***Quick View Plus*®** *Viewer APIs (**ViewAPI**)*

Easy access to the powerful viewer technology in *Quick View Plus*

Inso Corporation
401 North Wabash Suite 600
Chicago, IL 60611
(312) 329-0700

Level 1b

Copyright © 1993 - 1995, Inso Corporation

Changes for Level 1a

Added -x option to ViewAPI method 1 (SCCVAPI.EXE), allows multiple executions of SCCVAPI.EXE to use only one window.

Changes for Level 1b

Added Visual Basic Control documentation.

1. Introduction

The ***ViewAPI*** has been developed in response to a need of developers, system integrators, IS departments and informed users to access Inso's viewer technology in a straightforward way. This specification's goals are as follows;

- Allow developers of commercial applications to include the option of file viewers (through the purchase of *Quick View Plus*), without including extensive viewer code in their shrink wrapped product.
- Allow IS departments and system integrators to easily offer viewer technology as a solution to their users needs.
- Define a specification that does not require major changes to existing applications in order to utilize the full power for *Quick View Plus* technology.
- Define a specification that allows developers to access the technology in a way that fits their particular application (EXE command line, DLL function calls or Visual Basic Control).

ViewAPI requires that the user has a copy of *Quick View Plus* installed on their machine. Developers who wish to ship Inso's viewer technology as part of their product and/or need tighter integration of the viewer technology into their product, should contact Inso's OEM Sales Manager and ask for the **SCCVIEWER Class Specification**. This is the underlying specification on which ***ViewAPI*** is built.

2. Overview

ViewAPI provides three methods of accessing the viewer technology in *Quick View Plus*.

1. Command line parameters to SCCVAPI.EXE
2. Function calls to SCCVAPI.DLL
3. Visual Basic Control through SCCVIEW.VBX

The second method is the most functional and we would like developers of commercial applications to consider this as their first choice when choosing an API.

3. Distribution and location of SCCVAPI.EXE and SCCVAPI.DLL

SCCVAPI.EXE, SCCVAPI.DLL, and SCCVIEW.VBX will be installed in the Windows directory by *Quick View Plus* (version 2.01 and above). Developers should have no reason to distribute these files with their applications.

4. Compatibility

Inso **WILL** guarantee that the APIs in any version of SCCVAPI.EXE, SCCVAPI.DLL, and SCCVIEW.VBX will be backwards compatible with older versions. This will allow new versions of *Quick View Plus* to update these files, thereby automatically adding new functionality to existing applications that support ***ViewAPI***. Inso **WILL NOT** guarantee the forwards or backwards compatibility of a given version of SCCVAPI.EXE, SCCVAPI.DLL, and SCCVIEW.VBX with newer or older versions of *Quick View Plus*. This should not be a problem since these files are only shipped with *Quick View Plus.*

5. What does *ViewAPI* do?

All ***ViewAPI*** APIs produce the same result; an independent, overlapped window containing a view of the requested file, and a tool bar that gives the user access to Launch, Print, Copy to clipboard, Search, and About.

6. *ViewAPI* method 1 - Command line parameters to SCCVAPI.EXE

SCCVAPI *filename* [*options*]

filename

Full path name of file to view

options

-p (Position)	
-p x,y,width,height	Position top/left of view window at x,y and size view window to width,height. Values are in screen coordinates. Example -p 100,100,400,200
-m (Maximize)	
-m	The view window starts maximized (-p option still valid)
-i (Iconize)	
-i	The view window starts minimized (-p option still valid)
-d (Display Name)	
-d text	Uses text in place of file name in display situations. Useful when dealing with temporary files. Example -d The Attachment would cause "The Attachment" to appear in the title bar above, instead of MAILBAG.WPG
-h (Toolbar Header Text)	
-h text	Uses text underneath the words Quick View Plus in the view window's toolbar. See sample view window above.

	Example -h for File Manager
-dt (Disable Tool bar)	
-dt	The tool bar is not displayed.
-dl (Disable Launch)	
-dl	The launch button on the tool bar is grayed.
-dp (Disable Print)	
-dp	The print button on the tool bar is grayed.
-dc (Disable Copy)	
-dc	The copy to clipboard button on the tool bar is grayed.
-ds (Disable Search)	
-ds	The search buttons on the tool bar are grayed.
-x (Use existing window)	
-x	If SCCVAPI.EXE has been called previously and the user has not closed the window it created, a second window will not be created, rather the new file will replace the file in the existing window. If there is no previous window, the -x is ignored. Please Note: If a previous window exists, all other options on the command line, except -d (Display Name), will be ignored.

Each time SCCVAPI.EXE is run with a valid filename, a new view window is created, unless the -x options is included on the command line. The task that executed SCCVAPI.EXE has no further control of the window.

For example, the sample screen shot above might have been generated with the command line:

```
SCCVAPI C:\WPG\MAILBAG.WPG -p 0,0,400,300 -h for File Manager
```

7. *ViewAPI* method 2 - Function calls to SCCVAPI.DLL

SCCVAPI.DLL has a single entry point with an ordinal value of 100 (decimal). A sample application showing the basic functionallity of this DLL is in the VIEWAPI sub-directory of your Quick View Plus directory. All values and structures defined below are in SCCVAPI.H in the same sub-directory.

The entry point is defined as follows:

DWORD **SccViewer**(WORD **wFunction**, HANDLE **hViewer**, VOID FAR * **pParam**);

Where **wFunction** is one of the following

SCCVAPI_GETLEVEL

May be called at any time
hViewer is not used
pParam is not used

Return value will be the release level of the ViewAPI. Will be 2 for this release.

SCCVAPI_INIT

Should be called when your application is run.
hViewer should be NULL
pParam should be a pointer to an SCCVAPIINIT structure

```
typedef struct SCCVAPIINITtag
        {
        WORD            wSize;
        BYTE            szHeader[40];
        } SCCVAPIINIT;
```

wSize	Should be set to sizeof(SCCVAPIINIT) before the call
szHeader	Text to be displayed below **Quick View Plus** in the view window For example "for File Manager"

Return value will be a handle to be passed as hViewer in subsequent calls to SccViewer() or NULL if the call failed.

If the call fails, remember to unload the DLL.

SCCVAPI_CREATE

Creates a view window.
hViewer should be the value returned by SCCVAPI_INIT
pParam should be a pointer to an SCCVAPICREATE structure

```
typedef struct SCCVAPICREATEtag
        {
        WORD            wSize;
        BOOL            bShowToolBar;
        BOOL            bAllowLaunch;
        BOOL            bAllowPrint;
        BOOL            bAllowCopy;
        BOOL            bAllowSearch;
        } SCCVAPICREATE;
```

wSize	Should be set to sizeof(SCCVAPICREATE) before the call
bShowToolBar	If FALSE, the tool bar is not displayed.
bAllowLaunch	If FALSE, the launch button on the tool bar is grayed.
bAllowPrint	If FALSE, the print button on the tool bar is grayed.
bAllowCopy	If FALSE, the copy to clipboard button on the tool bar is grayed.

bAllowSearch	If FALSE, the search buttons on the tool bar are grayed.

Return value is a handle to the view window, this handle can be used to size, position, show, maximize, etc... the window OR NULL if the call failed.

View window will be hidden until the application calls ShowWindow().
This function may be called again and again to create as many view windows as your application requires.

SCCVAPI_VIEW

Should be called to view a file
hViewer should be the value returned by SCCVAPI_INIT
pParam should be a pointer to an SCCVAPIVIEW structure

```
typedef struct SCCVAPIVIEWtag
        {
        WORD        wSize;
        HWND        hViewWnd;
        BYTE        szPathName[255];
        WORD        wViewAs;
        BYTE        szDisplayName[40];
        BOOL        bUseDisplayName;
        BOOL            bDeleteOnClose;
        } SCCVAPIVIEW;
```

wSize	Should be set to sizeof(SCCVAPIVIEW) before the call	
hViewWnd	Window to display the file in. Must be a handle returned by SCCVW_CREATE	
szPathNam	The full path name to the file to be viewed	
wViewAs	One of the following	
	NULL	View the file normally
	SCCVAPI_VIEWASASCII	Force an ASCII view
	SCCVAPI_VIEWASHEX	Force a Hex view
	SCCVAPI_VIEWASANSI	Force an ANSI view
	SCCVAPI_VIEWASUNICODE	Force an UNICODE view
	SCCVAPI_VIEWASASCII8	Force an ASCII8 view
	SCCVAPI_VIEWANSI8	Force an ANSI8 view
	SCCVAPI_DONTVIEW	Do not view
	SCCVAPI_VIEWASMAC	Force a MAC view
	SCCVAPI_VIEWASMAC8	Force a MAC8 view

szDisplayName	File name to use for display purposes
bUseDisplayName	If FALSE, szDisplayName will be ignored and the display name will be the file name part of szPathName. FALSE is the standard case
bDeleteOnClose	If TRUE then the viewer will delete the file when the view of it is closed. This is useful for temporary files created specifically for the view, such as attachments in mail applications

Return value will be 1 if the call was successful or 0 if it failed.

SCCVAPI_EXIST

The view windows created by this API are independent, overlapped windows. This means that the user can use a view window's system menu to close it at any time. Subsequent SCCVAPI_VIEW or SCCVAPI_DESTROY calls will fail gracefully, but it is more likely you will want to check if a view window still exists and possibly create a new window, before making a SCCVAPI_VIEW call. An IsWindow() call is not a sufficient substitute for this call.

hViewer should be the value returned by SCCVAPI_INIT
pParam should be a pointer to the window handle to be checked

Return value will be 1 if the window still exists or 0 if it has been closed by the user.

SCCVAPI_PRINT

Prints the file currently being viewed. The view window does not have to be visible.

hViewer should be the value returned by SCCVAPI_INIT
pParam should be a pointer to an SCCVAPIPRINT structure

```
typedef struct SCCVAPIPRINTtag
        {
        WORD            wSize;
        HWND            hViewWnd;
        BOOL            bDoDialog;
        } SCCVAPIPRINT;
```

wSize	Should be set to sizeof(SCCVAPIPRINT) before the call
hViewWnd	View window containing the file to print. Must be a handle returned by SCCVW_CREATE.
bDoDialog	If TRUE, the print dialog will be envoked, as if the user had clicked on the print button in the tool bar. If FALSE, the file will be printed to the default printer with no setup dialog. The printing progress dialog, will always appear.

Return value will be 1 if the file was printed or 0 if the user canceled the operation.

SCCVAPI_DESTROY

Should be called to destroy a view window

hViewer should be the value returned by SCCVAPI_INIT
pParam should be a pointer to the window handle to be destroyed.

SCCVAPI_DEINIT

Should be called when your application is closed.
hViewer should be the value returned by SCCVAPI_INIT
pParam should be NULL

All view windows will be automatically destroyed.
Note: After this call, remember to do a FreeLibrary on the DLL

The following sample code shows how to load the DLL, get its entry point and initialize it. Since the DLL will not exist if *Quick View Plus* is not installed, a NULL value in pVIEWAPI indicates the viewers are not available. Full sample code is in the VIEWAPI sub-directory of your Quick View Plus directory.

```
#include "SCCVAPI.H"

/* global variables */
HANDLE          hVIEWAPI;
SCCVAPIPROC pVIEWAPI;
HANDLE          hViewer;
```

```
VOID LoadVIEWAPI()
{
SCCVAPIINIT locInit;

    hVIEWAPI = LoadLibrary("SCCVAPI.DLL");

    if (hVIEWAPI > 32)
        {
        pVIEWAPI = GetProcAddress(hSCCVAPI,MAKEINTRESOURCE(100));

        locInit.wSize = sizeof(SCCVAPIINIT);
        strcpy(locInit.szHeader,"for My App");

        hViewer = pVIEWAPI(SCCVAPI_INIT,NULL,&locInit);

        if (hViewer == NULL)
            {
            FreeLibrary(hVIEWAPI);
            hVIEWAPI = NULL;
            pVIEWAPI = NULL;
            }

        }
    else
        {
        hViewer = NULL;
        hVIEWAPI = NULL;
        pVIEWAPI = NULL;
        }
    }
}
```

remember to FreeLibrary(hSCCVAPI) before your application exits, DLLs loaded with LoadLibrary() are not automatically freed when your application exits.

8. *ViewAPI* method 3 - Visual Basic Control SCCVIEW.VBX

1. Add the SCCVIEW.VBX to your Toolbox in Visual Basic. [File Add File command]
2. Add the SCCView Control to your form.

The properties supported in this control are:

Standard Properties

CtlName	default is "SCCView1"
BackColor	default set at design time to Windows default.
Left	position of the left edge of the control
Top	position of the top edge of the control
Width	width of the control
Height	height of the control

Visible	TRUE or FALSE
Parent	handle of the parent of the control
DragMode	specifies manual or automatic dragging of the control
DragIcon	the icon to display during the drag operation
Tag	user specified value for identification

For a description of each of these properties please refer to your Visual Basic documentation.

Additional Properties

When you want to view a file, set the **FileToView** property to the fully qualified path name of the file to be viewed. For example, if the view has been added to your form as SCCView1, then to view the user's C:CONFIG.SYS file, add the following code to your Visual Basic application.

```
SCCView1.FileToView = "C:\CONFIG.SYS"
```

Questions and problems concerning the ViewAPI should be directed to Inso product support at 1-312-527-HELP. Please mention the ViewAPI when you talk to product support.