# SwiftView C/IE ActiveX Control Reference Manual

The SwiftView C/IE ActiveX Control (hereafter referred to as "SAX") is a packaging of Windows 95/NT SwiftView as a Microsoft ActiveX control for the following configurations:

```
   Internet Explorer 5.x or 6.x, or
   Visual Basic 5.0, 6.0, on
   Windows 98 2nd edition, ME, NT 4.0 SP4-SP6b, 2000 (all
   versions), XP, or XP SP1 (all versions released before May 2002)
```

In general, ActiveX controls can be embedded in a variety of "container applications" such as Microsoft Office apps, and various Windows development environments including Microsoft C++ and Visual Basic. However, at this time SAX does not implement the extensive ActiveX and OCX API's required by some of these environments, in particular, it is not a "Document Object Server".

## Related Packages

The SwiftView Netscape Plug-In is a functional equivalent of SAX for Netscape.  It can also be used in IE, but normally is not recommended for IE.  This manual documents only the areas in which SAX differs from the Plug-In.  See the SwiftView Plug-In Reference Manual for the common functionality.

SwiftInside is an alternative packaging of SwiftView that supports programming of embedded SwiftView instances in C++ and VB applications. In particular, this package allows the application to take full control of the user interface, including mouse and keys in the drawing area.

## Installation

SAX consists of the following files, installed under the SwiftView program directory (usually c:\Program Files\SwiftView):

```
   svocx.ocx (~200 KB)        -- The control
   sview.exe (~800 KB)        -- Standalone SwiftView
   sview.tra                  -- Empty (or non-English) text translation file
   sview.str                  -- Sample (English) translation file
   sview_<lang>.tra           -- Text translation files for various
languages, e.g:
                                sview_French.tra
   svreadme.txt               -- Plugin/SAX release notes
   svaxref.txt                -- This manual
   mime_sample.typ            --  Example mime types file with default
settings.
   svinst.exe                 --  Install/deinstall program
   sview.inf                  --  Install/deinstall control file
Plus, for the ICS-only version:
   mime.typ                   -- MIME types/suffixes to register
```

The cabinet files are certified using Microsoft Authenticode, and when downloaded by Internet Explorer automatically installs the files in the given directories.  It is a "silent" install, presenting no SwiftView dialogs to the user - successful installation is detected because the object is immediately executed.

At SAX installation, Internet Explorer is registered for viewing SwiftView document files. This allows double-clicking or pressing the right mouse button on a file in Windows Explorer to display or print the file. Note that after the evaluation period ends, local file viewing requires licensing other than a basic web site license. Because of this, if another application is registered for the "Open" action, SwiftView is registered as extra "SwiftView" and "SwiftPrint" actions, and the default actions are unchanged. However, currently SAX is registered with IE for all suffixes, overriding any prior installed application, so opening local files in IE requires non-web SwiftView licensing.

The following table lists the default MIME types and suffixes that are registered by the standard installation to run SAX:

```
       application/cals-1840:ras,cal:Cals Raster File;
       image/pcx:dcx,pcx:PC-Paintbrush File;
       image/tiff:tiff,tif,001:Tagged Image File Format;
       application/vnd.hp-HPGL:hpgl,plt,hp,hpg,hgl:HP Graphics Language;
       application/vnd.hp-PCL:prn,pcl:HP Printer Control Language;
       application/vnd.SwiftView-ZIP:zhp:Zipped SwiftView Files;
       application/vnd.SwiftView-ICS:ics,vdt:SwiftView Command File
```

Note - unlike the plug-in, we register the standard MIME types for cals and pcx, to remain consistent with standards and facilitate merging actions as described above.

The "ICS-only" versions (svinstall...ics.cab and svinstall...ics.exe) only register the suffix zhp (not ics, because Microsoft took over .ics for calendar files) and the MIME types application/vnd.SwiftView-ICS and application/vnd.SwiftView-ZIP (see below for a discussion of MIME types). Note that they do not remove registrations for other MIME types; if you have previously installed a full-MIME-type version, you must unregister it before installing the ICS-only version (see Deinstallation below). Similarly, the mime.typ file installed by these packages must be removed to reinstall a full-MIME-type version.

## Non-Standard Installations

SAX registers Internet Explorer (and therefore SAX) to view files supported by SwiftView, e.g. from Windows Explorer.

- The program registration for files can be overridden to view the file in a new standalone SwiftView window, or removed entirely, by running the standalone SwiftView installed with SAX:

```
      c:\Program Files (x86)\SwiftView\sview -reg  (run standalone SwiftView)
      c:\Program Files (x86)\SwiftView\sview -delreg (remove all
registrations)
```

  Note that each time you click on a file in Explorer, a new SwiftView window is started.

- Newer Windows versions include WangImaging as the default .tif file viewer. When SwiftView installs, it does not take over existing default applications, but adds SwiftView as an alternate application, available with the right mouse button on the file.  If you wish to view .tif files with SwiftView by double-clicking on a .tif file, open Windows Explorer and select View, Options, File Types, TIF Image Document, Edit, Actions: SwiftView, Set Default.

- If you need to restrict local file access to fewer MIME types, add support for other types that SwiftView supports such as JPEG, or otherwise reconfigure local file access types or suffixes, you can do the following:

  1. Run

     ```
     sview -delreg
     ```

     to remove the default SwiftView registration.

  2. Edit the following text:

     ```
     application/cals-1840:ras,cal:Cals Raster File;
     image/pcx:dcx,pcx:PC-Paintbrush File;
     image/tiff:tiff,tif,001:Tagged Image File Format;
     application/vnd.hp-HPGL:hpgl,plt,hp,hpg,hgl:HP Graphics Language;
     application/vnd.hp-PCL:prn,pcl:HP Printer Control Language;
     application/vnd.SwiftView-ZIP:zhp:Zipped SwiftView Files;
     application/vnd.SwiftView-ICS:vdt,ics:SwiftView Command File
     ```

     into the file

     ```
     c:\Program Files (x86)\SwiftView\mime.typ
     ```

     All the lines in the file that do not begin with '#' (comment) are concatenated together to make one configuration string.  The format is:

     ```
     MIME:suffix,suffix...:description of mime type for file open
     dialog;MIME...
     ```

  3. Either reinstall SAX (not the ICS-only version) or run

     ```
     sview -reg
     ```

     depending on whether you want to use SAX under Internet Explorer or standalone SwiftView to view local files.

- If you would prefer to open a new window when viewing a file from Explorer, disable "Use DDE" in the Open action for the SwiftView file type(s) in Windows Explorer's View... Options... FileTypes... Open... Edit tabs.  (This is the default if you are using standalone SwiftView).

## Deinstallation

All SwiftView packages, including SAX, can be removed from your system by either:

- Clicking on "Start", "Settings", "Control Panel", "Add/Remove Programs", "SwiftView Products", "Add/Remove...", or

- Downloading and running ActiveX is always run by IE in preference to a plug-in, so SAX must be
deinstalled in order to reenable the SwiftView plug-in.

## Incorporating SAX into HTML pages

A SwiftView document can be viewed with SAX in Internet Explorer in one of five ways:

1. clicking on a hyperlink to a SwiftView document
2. entering a URL to a SwiftView document in the IE address bar
3. opening a local SwiftView document from Windows Explorer
4. an <EMBED> tag on the current HTML page
5. an <OBJECT> tag on the current HTML page

The first three cases by default result in a "full page" instance that fills the IE window, with a SwiftView control bar on the right-hand side. <EMBED> and <OBJECT> tags display the SwiftView document in a defined area of the current page, by default with no SwiftView control bar (see the WIDTH attribute and CONTROLS PARAMeter below for details).  A page can display more than one of these instances at the same time.

In all cases except <OBJECT>, for a SwiftView document to be viewed, it's MIME type from the Web server must match one of the registered SwiftView MIME types, or it's suffix must match one of the registered SwiftView suffixes (see installation above).

An HTML page may contain any number of instances of SAX, using the <OBJECT> or <EMBED> tag.  The <OBJECT> tag is the only method of automatically triggering a download and install of SAX on the client machine.  Here is an example of the <OBJECT> tag:

```
<p><OBJECT WIDTH=200 HEIGHT=400
    CLASSID="CLSID:7DD62E58-5FA8-11D2-AFB7-00104B64F126"
    ID=myobj

CODEBASE="http://www.swiftview.com/product/current/svinstall_a_stat.cab#Versi
on=5,3,0,4"
   >
   <PARAM NAME="SRC" VALUE="http://www.swiftview.com/tech/svrel.zhp">
   <PARAM NAME="SCROLLBARS" VALUE="vertical">
   <PARAM NAME="COMMANDS" VALUE="gui controls display rightbuttons|dpage
last">
   </OBJECT>
</p>
```

This example positions a 200x400 SAX window at the current position in the page, and draws the file http://www.swiftview.com/tech/svrel.zhp in it.
By default, like the SwiftView plug-in, the SAX window handles the usual SwiftView key and mouse interface in the draw area, but does not include the standard SwiftView control bar.  The "gui controls display rightbuttons" command enables the control bar just like with SwiftView Web plug-in in full-page mode.

To build web pages that view consistently in Internet Explorer and Netscape Navigator, you must use <EMBED> tags, not <OBJECT> tags in your

.html files.  Automatic download/install of SAX should be accomplished by including a single (optionally zero-sized) OBJECT in the web page where you want the installation to occur.  You can also support Netscape AutoInstall so that the plug-in can be automatically downloaded for Netscape Navigator just as SAX is downloaded for Internet Explorer.

For "full-page" or EMBED (non-OBJECT) references to work, your web server must have MIME types registered as described in the Plugin/SAX release notes.

Example EMBED tag:

```
        <p><embed src="http://www.swiftview.com/tech/2pages.hpg"
        type=application/vnd.SwiftView-ics
        width=400 height=300
        controls=topbuttons
        scrollbars=vertical>
```

See the SwiftView Plug-In Reference Manual for more details on EMBED tags.

## Attributes and Parameters

OBJECT tags have separate attributes and parameters, as you can see from the example above.  OBJECT parameters are equivalent to EMBED attributes, and are accessable from ActiveX interfaces.

The key OBJECT tag attributes are:

```
    ID:            A string identifying the OBJECT instance, to access the
                   instance from VBscript or Jscript.
    WIDTH, HEIGHT: The area of the containing document used by the OBJECT
                   instance.  Values are in pixels or a percentage of the
                   browser document window.  "100%" sets the size to match
                   the window.  If INLINE=false, must be set to 27x27, the
                   size of the "Show" button.  If width or height is zero,
                   INLINE=false is forced (not yet implemented).
    LEFTMARGIN
    RIGHTTMARGIN
    TOPMARGIN
    BOTTOMMARGIN   XXX do these exist? Set the margin between the
                   browser document window and the page, in pixels.
    CLASSID:       The universal ID that is used to locate the
                   installed SAX via the Windows registry.  This attribute
                  should always appear with the value shown above.
    TYPE:          A SwiftView document MIME type.  This is an alternative
                   to the CLASSID attribute to cause SAX to be used.
    CODEBASE:      The location of a local file or Web URL that can be
                   downloaded and executed to install and register SAX.
                   Optionally, the desired version of SAX (actually, the
                   File version in the .dll Properties dialog) can be
                   specified with #Version=<version> appended to the URL.
                   If the version of a currently registered SAX is less
                   than that specified, it will be updated.  To force
                   installation, specify #Version=9,9,9,9.

                  The CODEBASE tag is required, due to an IE bug: without
```

```
                   it, an incorrect file may be sent to the object.
                   However, no more than one OBJECT may have a CODEBASE
                   with a version number exceeding the downloaded
                   control's version, or the download will be started
                   multiple times.

                  Note - if SAX has already been run in this session of
                   IE, IE will not be able to immediately reinstall SAX,
                   and will request a reboot.

The following OBJECT PARAMeters are supported by SAX:

        SRC="URL[#ics cmds]"    - where to get the SwiftView document/ics
                                  file.  Also used to get Web config and
                                  license files.  Follows standard URL
syntax,
                                  including file: or http:.  If the URL
                                  contains the '#' character, everything
                                  after it is processed as ICS commands,
                                  prior to other OBJECT parameters.
                                 (The "#" feature does not work under IE.)
        INLINE=false|true       - if true, show the SwiftView document
                                 in-line in the HTML document, else in a
                                 separate "partner" SwiftView window,
                                 with a "Show" button in the HTML page.
                                  Default "true". (see plug-in partner
                                 mode.)
                                  Equivalent ICS command: gui inline
        AUTOSTART=false|true    - if true, show the SwiftView document
                                 as soon as the HTML page is viewed.
                                  Default "false".  Ignored if INLINE=true.
                                  Equivalent ICS command: gui autostart
        CONTROLS=rightbuttons|topbuttons|none
                               - Show a set of buttons at the top or
                                 right of the document, within the
                                 plug-in instance area.
                                  Default "rightbuttons" if WIDTH=100%,
                                 else "none".  Ignored if INLINE=false;
                                 use COMMANDS PARAMeter.
                                  Equivalent ICS command: gui controls
display
        FULLSCREEN=false|true   - causes the object to fill the browser
                                  window.  As a result, the controls
                                  default to "rightbuttons", per the above
                                  WIDTH rule.
        SCROLLBARS=vertical|horizontal|both|none
                               - show the desired scrollbars.
                                  Default "both".  Ignored if INLINE=false;
                                 use COMMANDS PARAMeter.
                                  Equivalent ICS command: gui scrollbars
                                 display
        BORDERWIDTH=n           - width of border around entire object
                                  Equivalent ICS command: gui borderwidth
        COMMANDS="ics commands" - a string of ICS commands to process.
                                 The string may contain multiple commands
                                 separated by newline or pipe ("\\n" or "|").
                                  Use "&#34" to embed a double quote.
```

```
FIRSTICS="ics commands" - Same as COMMANDS, but processed before
                         SRC.
LICENSE="license string"- a customer-specific license token (an
                          alternative to LAN/Web licensing). This
                          string must match the customer data
                          embedded in this copy of the control dll
                          to be a valid license.
```

All attribute/parameter names are case-insensitive. Each PARAMeter has an equivalent EMBED attribute.

PARAMeters map directly to ActiveX properties; each of the PARAMeters listed above represents a string-valued ActiveX property accessable from various container applications via e.g. IPersistPropertyBag. SRC, FIRSTICS, COMMANDS, and LICENSE are also accessable as individually settable custom properties that may set at any time. LICENSE is processed each time it is set. SRC, FIRSTICS, and COMMANDS changes issued prior to the setting in effect at the initial display of the control are ignored. The settings at initialization are processed in the order described in the "Plugin Reference Manual". The COMMANDS property can be used to provide low-level dynamic manipulation of the control both within and outside of a browser. The most recent setting is reapplied on a ICS reload command, so for many applications the sendCmd method is a better way to dynamically manipulate the control. See "Support for Microsoft Office, Visual Basic, and Visual C++" below for more information on ActiveX programming interfaces.

As noted, most SwiftView-specific PARAMeters have equivalent ICS commands; see the SwiftView Technical Reference Manual for details. Also note that CONTROLS, SCROLLBARS, and BORDERWIDTH do not affect the partner mode user interface (INLINE=false); use COMMANDS or FIRSTICS to configure controls and scrollbars in this case.

If SRC is not set, no document is defined, and web-site license and config files are not read. A SRC property change after the initial display of the control sends a reload command, which completely reinitializes and licenses the control from the new document, current property settings, and config files.

Note - IE's handling of #fragment is buggy - #fragment on a plain URL fails, and #fragment in an EMBED SRC attribute causes the control to fail to run, or the plug-in to be run instead.

## Invoking SAX from Web Browsers

See the SwiftView Plug-In Reference Manual; SAX works the same as the Plug-In. Sview.exe is started from the $PATH, and is installed in the <windows directory>.

## Browser Document Transfer to SAX

See the SwiftView Plug-In Reference Manual; SAX works the same as the Plug-In.

## Control of SAX from VBscript or JavaScript (JScript)

SAX can be controlled from VBscript or JavaScript code running in Internet Explorer. When combined with the LiveConnect version of the Netscape plug-in, JavaScript can be written that controls SwiftView in both Netscape and IE.

Your script can change any property, call the sendCmd method, and get callbacks (see Support for Visual Basic and Visual C++ below).

HTML code fragment examples:

```
<!-- --------- JavaScript: --------- -->


<script LANGUAGE="JavaScript">
<!--  these comments hide code from non-SCRIPT browsers
function sendCmd(cmd) {
   svctl.sendCmd(cmd)
   // "svctl.Commands = cmd" also works, but affects reload
   // return false so that the form doesn't submit and reload the page.
   return false;
}
//-->
</script>

<FORM NAME="cmdField"
  onSubmit="return sendCmd(cmdField.stringfield.value);">
  <INPUT TYPE=SUBMIT VALUE="Send a command to control:">
  <INPUT TYPE="string" NAME="stringfield"
         VALUE="ldoc http://www.swiftview.com/tech/svrel.zhp|draw">
</FORM>

<SCRIPT LANGUAGE="JavaScript" FOR="svctl" EVENT="Callback(Callback)">
<!-- If you put up a dialog here, IE's stack will blow!  -->
<!--
  document.showcb.text.value = Callback;
//-->
</SCRIPT>


<!-- --------- VBScript: --------- -->


<SCRIPT language="VBScript">
<!--
Sub sendCmd(cmd)
   call svctl.sendCmd(cmd)
   rem "svctl.Commands = cmd" also works, but affects reload
End Sub
-->
</SCRIPT>

<!-- return false so that the form doesn't submit and reload the page. -->
<FORM NAME="cmdField"
  ONSUBMIT="sendCmd(cmdField.stringfield.value); return false">
  <INPUT TYPE=SUBMIT VALUE="Send a command to control:">
  <INPUT TYPE="string" NAME="stringfield"
         VALUE="ldoc http://www.swiftview.com/tech/svrel.zhp|draw">
</FORM>

<SCRIPT language="VBScript">
<!--
Sub svctl_Callback(ByVal CBData)
   showcb.text.value = CBData
End Sub
-->
</SCRIPT>


<!-- --------- Both VBScript and JavaScript: --------- -->


<!-- I have no idea why this can't appear after the OBJECT: -->
<FORM NAME="showcb">
<p>Callback fetched on Callback event:
   <!-- I have no idea why WIDTH doesn't work -->
   <INPUT TYPE=TEXT NAME="text" WIDTH=600
    VALUE="no callback yet">
</p>
</FORM>

<!-- JavaScript and VBScript use different name parameters, just give both. -
->
<OBJECT WIDTH="400" HEIGHT="400"
 ID=svctl NAME=svctl
 CLASSID="CLSID:7DD62E58-5FA8-11D2-AFB7-00104B64F126"
 CODEBASE="svinstall_a_stat.cab#Version=5,3,0,4"
>
  <PARAM NAME="src" VALUE="http://www.swiftview.com/tech/svm.zhp">
</object>
--------------------
```

Note that some other documented mechanisms for calling functions on events have not been found to work.

## Support for Visual Basic and Visual C++

SAX provides a set of ActiveX interfaces that give access to ICS command and callback mechanisms for both VB and Visual C++ applications. The custom properties SRC, COMMANDS, and LICENSE are settable custom string properties, as discussed above. In addition, an ActiveX method and event are supported to send ICS commands and get ICS callback strings. Here are their Visual Basic declarations:

```
Method:
        sendCmd(ByVal commands as String)
                               Send ICS commands.  SwiftView will be
                               started if necessary.

Event:
        Callback(callback as String)
                                Fired once for each callback string.
```

SwiftView is fully active during Visual Basic design time, just like at runtime.  Access the control in the usual VB manner: click Project - Components, click the control checkbox, and OK.  In development environments, SAX appears as the "SwiftView ActiveX control module", class Sview, or "Sview Control".

A sample VB6 project demonstrating these interfaces is available.

Note: as the sample project describes, you should avoid sending commands to SAX in the Form_Load() subroutine - it will cause VB to crash.

## Support for Microsoft Office

SAX does not yet support Microsoft Office - Office requires property setting pages, which are not implemented.

## Support for Microsoft Document Object Containers

A SwiftView OLE Document Server is available on a limited-support basis that packages SAX to conform to the requirements of some Document Object containers.  Contact SwiftView for more information.

## ICS Commands

Any standard ICS command sequence may be sent to the sendCmd method or set in the COMMANDS property (except that "reload" or "gui controls update" should not be included in the COMMANDS property).  Multiple commands separated by the pipe character may be used.  Any errors are reported in a Callback event.  See the SwiftView Technical Reference Manual for the specification of ICS commands and callback strings.

Note that if the application sends ICS gui configuration commands with the sendCmd method or COMMANDS property after the initial display of the control, or if no SRC parameter is set, a "gui controls update" command must be sent for the commands to take effect and the user interface to be (re-)constructed.  Without a SRC parameter or a "gui controls update" command, NO SwiftView USER INTERFACE WILL APPEAR.  "gui controls update" may be sent at any time to reconfigure the user interface.

For example, to create the full standard SwiftView user interface and load and display a document, send the following commands:

```
   "ldoc mydocfile|gui controls display rightbuttons|gui controls
update|draw"
```

Currently, SAX does not support OCX compound document containers.  Contact SwiftView if you are interested in such a control.

Direct control of user input in the SAX drawing area by the application is not currently supported.  See CmdSwiftView() in the SwiftInside Pluggable Object External Specification for details on using the SwiftInside library directly to accomplish this.

## Configuring SAX

SAX can be configured using the same mechanisms as the SwiftView plug-in, except that the local system SwiftView plug-in configuration file is not searched for in the browser's home directory.  See "Configuring the Plug-In" in the SwiftView Plug-In Reference Manual for details.

## Licensing SAX

SAX supports the the same licensing methods as the SwiftView plug-in, plus the LICENSE parameter, which licenses a copy of the SAX programs to a particular customer.  Plug-in-LAN and Web licenses also license SAX.  The LAN license must be in the SwiftView program directory, else a directory in $PATH.  LAN licenses may not be in the c:\ filesystem; they are intended to be installed with the software on a LAN server.

Each time the SRC parameter is set or an ldoc command issued, licensing is checked.  It is not checked until then, so no license messages are issued until a document is viewed and it's licensing fully established.

## Native Language Support

See the SwiftView Plug-In Reference Manual; SAX works the same as the plug-in, and shares the same translation file, sview.tra, except that the translation file is not searched for in the browser's home directory.

## Unsupported Features

The following common features of an ActiveX control are not currently supported by SAX (Who knows, they might actually work, but they have not been tested!).  Contact SwiftView if you need support for any of these features; some may be supported in future releases.

- aggregation - embeding SAX in another ActiveX control.

- Ambient properties, e.g. font, BackColor, ForeColor, etc.

- Stock properties: any not documented above.  (Many Stock properties, e.g. Font, default to the corresponding Ambient property.)

- older IPersist interfaces other than IPersistPropertyBag that may be required by certain older containers.

- embedded printing - SAX displays the document name when the container prints the page.  Actual document printing is only available through the SwiftView user interface.

- fancy OLE compound document mechanisms (e.g. in-place editing/zoom) are not supported.

- windowless controls - SAX cannot draw over stuff drawn by the container - it draws in an opaque rectangle.

SAX is not recommended for use with the AOL browser.  Use the plug-in instead, by downloading and running svinstall_p_stat_libs.exe, or installing the plug-in with Netscape. Alternatively, you can run Internet Explorer over the AOL connection.

## Debugging

If you encounter problems integrating SAX, and suspect some internal problem in the .ocx, a debug version is available. It writes the log file $NDGDBUG_FILE\sviewocx.log, else if $NDGDBUG_FILE is not defined, sviewocx.log in c:\, else $TMP. Each line is prefixed by the current time since the epoch in seconds . Note that sview.log from sview.exe is also written in the same directory. These files are overwritten by each subsequent Active X instance.

## Security

All "plot", "system", and "save" commands are disallowed except for "plot 99" on Windows, which plots to the current windows printer. This restriction is necessary because these commands allow writing an arbitrary file. This includes commands in ICS files obtained from the web server and commands from LiveConnect or Active-X interfaces.

ICS files obtained via SwiftServe or commands configured on keys or buttons are currently deemed safe and are not restricted. However, they may be restricted in the future.

In summary, the only file that can be written by SwiftView without the user explicitly naming the file or pressing a button or key on the SwiftView user interface is sview.ini in the Windows directory.