.

Default

SCCVW_SSPRINTDIRECTION_DOWN

A.4.11 SCCID_SSPRINTFITTOPAGE

This option requests that the spreadsheet file be fit to one page.

> **Note:**
>
> Any margins applied as a result of settings for the SCCID_DEFAULTPRINTMARGINS option will be included in any scaling that is applied to the output image as a result of settings for this option.

This option is overridden when the SCCID_USEDOCPAGESETTINGS option is set to TRUE and fitting the page to the printer's image limits is specified in the input document.

Data Type

VTDWORD

Data

One of the following values:

- SCCVW_SSPRINTFITMODE_NOMAP: No scaling is performed on the spreadsheet image. It will render in its original size onto as many pages as are required to fit the data.
- SCCVW_SSPRINTFITMODE_FITTOPAGES: Will scale the spreadsheet in the rendered image to fit to the number of pages specified in the SCCOPT_SSPRINTSCALEXHIGH and SCCOPT_SSPRINTSCALEXWIDE options. Since aspect ratio is maintained, the lesser of the two dimensions (width or height) will determine the scale factor. Note that if either SCCOPT_SSPRINTSCALEXHIGH or SCCOPT_SSPRINTSCALEXWIDE is set to 0, the value in the other option will be nullified.
- SCCVW_SSPRINTFITMODE_FITTOWIDTH: Will scale the spreadsheet in the rendered image so it is no larger than one page wide.
- SCCVW_SSPRINTFITMODE_FITTOHEIGHT: Will scale the spreadsheet in the rendered image so it is no larger than one page high.
- SCCVW_SSPRINTFITMODE_SCALE: Will scale the spreadsheet in the rendered image using the scale value stored in the SCCOPT_SSPRINTSCALEPERCENT option.

Default

- SCCVW_SSPRINTFITMODE_SCALE: Scales the rendered image of the spreadsheet using the scale value stored in the SCCOPT_SSPRINTSCALEPERCENT option (which is 100 by default).

A.4.12 SCCID_SSPRINTGRIDLINES

If this option is TRUE, a dotted grid is printed between cells.

This option is overridden when the SCCID_USEDOCPAGESETTINGS option is set to TRUE and printing grid lines between cells is specified in the input document.

Data Type

VTBOOL

A.4.13 SCCID_SSPRINTHEADINGS

If this option is TRUE, row and column headings will be printed along with the data.

This option is overridden when the SCCID_USEDOCPAGESETTINGS option is set to TRUE and printing column and row headers is specified in the input document.

Data Type

VTBOOL

A.4.14 SCCID_SSPRINTSCALEPERCENT

This option will scale spreadsheet pages by the percentage specified. The option has no effect unless the SCCID_SSPRINTFITTOPAGE option is set to SCCVW_SSPRINTFITMODE_SCALE.

This option must take a value between 1 and 100. If any value outside of this range is used, the option will be ignored.

Data Type

VTDWORD

A.4.15 SCCID_SSPRINTSCALEXHIGH

This option will fit the spreadsheet image to the number of vertical pages specified. The setting for this option will have no effect unless the SCCID_SSPRINTFITTOPAGE option is set to SCCVW_SSPRINTFITMODE_FITTOPAGES.

Data Type

VTDWORD

A.4.16 SCCID_SSPRINTSCALEXWIDE

This option will fit the spreadsheet image to the number of horizontal pages specified. The setting for this option will have no effect unless the SCCID_SSPRINTFITTOPAGE option is set to SCCVW_SSPRINTFITMODE_FITTOPAGES.

Data Type

VTDWORD

A.4.17 SCCID_FILTERNOBLANK

If this option is TRUE, blank spreadsheet pages will not be produced when printing a file or rendering it with the SCCVW_DRAWPAGE message.

Data Type

VTBOOL

A.4.18 SCCID_SSSHOWGRIDLINES

If this option is TRUE, a dotted grid is displayed between cells.

Data Type

VTBOOL

A.4.19 SCCID_SSSHOWHIDDENCELLS

This option lets you determine whether or not to show hidden rows or columns when rendering spreadsheets. This is a BOOLEAN option that will leave the data hidden when it is FALSE, and show all hidden rows and columns when it is TRUE, displayed using the default row width or default column height.

Data Type

VTBOOL

Default

FALSE

A.5 Graphics File Rendering

The following information pertains to graphics file rendering.

A.5.1 SCCID_ANTIALIAS

This option determines the way the viewer stretches bitmaps. Antialiased bitmaps may appear smoother and can be easier to read, but rendering when this option is set will require additional processing time.

Data Type

VTDWORD

Data

One of the following values:

- SCCVW_ANTIALIAS_OFF: Do not antialias bitmaps.
- SCCVW_ANTIALIAS_ALL: Antialias all bitmaps. Default.

A.5.2 SCCID_BMPDITHER

This option tells the bitmap display engine to use a dithering algorithm on bitmaps with higher color resolution than the screen in order to get better color display. It can have the value SCCVW_DITHER_ON or SCCVW_DITHER_OFF.

Data Type

VTBOOL

A.5.3 SCCID_BMPDITHERAVAILABLE

This option is a read only option that indicates the bitmap display engine's ability to dither the current bitmap. For instance dithering is not possible or necessary for a 16-color bitmap on a 256-color display, but it is possible for a 256-color bitmap on a 16-color display.

Data Type

VTBOOL

A.5.4 SCCID_BMPFITMODE

This option sets the way the bitmap display engine stretches or shrinks a bitmap in relation to the size of the view window.

Data Type

VTDWORD

Data

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_ORIGINAL: The bitmap is displayed bit for bit on the screen, the size of the window has no effect.
- SCCVW_FITMODE_WINDOW: The bitmap will be stretched to fill as much of the window as possible while maintaining its proper aspect ratio.

- SCCVW_FITMODE_WINDOWHEIGHT: The bitmap will be stretched so its full height fits in the height of the window. Depending on the bitmap, 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. Depending on the bitmap, its full height may or may not fit inside the window.
- SCCVW_FITMODE_IMAGESIZE: Scale to image size.
- SCCVW_FITMODE_PIXELS: The size of the window has no effect.
- SCCVW_FITMODE_STRETCHWINDOW: The image will be stretched to fill the window. The image's aspect ratio is NOT maintained.

A.5.5 SCCID_BMPPRINTASPECT

This option indicates how the bitmap will be stretched when printed.

Data Type

VTDWORD

Data

One of the following values:

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

A.5.5.1 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)

A.5.6 SCCID_BMPPRINTBORDER

If set to TRUE, a 1-pixel border will be printed around bitmaps.

Data Type

VTBOOL

A.5.7 SCCID_BMPROTATION

This option indicates how bitmaps should be rotated.

Data Type

VTDWORD

Data

One of the following values:

- SCCVW_ROTATION_NONE: Do not rotate the bitmap.
- SCCVW_ROTATION_90: Rotate the bitmap 90 degrees clockwise.
- SCCVW_ROTATION_180: Rotate the bitmap 180 degrees clockwise.
- SCCVW_ROTATION_270: Rotate the bitmap 270 degrees clockwise.

These rotations are absolute from the initial state of the graphic.

A.5.8 SCCID_BMPZOOM

This option indicates the X and Y ratios used to stretch/shrink the original image. For instance, setting the X ratio to 1 and the Y ratio to 2 displays the image pixel for pixel horizontally, but two display pixels for every one-bitmap pixel in the vertical direction.

Data Type

SCCVWIMAGEZOOM structure

SCCVWIMAGEZOOM Structure

This structure is used in the SCCID_BMPZOOM and SCCID_VECZOOM options.

SCCVWIMAGEZOOM is a C data structure defined in sccvw.h as follows.

```
typedef struct
   {
   VTDWORD   dwXNu;
   VTDWORD   dwXDe;
   VTDWORD   dwYNu;
   VTDWORD   dwYDe;
   } SCCVWIMAGEZOOM;
```

The image is zoomed in the X (horizontal) direction by a factor of dwXNu/dwXDe. The image is zoomed in the Y (vertical) direction by a factor of dwYNu/dwYDe.

A.5.9 SCCID_BMPZOOMEVENT

This option acts like an event. Normally, its value is SCCVW_ZOOM_NOP.

Data Type

VTDWORD

Data

One of the following values:

- SCCVW_ZOOM_IN: Zooms in (more magnification).
- SCCVW_ZOOM_OUT: Zooms out (less magnification).
- SCCVW_ZOOM_SELECTION: Zooms to the current selection.
- SCCVW_ZOOM_RESET: Restores the display to its original state (based on the value SCCID_BMPFITMODE).

A.5.10 SCCID_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 SCCID_VECFITMODE, and a bitmap file will be initialized to the defined SCCID_BMPFITMODE and rotation will be reset to SCCVW_ROTATION_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 SCCID_VECZOOM, and new bitmap sections will be initialized to the current SCCID_BMPZOOM and SCCID_BMPROTATION.

Data Type

VTBOOL

Default Value

TRUE

A.5.11 SCCID_VECFITMODE

This option sets the way the vector display engine stretches or shrinks an image in relation to the size of the view window.

Data Type

VTDWORD

Data

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_ORIGINAL: The image is displayed one bit on the screen for every unit in the images coordinate system, 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 image will be stretched so its full width fits in the width of the window. Depending on the image, its full height may or may not fit inside the window.
- SCCVW_FITMODE_STRETCHWINDOW: The image will be stretched to fill the window. The images aspect ratio is *not* maintained.

A.5.12 SCCID_VECPRINTASPECT

This option indicates how the vector image will be stretched when printed.

Data Type

VTDWORD

Data

One of the following values:

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

A.5.13 SCCID_VECPRINTBACKGROUND

If TRUE, the background of a vector image will be printed. If it is FALSE the background will not be printed.

Data Type

VTBOOL

A.5.14 SCCID_VECPRINTBORDER

If TRUE, a one-pixel border will be printed around vector images.

Data Type

VTBOOL

A.5.15 SCCID_VECSHOWBACKGROUND

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

Data Type

VTBOOL

A.5.16 SCCID_VECSHOWFULLSCREEN

This option acts like an event. Normally its value is FALSE, but when set to TRUE the image will be displayed using the entire screen until the user presses ESC.

The default value of this option has no meaning.

Data Type

VTBOOL

A.5.17 SCCID_VECZOOM

This option indicates the X and Y ratios used to stretch/shrink the original image. For instance, setting the X ratio to 1 and the Y ratio to 2 displays the image pixel for pixel horizontally but 2 display pixels for every one-bitmap pixel in the vertical direction.

Data Type

SCCVWIMAGEZOOM structure

SCCVWIMAGEZOOM Structure

This structure is used in the SCCID_BMPZOOM and SCCID_VECZOOM options.

SCCVWIMAGEZOOM is a C data structure defined in sccvw.h as follows:

```
typedef struct
   {
   VTDWORD   dwXNu;
   VTDWORD   dwXDe;
   VTDWORD   dwYNu;
   VTDWORD   dwYDe;
   } SCCVWIMAGEZOOM;
```

The image is zoomed in the X (horizontal) direction by a factor of dwXNu/dwXDe.

The image is zoomed in the Y (vertical) direction by a factor of dwYNu/dwYDe.

A.5.18 SCCID_VECZOOMEVENT

This option acts like an event. Normally, its value is SCCVW_ZOOM_NOP.

Data Type

VTDWORD

Data

One of the following values:

- SCCVW_ZOOM_IN: Zooms in (more magnification).
- SCCVW_ZOOM_OUT: Zooms out (less magnification).
- SCCVW_ZOOM_SELECTION: Zooms to the current selection.
- SCCVW_ZOOM_RESET: Restores the display to its original state (based on the value SCCID_VECFITMODE).

A.6 Page Rendering

The following information pertains to page rendering.

A.6.1 SCCID_DEFAULTPRINTMARGINS

This option specifies the top, left, bottom and right margins in twips from the edges of the page. For instance, setting all the values to 1440 creates a 1-inch margin on all sides. Page margins will only be applied when formatting word processing, database and spreadsheet files.

Please note all margins are applied before scaling with the SCCID_SSPRINTFITTOPAGE or SCCID_SSPRINTFITTOPAGE options.

This option is overridden when the SCCID_USEDOCPAGESETTINGS option is set to TRUE and print margins are specified in the input document.

This option does not affect the output of bitmap, presentation, vector or archive files.

Data Type

The SCCVWPRINTMARGINS structure.

SCCVWPRINTMARGINS Structure

This structure is used by the SCCID_DEFAULTPRINTMARGINS option to specify margin settings.

SCCVWPRINTMARGINS is a C data structure defined in sccvw.h as follows:

```
typedef struct SCCVWPRINTMARGINStag
   {
   VTDWORD  dwTop;
   VTDWORD  dwBottom;
   VTDWORD  dwLeft;
   VTDWORD  dwRight;
} SCCVWPRINTMARGINS,  * PSCCVWPRINTMARGINS;
```

Parameters

- dwTop: Margin from the top edge of the page (in twips). Default is 1 inch.

- dwBottom: Margin from the bottom edge of the page (in twips). Default is 1 inch.
- dwLeft: Margin from the left edge of the page (in twips). Default is 1 inch.
- dwRight: Margin from the right edge of the page (in twips). Default is 1 inch.

A.6.2 SCCID_PRINTENDPAGE

This option indicates the page that printing should end on. It is only valid if the option SCCID_WHATTOPRINT has the value SCCVW_PRINT_PAGERANGE.

> **Note:**
>
> Page range settings are one-based and inclusive. Therefore, specifying a range with SCCID_PRINTENDPAGE equal to 5 and SCCID_PRINTSTARTPAGE equal to 3 would print any of the three pages that follow, if they exist: 3, 4 and 5.

Data Type

VTDWORD

Default

- 0: The last page at the end of the document.

A.6.3 SCCID_PRINTSTARTPAGE

This option indicates the page printing should start on. It is only valid if the option SCCID_WHATTOPRINT has the value SCCVW_PRINT_PAGERANGE.

> **Note:**
>
> Page range settings are one-based and inclusive. Therefore, specifying a range with SCCID_PRINTENDPAGE equal to 5 and SCCID_PRINTSTARTPAGE equal to 3 would print any of the three pages that follow, if they exist: 3, 4 and 5.

Data Type

VTDWORD

A.6.4 SCCID_USEDOCPAGESETTINGS

This option is used to select the document's page layout information when printing.

If TRUE the document's native (or author selected) page margins, paper size, page scaling and page orientation are used when available from the filter.

The values of the SCCID_DEFAULTPRINTMARGINS, SCCID_SSPRINTGRIDLINES, SCCID_SSPRINTHEADINGS, SCCID_SSPRINTDIRECTION, and SCCID_SSPRINTFITTOPAGE options are overridden if this option is set to TRUE and the properties associated with those options are specified in the input document. Additionally,

print area and page breaks in spreadsheet documents are ignored unless this option is set to TRUE.

If FALSE, the printer page margins, paper size, orientation and scaling are set to specific values rather than those in the native document. The page size is forced to 8 1/2" x 11" in portrait orientation, but this may be changed by setting the SCCOPT_GRAPHIC_HEIGHT and/or SCCOPT_GRAPHIC_WIDTH options. The margins are forced 1" all around, but may be changed by setting the SCCID_DEFAULTPRINTMARGINS option. The scaling for the document will be set to 100%, although this may be changed by setting any of the various scaling options.

Data Type

VTBOOL

A.6.5 SCCID_WHATTOPRINT

> **Note:**
>
> This option is only valid on the Windows platforms.

This option indicates whether the whole file or a selected range of pages should be printed when the SCCVW_PRINT message is sent.

Data Type

VTDWORD

Data

One of the following values:

- SCCVW_PRINT_PAGERANGE: The pages in the one-based, inclusive range from SCCOPT_PRINTSTARTPAGE to SCCOPT_PRINTENDPAGE will be printed.
- SCCVW_PRINT_ALLPAGES: The entire document will be printed.
- SCCVW_PRINT_SELECTION: Just the selected area will be printed.
- SCCVW_PRINT_CURSECTION: The current section will be printed.

Default

SCCVW_PRINT_ALLPAGES

A.7 Word Processor File Rendering

The following information pertains to word processor file rendering.

A.7.1 SCCID_WPDISABLEEMAILHEADER

Enables or disables the email view. If disabled, the Normal and Draft viewing modes will not display email headers. If enabled, the email header will be shown for formats identified as email.

Data Type

VTBOOL

Default

0 (email display feature enabled)

A.7.2 SCCID_WPDISPLAYMODE / SCCID_HTMLDISPLAYMODE / SCCID_EMAILDISPLAYMODE

Indicates how the word processor display engine displays documents. Each of these options is applied to different types of Word Processing documents. HTML files are displayed in the mode indicated by SCCID_HTMLDISPLAYMODE, email files are displayed in the mode indicated by SCCID_EMAILDISPLAYMODE, and all other WP files are displayed in the mode specified by SCCID_WPDISPLAYMODE.

Data Type

VTDWORD

Data

One of the following values:

- SCCVW_WPMODE_DRAFT: Display using only a single font and size (SCCID_DEFAULTDISPLAYFONT), do not display embedded graphics, do not display graphic or table borders, wrap the text to the size of the view window.
- SCCVW_WPMODE_NORMAL: Display all supported formatting, wrap the text to the size of the view window.
- SCCVW_WPMODE_PREVIEW: Display all supported formatting, wrap the text as it will be printed.
- SCCVW_WPMODE_WEBLAYOUT: Display all supported formatting, wrap the text as it would appear in a browser.

A.7.3 SCCID_WPFITMODE / SCCID_HTMLFITMODE / SCCID_EMAILFITMODE

Controls the size of word processor pages when using preview or weblayout mode.

Data Type

VTDWORD

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.

A.7.4 SCCID_WPEMAILHEADEROUTPUT

This option controls the rendering of headers for email and associated file types (calender entries, contacts, appointments, etc.). For finer control over these headers, see SCCID_MAILHEADERVISIBLE and SCCID_MAILHEADERHIDDEN. It should be noted that setting this option will reset any fields specified with the SCCID_MAILHEADERVISIBLE or SCCID_MAILHEADERHIDDEN options.

Scope

Global

Data Type

VTDWORD

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_EMAILHEADERNONE: Displays no email header fields.
- SCCUT_WP_EMAILHEADERALL: Displays all available email headers.

Default

SCCUT_WP_EMAILHEADERSTANDARD

A.7.5 SCCID_MAILHEADERVISIBLE

Along with SCCID_MAILHEADERHIDDEN, these options exist to allow the developer fine-grained control over what email headers are rendered. These options modify which email headers are displayed, and are based on the most recent setting of SCCID_WPEMAILHEADEROUTPUT. To implement a fully customized set of email headers for display, your code should first set the SCCID_WPEMAILHEADEROUTPUT option to select a baseline set of headers, then use these options to selectively add or remove headers from that set.

Setting a header to be visible means that it will be rendered when that header is found in a document of the appropriate type. Selected headers that are not present in the input file will not have any corresponding output created for them (no 'empty' headers will be created). Setting a header to be hidden means that it will not be rendered for the document types specified.

Scope

Global

Data Type

SCCUTEMAILHEADERINFO structure

SCCUTEMAILHEADERINFO structure

This structure is used by the SCCID_WPMAILHEADERVISIBLE/ SCCID_WPMAILHEADERHIDDEN options to specify the headers to show or hide.

```
typedef struct SCCUTEMAILHEADERINFOtag
{
    VTDWORD    dwHeaderID;
    VTDWORD    dwSubtypeID;
    VTWORD     wsMimeHeaderName[SCCUT_MAIL_NAMELENGTH];
    VTWORD     wsMimeHeaderLabel[SCCUT_MAIL_NAMELENGTH];
} SCCUTEMAILHEADERINFO, *PSCCUTEMAILHEADERINFO;
```

Parameters:

- dwHeaderID

  Either the ID of a predefined email header field, found in sccca.h (for example SCCCA_MAIL_TO), or an identifer between NONSTANDARD_HEADER_ID_BASE and NONSTANDARD_HEADER_ID_TOP for tracking a user-defined header.

- dwSubTypeID

  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

- wsMimeHeaderName

  A Unicode string containing the value of a user-specified MIME header name. This value is only used when the dwHeaderId field contains a user-defined ID value between NONSTANDARD_HEADER_ID_BASE and NONSTANDARD_HEADER_ID_TOP.

- wsMimeHeaderLabel

  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.

Default

Not used

A.7.6 SCCID_MAILHEADERHIDDEN

Along with SCCID_MAILHEADERVISIBLE, these options exist to allow the developer fine-grained control over what email headers are rendered. These options modify which email headers are displayed, and are based on the most recent setting of SCCID_WPEMAILHEADEROUTPUT. To implement a fully customized set of email headers for display, your code should first set the SCCID_WPEMAILHEADEROUTPUT option to select a baseline set of headers, then use these options to selectively add or remove headers from that set.

Setting a header to be visible means that it will be rendered when that header is found in a document of the appropriate type. Selected headers that are not present in the input file will not have any corresponding output created for them (no 'empty' headers will be created). Setting a header to be hidden means that it will not be rendered for the document types specified.

Scope

Global

Data Type

See SCCUTEMAILHEADERINFO structure under SCCID_MAILHEADERVISIBLE

Default

Not used

A.7.7 SCCID_WPWRAPTOWINDOW

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 in 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.

Data Type

VTBOOL

A.8 Archive Rendering

The following information pertains to archive rendering.

A.8.1 SCCID_ARCOUTPUTPATH

This option string contains the path to save files in.

Data Type

C String

A.8.2 SCCID_ARCRENAME

This option allows prompting the user for input when nodes that are being extracted need to be renamed because the node name already exists, the node name contains illegal characters, or both. The default behavior is for neither of the possible values to be set.

Data Type

VTDWORD

Data

The option can have the following values, OR-ed together if necessary:

- SCCVW_ARCHIVE_PROMPTONCOLLISION: If this value is set, a dialog box will appear when two files being extracted have the same name. It will prompt the user to either skip extracting the current file or to overwrite the existing file. If this value is not set, the file will be renamed with a unique number appended to the end.
- SCCVW_ARCHIVE_PROMPTONILLEGALCHAR: If this value is set, a dialog box will appear when a file being extracted has illegal characters in it. It will prompt the user to rename the file using only legal characters. The list of illegal characters are operating system dependent (Windows: ? [ ] / \ = + < > : ;) , and are all characters that cannot be included in file names or file paths. If this value is not set, all illegal characters will be stripped out of the file name.

A.8.3 SCCID_ARCSAVEEVENT

This option acts like an event.

Data Type

VTDWORD

Data

Normally, the value for this option is SCCVW_ARCHIVE_SAVENOP, but it may be set to either of the following values, as well:

- SCCVW_ARCHIVE_SAVESELECTION: Allows the user to save the selected files to the directory of their choice.
- SCCVW_ARCHIVE_SAVEALL: Allows the user to save all the files to the directory of their choice.
- SCCVW_ARCHIVE_SAVETHIS: Allows the user to save the current file to the directory of their choice.

A.8.4 SCCID_ARCSORTORDER

This option sets which attribute to use and the order in which files are displayed in the archive display engine.

Data Type

VTDWORD

Data

One of the following values. Any of these values can be OR-ed with SCCVW_SORT_DESCENDING to force the sorting of the selected attribute to descending order (ascending order is the default when this flag is not set):

- SCCVW_SORT_NONE: Display the files in the order they appear in the archive.
- SCCVW_SORT_NAME: Display the files sorted by their names.
- SCCVW_SORT_SIZE: Display the files sorted by their sizes.
- SCCVW_SORT_DATE: Display the files sorted by their dates/times.

A.8.5 SCCID_ARCFULLPATH

In the Viewer and rendering products, this option tells the archive display engine to show the full path to a node in the szNode field in response to a SCCVW_GETTREENODE message. It also causes the name fields in DAGetTreeRecord and DAGetObjectInfo to contain the full path instead of just the archive node name.

Data Type

VTBOOL

Data

- TRUE: Display the full path.
- FALSE: Do not display the path.

Default

FALSE

A.9 Data Rendering

The following information pertains to data rendering.

A.9.1 SCCID_DAYNAMES

This option sets the full and abbreviated names used to format the days of the week when formatting dates. The default values are retrieved from the localization resource for each platform.

Data Type

SCCVWDAYNAMES structure

SCCVWDAYNAMES Structure

This structure is used by the SCCID_DAYNAMES option to specify the format of day names.

SCCVWDAYNAMES is a C data structure defined in sccvw.h as follows:

```
typedef struct SCCVWDAYNAMEStag
   {
   VTTCHAR  aFullNames[7][16];
   VTTCHAR  aAbbrevNames[7][16];
   } SCCVWDAYNAMES,  * PSCCVWDAYNAMES;
```

Parameters

- aFullNames: An array of character strings which identify the full name for each day of the week, starting with Sunday. This option is used when formatting dates which use the full name of the day of the week.
- aAbbrevNames: An array of character strings, which identify the abbreviated name for each day of the week, starting with Sunday. This option is used when formatting dates that use the abbreviated name of the day of the week.

A.9.2 SCCID_IGNORE_PASSWORD

This option can disable the password verification of files where the contents can be processed without validation of the password. If this option is not set, the filter should prompt for a password if it handles password-protected files.

> **Note:**
>
> As of Release 8.4.0, only the PST and MDB Filters support this option.

Scope

Global

Data Type

VTBOOL

Data

- TRUE: Ignore validation of the password.
- FALSE: Prompt for the password.

Default

FALSE

A.9.3 SCCID_INTLFLAGS

This option is used to set flags related to international support. The default value is retrieved from the operating system for the Windows platform, and defaults to SCCVW_ENGLISHUNITS OR-ed with SCCVW_12HOURTIME on all other platforms.

Data Type

VTDWORD

Data

One of the following values:

- SCCVW_ENGLISHUNITS: If set, the print margins are displayed to the user 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.

A.9.4 SCCID_MONTHNAMES

This option sets the full and abbreviated names used for formatting dates. The default values are retrieved from the localization resource for each platform.

Data Type

SCCVWMONTHNAMES structure

SCCVWMONTHNAMES Structure

This structure is used to set the SCCID_MONTHNAMES option.

SCCVWMONTHNAMES is a C data structure defined in sccvw.h as follows:

```
typedef struct
   {
   VTTCHAR   aFullNames[13][16];
   VTTCHAR   aAbbrevNames[13][16];
   } SCCVWMONTHNAMES;
```

Parameters

- aFullNames: An array of character strings which identify the full name for each month of the year starting with January. This option is used when formatting dates which use the full name of the month of the year.
- aAbbrevNames: An array of character strings, which identify the abbreviated name for each month of the year, starting with January. This option is used when formatting dates that use the abbreviated name of the month of the year.

A.9.5 SCCID_NUMBERFORMAT

This option is used to control the formatting of numbers. It is useful for setting environment dependent variables related to international support. The default values are retrieved from the operating system for the Windows platform, and are set to logical U.S. defaults on all other platforms.

Data Type

SCCVWNUMBERFORMAT and SCCVWNUMBERFORMAT775 structures

SCCVWNUMBERFORMAT775 and SCCVWNUMBERFORMAT Structures

These structures are used to set the SCCID_NUMBERFORMAT option. The fields of the structures allow the developer to control variables related to international support. Please note that the SCCVWNUMBERFORMAT775 structure always assumes 2-digit year data, whereas the SCCVWNUMBERFORMAT structure allows for both 2- and 4-digital year data.

These are C data structures defined in sccvw.h as follows:

```
typedef struct SCCVWNUMBERFORMAT775tag
   {
   VTTCHAR   cDecimalSep;
   VTTCHAR   cThousandSep;
   VTTCHAR   cDateSep;
   VTTCHAR   cTimeSep;
   VTTCHAR   szCurrencySymbol[8];
   VTTCHAR   szAM[8];
   VTTCHAR   szPM[8];
   VTDWORD   dwNumBytesAM;
   VTDWORD   dwNumBytesPM;
   VTWORD    wCurrencyPosition;
   VTWORD    wShortDateOrder;
   } SCCVWNUMBERFORMAT775,  * PSCCVWNUMBERFORMAT775;


typedef struct SCCVWNUMBERFORMATtag
   {
   VTTCHAR   cDecimalSep;
   VTTCHAR   cThousandSep;
   VTTCHAR   cDateSep;
   VTTCHAR   cTimeSep;
   VTTCHAR   szCurrencySymbol[8];
   VTTCHAR   szAM[8];
   VTTCHAR   szPM[8];
   VTDWORD    dwNumBytesAM
   VTDWORD   dwNumBytesPM;
   VTWORD    wCurrencyPosition
   VTWORD    wShortDateOrder;
```

```
VTWORD    wShortDateYearDigits;
VTWORD    wShortDateMonthDigits;
VTWORD    wShortDateDayDigits;
VTWORD    wShortDateFlags;
} SCCVWNUMBERFORMAT,  * PSCCVWNUMBERFORMAT;
```

Parameters

- cDecimalSep: The character used for the decimal separator when formatting currency.
- cThousandSep: The character used for the thousands separator when formatting currency.
- cDateSep: The character used to separate years, months, and days when formatting dates. This option only works on variable formats. For example, only one of the several date formats in Microsoft Excel is variable.
- cTimeSep: The character used to separate hours, minutes, and seconds when formatting times. This option only works on variable formats. For example, only one of the several time formats in Microsoft Excel is variable.
- szCurrencySymbol: The string used for the currency symbol when formatting currency.
- szAM: The string used to indicate "AM" when formatting times.
- szPM: The string used to indicate "PM" when formatting times.
- dwNumBytesAM: Number of bytes of the string stored in `szAM`.
- dwNumBytesPM: Number of bytes of the string stored in `szPM`.
- wCurrencyPosition: Flags that indicate the positioning of the currency symbol when formatting currency. Only six specific filters are supported: SOC6, WG2, WK4, WK6, WPW, and VISO.
  - SCCVW_CURRENCY_LEADS: The currency symbol is placed before the amount.
  - SCCVW_CURRENCY_TRAILS: The currency symbol is placed after the amount.
  - SCCVW_CURRENCY_SPACE: A space is placed between the currency and the amount.
  - SCCVW_CURRENCY_NOSPACE: A space is not placed between the currency and the amount.
- wShortDateOrder: Indicates the order used when formatting short dates (numeric dates). This option only works on variable formats. For example, only one of the several date formats in Microsoft Excel is variable. One of the following:
  - SCCVW_DATEORDER_MDY: Month, Day, Year
  - SCCVW_DATEORDER_DMY: Day, Month, Year
  - SCCVW_DATEORDER_YMD: Year, Month, Date
- wShortDateYearDigits: This parameter is specific to the SCCVWNUMBERFORMAT structure. This is the number of digits in the year as specified by the Windows registry entry sShortDate. This option only works on variable formats. For example, only one of the several date formats in Microsoft Excel is variable.

- wShortDateMonthDigits: This parameter is specific to the SCCVWNUMBERFORMAT structure. This is the number of digits in the month as specified by the Windows registry entry sShortDate.
- wShortDateDayDigits: This parameter is specific to the SCCVWNUMBERFORMAT structure. This is the number of digits in the day as specified by the Windows registry entry sShortDate.
- wShortDateFlags: This parameter is specific to the SCCVWNUMBERFORMAT structure. It is reserved for internal use.

A.9.6 SCCID_REORDERMETHOD

This option controls how the viewer reorders bidirectional text.

Data Type

VTDWORD

Data

One of the following values:

- SCCUT_REORDER_UNICODE_OFF: This disables any processing for bidirectional characters. This option is the default.
- SCCUT_REORDER_UNICODE_LTOR: Characters displayed using the Unicode bidirectional algorithm assuming a base left-to-right order. Use this option to enable bidirectional rendering.
- SCCUT_REORDER_UNICODE_RTOL: Characters displayed using the Unicode bidirectional algorithm assuming a base right-to-left order. Use this option to force starting bidirectional rendering in the right-to-left order.

A.9.7 SCCID_STROKE_TEXT

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.

- If the export for the conversion is text only, text is never stroked out.
- If the export is not text only, and the drawing is perspective, text will always be stroked out (regardless of this option). This is due to the fact that in non-text only situations visual fidelity is of importance, and handling of textual objects in perspective drawings is more accurate with stroked out text. If the conversion is non-text only and the drawing is not perspective, this option determines if text should be stroked.

Note that when this option is TRUE text is non-searchable, and some special characters appear as asterisks or question marks due to limited support of characters for stroking out text.

Handle Types

VTHDOC, VTHEXPORT

Scope

Local

Data Type

VTBOOL

Default

TRUE

A.10 View Window

The following information pertains to the View Window.

A.10.1 SCCID_DEFAULTCLIPBOARDFONT

This option is the font placed on the clipboard (in clipboard formats that support this functionality) when no font is contained within the file being viewed. It is also the font placed on the clipboard for spreadsheets and databases.

Data Type

`SCCVWFONTSPEC` structure (see the structure documentation in SCCID_DEFAULTPRINTFONT)

A.10.2 SCCID_DEFAULTDISPLAYFONT

This option sets the font used when no font is contained within the file being viewed. It is also the font used in draft mode (when viewing documents) and the font used to display spreadsheets and databases.

Data Type

`SCCVWFONTSPEC` structure (see the structure documentation in SCCID_DEFAULTPRINTFONT)

A.10.3 SCCID_DIALOGFLAGS

This option controls a number of aspects of the method by which dialogs are displayed and by which dialog choices are added to menus.

Data Type

VTDWORD

Data

One or more of the following values OR-ed together:

- 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_NO3D: The dialogs available in the Viewer Technology 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. CTL3DV2.DLL is loaded dynamically (LoadLibrary) so systems without it will not cause a problem.
- SCCVW_DIALOG_NOADDDOCOPYTOMENU: Normally, the menu retrieved from SCCVW_GETDISPLAYINFO includes an item called Copy, which copies to the clipboard. This flag disables this menu item.
- SCCVW_DIALOG_NOADDDOPRINTTOMENU: Normally, the menu retrieved from SCCVW_GETDISPLAYINFO includes an item called Print..., which prints the file. This flag disables this menu item.
- SCCVW_DIALOG_NOADDOPTIONSTOMENU: Normally, the menu retrieved from SCCVW_GETDISPLAYINFO ends in a popup called Options, which has three items: Display..., Print..., and Clipboard.... These items provide access to the options dialogs available in the Viewer Technology. If this is not appropriate for your application, setting this flag will disable this extra popup.
- SCCVW_DIALOG_NOADDDISPLAYTOMENU: Setting this flag removes just the Display... item from the option menu described earlier in this bulleted list.
- SCCVW_DIALOG_NOADDPRINTTOMENU: Setting this flag removes just the Print... item from the option menu described earlier in this bulleted list.
- SCCVW_DIALOG_NOADDCLIPBOARDTOMENU: Setting this flag removes just the Clipboard... item from the option menu described earlier in this bulleted list.
- 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 the original size. This flag disables this popup menu item.
- SCCVW_DIALOG_NOADDSHOWFULLSCREEN: The dialog should not display the "Show Full Screen" menu option from the context menu.

A.10.4 SCCID_DISPLAYFONTALIAS

This option sets or gets display font aliases according to the SCCVWFONTALIAS structure.

Data Type

SCCVWFONTALIAS structure

A.10.5 SCCID_FONTSCALINGFACTOR

This option scales fonts in the view of a file by a percentage of the original size. The normal size is 100%. The valid range for this factor is from 40 to 300 with 100 as the default.

> **Note:**
>
> For word processor documents, this only affects normal and draft modes. It will not affect paragraph, page or table attributes (for example, margins). For spreadsheet and database documents, the font scaling will affect the row height and column widths. This option will not affect metafile or raster graphics views. For all other document types, only the fonts will be affected

In the context menu, the granularity of an increase/decrease of a font is 20% of the normal size.

Data Type

VTDWORD

A.10.6 SCCID_OLEFLAGS

This option is used to set flags related to OLE features enabled. This flag has meaning only on systems that have OLE 2.0 available. The default value is to have none of the flags set.

Data Type

VTDWORD

Data

- SCCVW_OLE_ENABLEDRAGDROP: If this flag is set, the viewer enables OLE drag-and-drop copying. This allows the user to click on a highlighted selection in a document and drag it to a target application that can accept it.

A.10.7 SCCID_RESOURCELIBRARYID

Selects the localization string file based on the Resource ID.

Data Type

SCCVWRESOURCEID structure

SCCVWRESOURCEID Structure

This structure is used in the SCCID_RESOURCELIBRARYID option.

SCCVWRESOURCEID is a C data structure defined in sccvw.h as follows:

```
typedef struct
   {
   VTDWORD   dwSize;
   VTBYTE    szResourceID [SCCVW_RESOURCEIDMAX];
   } SCCVWRESOURCEID;
```

Parameters

- dwSize: Must be set by the developer to sizeof(SCCVWRESOURCEID).
- szResourceID: This is a 0-5 character ID defining the new version of the library that has a unique version of the resources.

> **Note:**
>
> szResourceID is *not* your OEM ID. This ID will be appended to the platform-specific library name to uniquely identify your resource file. If no ID is defined or it is cleared out (NULL), the default library containing English resources will be used

Example

If you use "GR" for a resource file that is localized with German resources the new library must be named as follows:

- New Name: SCCLOGR.DLL
- Default Name: SCCLO.DLL

A.10.8 SCCID_SCROLLFLAGS

This option controls the display of the horizontal and vertical scroll bars. You will need to use both an HSCROLL and a VSCROLL value (from the following list) in the SCCVW_SETOPTION call.

Data Type

VTDWORD

Data

One of the following values:

- SCCVW_HSCROLL_NEVER: No horizontal scroll bar will be visible.
- SCCVW_HSCROLL_SOMETIMES: Visibility of the horizontal scroll bar is under the control of the display engines.
- SCCVW_HSCROLL_ALWAYS: The horizontal scroll bar is always visible.
- SCCVW_VSCROLL_NEVER: No vertical scroll bar will be visible.
- SCCVW_VSCROLL_SOMETIMES: Visibility of the vertical scroll bar is under the control of the display engines.
- SCCVW_VSCROLL_ALWAYS: The vertical scroll bar is always visible.

> **Note:**
>
> The display engines do not currently show/hide the scroll bars, so SOMETIMES and ALWAYS are logically equivalent. This may change in the future.

A.10.9 SCCID_SYSTEMFLAGS

This option controls a number of miscellaneous interactions between the developer and the Viewer Technology.

> **Note:**
>
> Every new window automatically has this option reset to 0. If you wish to use any of these flags they must be explicitly set for each view window after it is created.

Data Type

VTDWORD

Data

One of the following values:

- SCCVW_SYSTEM_RAWTEXT: The developer will receive SCCVW_RAWTEXTEVENT messages.
- SCCVW_SYSTEM_NOTIMER: Disables the view window's internal timer that controls background reading, caret blinking and auto scroll. If this flag is set, the developer must send SCCVW_IDLE messages to the view window.
- SCCVW_SYSTEM_NOOPTIONSSAVE: Normally, when a view window is destroyed, its current options are copied to the default options and saved. If this interaction is not acceptable for the developer's application, this flag may be set to disable it. If this flag is set, the developer may manually cause the current options to be saved into the default set by calling SCCVW_SAVEOPTIONS.
- SCCVW_SYSTEM_NOREADAHEAD: This flag disables the view window's process that reads through the rest of the file in the background while the first part of the file is being viewed. When this flag is set, the file will be read "on demand" as the user scrolls down through the document. Please note that use of this flag will cause unusual scroll bar behavior as the user scrolls down.
- SCCVW_SYSTEM_UNICODE: This flag will enable the following APIs to process Unicode: SCCVW_GETRAWTEXT, SCCVW_SEARCH, SCCVW_SEARCHDIALOG and SCCVW_GETANNOTATIONDATA.
- SCCVW_SYSTEM_TEXTOUTPRINTERDRIVERBUG, SCCVW_SYSTEM_TEXTOUTDISPLAYDRIVERBUG

  Some device drivers incorrectly implement ExtTextOutW. The fault is tied to the character width array. Turning these flags on causes the Oracle Outside In Viewer Technology to work around the driver bug. Turning these system flags on when the driver bug does not exist will cause incorrect behavior.

A.10.10 SCCID_TOCLIPBOARD

This option controls the clipboard formats that the viewer attempts to place on the clipboard in response to a SCCVW_COPYTOCLIP message.

Data Type

VTDWORD

Data

One or more of the following 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.

A.11 Printing

The following information pertains to printing.

A.11.1 SCCID_DEFAULTPRINTFONT

This option has the same behavior for printing as SCCID_DEFAULTDISPLAYFONT has for display.

SCCVWFONTSPEC Structure

This structure is used by various options to specify a font.

SCCVWFONTSPEC is a C data structure defined in sccvw.h as follows:

```
typedef struct
   {
   VTTCHAR  szFace[40];
   VTWORD   wHeight;
   VTWORD   wAttr;
   VTWORD   wType;
   } SCCVWFONTSPEC,  * LPSCCVWFONTSPEC;
```

Parameters

wAttr: The attributes of the font. SCCVW_CHARATTR_NORMAL is the default. Any of the following values can be OR-ed together:

- SCCVW_CHARATTR_NORMAL
  - SCCVW_CHARATTR_UNDERLINE
  - SCCVW_CHARATTR_ITALIC
  - SCCVW_CHARATTR_BOLD
  - SCCVW_CHARATTR_STRIKEOUT
  - SCCVW_CHARATTR_SMALLCAPS
  - SCCVW_CHARATTR_OUTLINE
  - SCCVW_CHARATTR_SHADOW
  - SCCVW_CHARATTR_CAPS
  - SCCVW_CHARATTR_SUBSCRIPT
  - SCCVW_CHARATTR_SUPERSCRIPT
  - SCCVW_CHARATTR_DUNDERLINE
  - SCCVW_CHARATTR_WORDUNDERLINE
  - SCCVW_CHARATTR_DOTUNDERLINE
  - SCCVW_CHARATTR_DASHUNDERLINE
- wType: Should be set to 0.

A.11.2 SCCID_PRINTCOLLATE

If TRUE, multiple copies of a printout will be collated. If FALSE, the copies for each page will be printed together.

Data Type

VTBOOL

A.11.3 SCCID_PRINTCOPIES

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

Data Type

VTDWORD

A.11.4 SCCID_PRINTFONTALIAS

This option sets or gets printer font aliases according to the SCCVWFONTALIAS structure.

Data Type

The SCCVWFONTALIAS structure.

SCCVWFONTALIAS Structure

This structure is used in the SCCID_PRINTFONTALIAS option.

SCCVWFONTALIAS is a C data structure defined in sccvw.h as follows:

```
typedef struct SCCVWFONTALIAS
   {
   VTDWORD   dwSize;
   VTDWORD   dwAliasID;
   VTDWORD   dwFlags;
   VTWORD    szwOriginal[SCCVW_FONTNAMEMAX];
   VTWORD    szwAlias[SCCVW_FONTNAMEMAX];
   } SCCVWFONTALIAS;
```

Parameters

- dwSize: Must be set by the developer to sizeof(SCCVWFONTALIAS).
- dwAliasID: ID of the aliasing in the current list of aliases.
- dwFlags: The usage of these flags depends on whether this structure is being used with the SCCVW_SETOPTION or SCCVW_GETOPTION message. It should be set to one of the following:
  - SCCVW_FONTALIAS_COUNT (SCCVW_GETOPTION): dwAliasID will be filled with the count of current font aliases for that device.
  - SCCVW_FONTALIAS_ALIASNAME (SCCVW_SETOPTION): The alias of szwAlias for szwOriginal will be used when szwOriginal is not available on the device. When a font alias is added to the list, this can affect the alias count. If an alias already exists for szwOriginal, the new szwAlias will replace it.

  - SCCVW_FONTALIAS_ALIASNAME (SCCVW_GETOPTION): szwAlias will be filled if there is an alias in the alias list for the font in szwOriginal on that device.
  - SCCVW_FONTALIAS_GETALIASBYID (SCCVW_GETOPTION): szwAlias and szwOriginal will be filled by the technology for the alias in the numbered slot identified by the ID.
  - SCCVW_FONTALIAS_GETALIASID (SCCVW_GETOPTION): dwAliasID will be set for the font in szwOriginal. If none exists, the dwAliasID will be 0xFFFFFFF.
  - SCCVW_FONTALIAS_REMOVEALIASBYID (SCCVW_SETOPTION): The alias in that slot will be removed if one exists. When a font alias is removed from the list, this can affect the other alias IDs.
  - SCCVW_FONTALIAS_REMOVEALIASBYNAME (SCCVW_SETOPTION): The alias for the font szwOriginal will be removed from the alias list if one exists. When a font alias is removed from the list, this can affect the other alias IDs.
  - SCCVW_FONTALIAS_REMOVEALL (SCCVW_SETOPTION): The alias list will be cleared out and the count will be zero.
  - SCCVW_FONTALIAS_USEDEFAULTS (SCCVW_SETOPTION): This clears the existing alias list and sets it to a list of default aliases that is variable by platform.

- szwOriginal: This represents the original name of a font that will be mapped when this font is not available. This name should be a Unicode string.
- szwAlias: This represents the new name of a font that will be used as a replacement for the unmapped font named in szwOriginal. This name should be a Unicode string.

Data

The technology assumes the following default mappings. The first value is the szwOriginal Value, the second is the szwAlias Value.

- Chicago = Arial
- Geneva = Arial
- New York = Times New Roman
- Helvetica = Arial
- Helv = Arial
- times = Times New Roman
- Times = Times New Roman
- Tms Roman = Times New Roman
- Symbol = Symbol
- itc zapfdingbats = Zapfdingbats
- itc zapf dingbats = Zapfdingbats

A.11.5 SCCID_PRINTHEADER

If the value is TRUE, a header containing the page number and the text specified by the SCCID_PRINTJOBNAME option will be printed at the top of each page.

> **Note:**
>
> If necessary, the top margin (SCCID_DEFAULTPRINTMARGINS) will be moved down to accommodate the header.

Data Type

VTBOOL

A.11.6 SCCID_PRINTHEADERFONT

This option is the font used when printing the header if SCCID_PRINTHEADER is TRUE.

Data Type

SCCVWFONTSPEC structure

A.11.7 SCCID_PRINTJOBNAME

This option sets the text printed in the header on each page and the job name passed to the OS (if applicable). The characters "%F" appearing in the string will be replaced by the display name of the file being viewed. For instance "My document is [%F]" might result in "My document is [TEST.DOC]."

Data Type

C string of 128 characters (data size is 128 characters, including the terminating zero)

A.12 File System

The following information pertains to file systems.

A.12.1 SCCID_IO_BUFFERSIZE

This provides three options that allow the user to adjust buffer sizes to take advantage of faster computers/more memory. This is an advanced option that casual users may ignore. This option allows users to tune memory usage to a particular target machine. The number specified is in kilobytes.

Scope

Global

Data Type

SCCBUFFEROPTIONS Structure

Data

A buffer options structure

SCCBUFFEROPTIONS Structure

```
typedef struct SCCBUFFEROPTIONStag
{
   VTDWORD dwReadBufferSize;    /* size of the I/O Read buffer
                                in KB */
   VTDWORD dwMMapBufferSize;    /* maximum size for the I/O
                                Memory Map buffer in KB */
   VTDWORD dwTempBufferSize;    /* maximum size for the memory-
                                mapped temp files in KB */
   VTDWORD dwFlags;             /* use flags */
} SCCBUFFEROPTIONS, *PSCCBUFFEROPTIONS;
```

Parameters

- dwReadBufferSize: Used to define the number of bytes that will read from disk into memory at any given time. Once the buffer has data, further file reads will proceed within the buffer until the end of the buffer is reached, at which point the buffer will again be filled from the disk. This can lead to performance improvements in many file formats, regardless of the size of the document.
- dwMMapBufferSize: Used to define a maximum size that a document can be and use a memory-mapped I/O model. In this situation, the entire file is read from disk into memory and all further I/O is performed on the data in memory. This can lead to significantly improved performance, but note that either the entire file can be read into memory, or it cannot. If both of these buffers are set, then if the file is smaller than the `dwMMapBufferSize`, the entire file will be read into memory; if not, it will be read in blocks defined by the `dwReadBufferSize`.
- dwTempBufferSize: The maximum size that a temporary file can occupy in memory before being written to disk as a physical file. Storing temporary files in memory can boost performance on archives, files that have embedded objects or attachments. If set to 0, all temporary files will be written to disk.
- dwFlags
  - SCCBUFOPT_SET_READBUFSIZE 1
  - SCCBUFOPT_SET_MMAPBUFSIZE 2
  - SCCBUFOPT_SET_TEMPBUFSIZE 4

  To set any of the three buffer sizes, set the corresponding flag while calling dwSetOption.

Default

The default settings for these options are:

- #define SCCBUFOPT_DEFAULT_READBUFSIZE 2: A 2KB read buffer.
- #define SCCBUFOPT_DEFAULT_MMAPBUFSIZE 8192: An 8MB memory-map size.
- #define SCCBUFOPT_DEFAULT_TEMPBUFSIZE 2048: A 2MB temp-file limit.

Minimum and maximum sizes for each are:

- SCCBUFOPT_MIN_READBUFSIZE 1: Read one Kbyte at a time.
- SCCBUFOPT_MIN_MMAPBUFSIZE 0: Don't use memory-mapped input.
- SCCBUFOPT_MIN_TEMPBUFSIZE 0: Don't use memory temp files

- SCCBUFOPT_MAX_READBUFSIZE 0x003fffff, SCCBUFOPT_MAX_MMAPBUFSIZE 0x003fffff, SCCBUFOPT_MAX_TEMPBUFSIZE 0x003fffff: These maximums correspond to the largest file size possible under the 4GB `DWORD` limit.

A.12.2 SCCID_TEMPDIR

From time to time, the technology needs to create one or more temporary files. This option sets the directory to be used for those files.

It is recommended that this option be set as part of a system to clean up temporary files left behind in the event of abnormal program termination. By using this option with code to delete files older than a predefined time limit, the OEM can help to ensure that the number of temporary files does not grow without limit.

> **Note:**
>
> This option will be ignored if SCCOPT_REDIRECTTEMPFILE is set.

Scope

Global

Data Type

SCCUTTEMPDIRSPEC structure

SCCUTTEMPDIRSPEC Structure

This structure is used in the SCCID_TEMPDIR option.

SCCUTTEMPDIRSPEC is a C data structure defined in sccvw.h as follows:

```
typedef struct SCCUTTEMPDIRSPEC
{
   VTDWORD   dwSize;
   VTDWORD   dwSpecType;
   VTBYTE    szTempDirName[SCCUT_FILENAMEMAX];
} SCCUTTEMPDIRSPEC,  * LPSCCUTTEMPDIRSPEC;
```

There is a limitation in the current release. dwSpecType describes the contents of szTempDirName. Together, dwSpecType and szTempDirName describe the location of the source file. The only dwSpecType values supported at this time are:

- IOTYPE_ANSIPATH: szTempDirName points to a NULL-terminated full path name using the ANSI character set and FAT 8.3 (Win16) or NTFS (Win32 and Win64) file name conventions.
- IOTYPE_UNICODEPATH: szTempDirName points to a NULL-terminated full path name using the Unicode character set and NTFS file name conventions. Note that the length of the path name is limited to SCCUT_FILENAMEMAX bytes, or (SCCUT_FILENAMEMAX / 2) double-byte Unicode characters.

Specifically not supported at this time is IOTYPE_REDIRECT.

Parameters

- dwSize: Set to sizeof(SCCUTTEMPDIRSPEC).
- dwSpecType: IOTYPE_ANSIPATH, IOTYPE_UNICODE.
- szTempDirName: The path to the directory to use for the temporary files. Note that if all SCCUT_FILENAMEMAX bytes in the buffer are filled, there will not be space left for file names.

A.12.3 SCCOPT_DOCUMENTMEMORYMODE

This option determines the maximum amount of memory that the chunker may use to store the document's data, from 4 MB to 1 GB. The more memory the chunker has available to it, the less often it needs to re-read data from the document.

Scope

Global

Data Type

VTDWORD

Parameters

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

Default

SCCDOCUMENTMEMORYMODE_LARGE (256MB)

A.12.4 SCCOPT_REDIRECTTEMPFILE

This option is set when the developer wants to use redirected IO to completely take over responsibility for the low level IO calls of the temp file.

Scope

Global (not persistent)

Data Type

VTLPVOID: pCallbackFunc

Function pointer of the redirect IO callback.

Redirect call back function:

```
typedef
{
     VTDWORD (* REDIRECTTEMPFILECALLBACKPROC)
```

```
(HIOFILE *phFile,
VTVOID *pSpec,
VTDWORD dwFileFlags);
```

There is another option to handle the temp directory, SCCOPT_TEMPDIR. Only one of these two can be set by the developer. The SCCOPT_TEMPDIR option will be ignored if SCCOPT_REDIRECTTEMPFILE is set. These files may be safely deleted when the Close function is called.