WebSign SDK (ActiveX) Developer’s Guide

Table of Contents

1.0 – Introduction

Thank you for choosing WebSign SDK (ActiveX), the most comprehensive signature capture software SDK in the market. WebSign SDK (ActiveX) gives you the features and functionality to Sign and Secure electronic documents/e-mails of any COM compatible applications, using powerful authentication tools. WebSign SDK (ActiveX) comes with built- in support for handwritten electronic signatures. Electronic signature capture is done using ePad device.

2.0– WebSign SDK (ActiveX) Key Features

1. Enables applications with Electronic Handwritten Signature capability to capture the signature and secure the content using advanced encryption technologies.

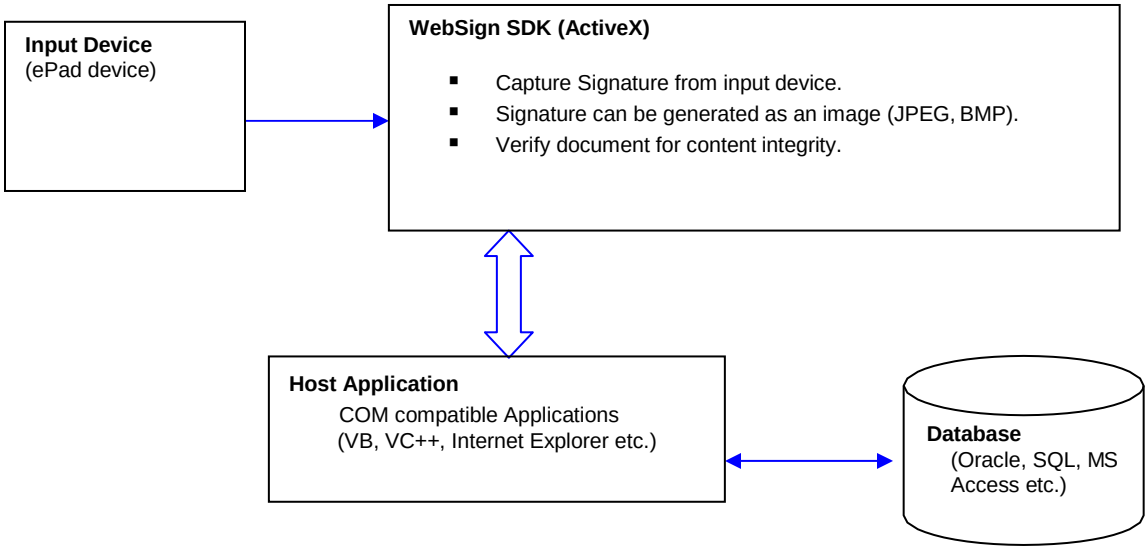
2. Allows signature capture through ePad device.

3. Simultaneously captures the signer details, time-stamp of signing, location that provide validity to authenticated electronic content.

4. Signed content integrity can be verified at any point of electronic transmission or storage.

3.0 – WebSign SDK (ActiveX) Integration Overview

WebSign APIs are bundled within the DLL along with standard User Interface elements. When the host application invokes these API functions, to initiate the act of Signing [StartSign ()], the signature from the input device is captured along with other signer details. This signature data along with content hash [HashData] is encrypted and processed into BASE64 string format.



Input Device: ePad can be used for signature capture.

Host Application: Any COM compliant application like Visual Basic, VC++, Internet Explorer etc.

WebSign SDK (ActiveX): WebSign SDK (ActiveX) contains capture component (COM Object with User Interface) and supporting libraries. API functions are available to perform different tasks such as to generate signature image [SaveToFile ()] and check content Integrity [VerifyContents ()].

Database: The captured signature data (Encrypted Base64 string) can be stored in the host database (Oracle, SQL, DB2, Sybase, MS Access) or in a file system. The stored data is interpreted by IntegriSign API function [OpenSign ()] to reproduce the signature at a later time.

4.0 – Pre-requisites

Optimum system requirements to install and run WebSign SDK (ActiveX):

1. Windows 7 and up operating system
2. Pentium-class PC
3. ePad device
4. 32 megabytes of RAM
5. 20 megabytes of free disk space

5.0 – Components of WebSign SDK (ActiveX)

The following are the components required by WebSign SDK (ActiveX) to integrate into any application:

1. websignax.cab – The CAB file that installs all the necessary signature capture components onto the client machine where signatures are captured (Which will be referred by the <object> tag in htm file).
2. esCapSvr.dll – Component responsible for decoding the signature data and then saving it as an image file (bmp, jpeg). This is usually installed on the server.
3. esUtil.dll – WebSign SDK (ActiveX) supporting library.

6.0 – Running WebSign enabled sample

To run the WebSign sample follow these steps:

1. Publish the simplecapture.htm, APIdemo.htm and processintegrisign.asp onto the web server (IIS).
2. Edit the simplecapture.htm (form action and URLs) and ASP pages to suit your web server requirements. Follow the comments inside the htm and asp page.
3. Copy the websignax.cab from SDK folder to the same directory where, htm and asp pages are copied.
4. Copy and register the esCapsvr.dll from the SDK folder onto the machine where Web server is installed.
   To register, click Start → Run and type regsvr32 <dll_path>/ esCapsvr.dll Where <dll_path> is the location where the esCapsvr.dll dll is copied.
5. Copy eSUtil.dll from the SDK folder into the windows system folder of the Web Server machine.
6. Start IE on client computer and Access the simplecapture.htm page.

Notes:

1. esCapsvr.dll is required if images need to be stored on the web server.
2. If the SDK is used in Web development, esCapsvr.dll must be copied and registered onto the system where web server is installed and then access it from the ASP pages.

7.0 – Steps to Use WebSign SDK (ActiveX)

Follow these steps to make your application WebSign enabled:

1. List the documents (forms) that are to be signed.

2. Add the Signature component (IntegriSign Core Component for WebSign SDK (ActiveX)) to the forms by referring to esW25COM.ocx. <object> tag should look like the one specified in simplecapture.htm page.

3. Set the **HashData** property on the instance of esW25COM component wherever the content integrity is necessary. Typically, it should be set before the **StartSign**, **VerifyContents** and **OpenSign** methods.

4. Implement other functionality for Capturing, Storing & Opening signatures taking the following SDK reference.

8.0 – WebSign SDK API

Following are the properties, methods and functions that can be used to enable an application with IntegriSign Capture functionality. These functions are available through the esW25COM and esCapSvr components.

esW25COM Component

Properties

HashData

Description
This property has to be set with the data that has to be authenticated. These values are taken from the present container viz. a form or document. To implement the content integrity functionality this property has to be set before the sign capturing process. This should also be set wherever you need to check the authenticity of the content. Typically, this should be called before using the following methods:

1. StartSign
2. OpenSign
3. VerifyContents

Note: When this property is set to null, hashing is not implemented.

BackColor

Description
Sets the background color of the signature component.

ForeColor

Description
Sets the color of the signature.

Cross

Description
Sets whether the cross-mark needs to be displayed on the signature when the content authentication failed. Two possible values are CROSS_OFF and CROSS_ON.

SignDlgCaption

Description
Sets caption for the signature capture dialog.

ShowSignerName

Description
Sets whether Signer Name field should be displayed in the signature capture dialog. Two possible values are TRUE or FALSE.

ImageThickness

Description
Sets the pen thickness to be used while generating the image of the signature. The values can range from 1 to 10. Default value for this property is 2.

Note: This property is obsolete and is there only for compatibility purposes. Use SignThickness instead.

ConnectedDevice

Description
Returns the identifier of the currently connected ePad device. For example

{7FCD9512-8763-436E-8747-40972EE28EFD}: ePad
{98C174D3-6D2A-4509-96DB-D5B34FA7A561}: ePad-ink
{C1AF33B2-529E-4D2D-B885-DAF2780009EE}: ePadII
{AC57EF90-C108-417F-A43B-E01613A93BD5}: ePad-vision

DisplayName

Description
Sets the text to be displayed on the signature object. This text appears when the signature object is not signed.

AvailableStartY

Description
Gets the starting point on the Y co-ordinate system available for the Inking Region. The value returned by this property can be used when defining the Inking Region in the SetBkImageEx function. This property needs to be invoked after SetAffirmationText and SetUserInfo methods.

AvailableEndY

Description
Gets the ending point on the Y co-ordinate system available for the Inking Region. The value returned by this property can be used when defining the Inking Region in the SetBkImageEx function. This property needs to be invoked after SetAffirmationText and SetUserInfo methods.

EnableAntiAliasing

Description
Setting this property causes the signature drawing algorithm to use Anti-aliasing technique to smoothen the signature. SignThickness value will not be used when Anti-Aliasing is used.

SignThickness

Description
Setting this property causes the signature drawing algorithm to use the specified thickness to draw the signature. Anti-aliasing technique will not be used.

EnableWhiteSpaceRemoval

Description
By default, the value of this property is set to FALSE. When this setting is FALSE, the signature pad sends to the PC the entire image of the drawing surface of the pad. When this property is set to TRUE, the pad removes the white space around the signature drawn on the pad and sends only the rectangle that contains the signature information to the PC. Thus, when the PC displays the signature on a Signature Field, the signature typically appears larger and more legible (compared to no white space removal), especially in the usual case where the Signature Field in the document is small.

EnableMaxEnlargementFeature

Description
By default, the value of this property is set to FALSE. This property lets the user to specify if the Maximum Enlargement Feature should be enabled or not. The Maximum Enlargement feature is used only in cases where the size of signature on the signature pad (after white space removal) is smaller than the size of the Signature Field in the application. When the signature drawn on the signature pad is smaller than the size of the Signature Field in the document, IntegriSign will enlarge the signature to fill the Signature Field. However, to limit the enlargement the user can specify a limit to the enlargement as described in the next property.

MaxEnlargementFactor

The input variable type for this property is double and the value can range between 1.0 to 2.0. If the **EnableMaxElargementFeature** property explained above is set to TRUE and this value is not set, then a default value of 1.25 is considered.

Description
The Maximum Enlargement factor is used only in cases where the size of signature on the signature pad (after white space removal) is smaller than the size of the Signature Field in the application. If the Maximum Enlargement Factor is 1.50 then the signature in the document will be no more than 50% larger than the signature captured on the signature pad.

Both EnableMaxEnlargementFeature and MaxEnlargementFactor are considered only when EnableWhiteSpaceRemoval is set to TRUE.

Events Supported

The developer may handle/override the following events from IntegriSign Component.

DblClick
MouseDown
MouseMove
MouseUp

SignCompleteStatus (SignStatus as Long)

Description
Event indicating whether the signature field was signed or not. If the signing process is incomplete (i.e. cancelled) SignStatus is 0 else SignStatus is 1.

This event is fired only when the StartSign method uses a separate signature capture dialog to sign. Hence, event fires when the second parameter of StartSign method is set to "1".

OnFirstTouch ()

Description
Event fires when the user first touches the device (i.e. when the user starts inputting the signature).

Signing Methods/Functions

SetSignerDetailsEx

Description

The Signer details like Name, Organization etc. can be set using this method. This information is set so that it can be used as signer reference. This method, if implemented, should be called before the **StartSign** method to define 'who' is signing.

Parameters

Name (In) – Data type String - Name of the signer

OrgUnit (In) – Data type String - Organization unit of the signer OrgName (In) –

Data type String - Organization name of the signer City (In) – Data type String -

Location of the signer

State (In) – Data type String – State

Country (In) – Data type String - Country of the signer

eMail (In) – Data type String – Signer eMail address

SignLocation (In) – Data type String - Location from where the signer is signing.

Note: All the Parameters can take NULL values.

Return Value

This function returns nothing.

GetSignerDetailsEx

Description

This method returns the Signer details stored in the signature component.

Parameters

Name (In/Out) – Data type String - Name of the signer

OrgUnit (In/Out) – Data type String - Organization unit of the signer

OrgName (In/Out) – Data type String - Organization name of the signer

City (In/Out) – Data type String - Location of the signer

State (In/Out) – Data type String – State

Country (In/Out) – Data type String - Country of the signer

eMail (In/Out) – Data type String – Signer eMail address

SignLocation (In/Out) – Data type String - Location from where the signer is signing.

Note: All the Parameters can take NULL values.

Return Value

Returns values are set through the parameters in the above order.

StartSign

Description

Invokes Signature capture Dialog and user can start the "act of signing".

Parameters

bs – Type ButtonStyle – To specify the type of buttons required.

The Styles available are BT_OK, BT_OK_CLEAR, BT_OK_CANCEL, BT_OK_CANCEL_CLEAR (default button style) and BT_NONE.

SignDlgReq – Type Integer – to specify whether separate Signing dialog is required or not

There are two possible values 0 (signing without No dialog) and 1 (Signing with Dialog).

Return Value

StartSign method returns following values depending on the hotspots clicked.

BT_PRESSED_OK
BT_PRESSED_CANCEL

GetSignData

Description

This function returns the signature data in an encrypted Base64string Format. If not signed, it returns a null string.

Parameters

This takes no inputs.

Return Value

Signature data in encrypted Base64String format.

OpenSign

Description

Use this function to open and view the signature from the encrypted Base64String on the signature component.

Parameters

StrData – An IntegriSign Pro signature string of encrypted Base64String Format.

Return Value

This function returns nothing.

*Note: Method **GetSignData** was used originally to generate encrypted base64string from the signature component.*

ClearSign

Description

This clears the signature on the component, if any.

Parameters

This takes no inputs.

Return Value

This function returns nothing.

IsSigned

Description

Indicates whether the component is signed or not.

Parameters

This takes no inputs ***Return Value*** Returns a Long.

0 – The component is not signed

1 – The component is signed.

SaveToFile

Description

Saves the signature as an image file (bmp, jpeg, gif) in the set location.

Parameters

FileName – Data type String - Complete file Path where the bmp image is to be saved.

nHeight – Data type Integer - Height of the bmp image in pixels.

nWidth – Data type Integer - Width of the bmp image in pixels.

FileType **-** Data type FILETYPE – Indicates type of file (image) storage. (BMP =

0, JPEG =1, GIF=2)

ImageQuality – Data type Integer – For image quality (jpeg). An Optional parameter.

GIFTransparency – Data type Integer – For Non-transparent GIF the value is 0 and for transparent GIF the value is 1. An Optional parameter.

Note: ImageQuality should be between 0 and 100. If the parameter is not set or is set to zero ImageQuality is taken as 80 by default.

Return Value

This returns nothing.

VerifyContents

Description

Performs the hashing operation and returns the content integrity status. **HashData** property has to be set before calling this method.

Parameters

This takes no inputs.

Return Value

0 – Hashed content has not changed since signing.

1 – Hashed content was modified after signing.

2 – Hashing was not implemented.

GetImageData

Description

Returns encoded (Base64) Signature image data for storage as string.

Parameters

nHeight – Data type Integer - Height of the bmp image in pixels.

nWidth – Data type Integer - Width of the bmp image in pixels.

FileType **-** Data type FILETYPE – Indicates type of file (image) storage. (BMP = 0, JPEG =1, GIF=2)

ImageQuality – Data type Integer – For image quality (jpeg). An Optional parameter.

GIFTransparency – Data type Integer – For Non-transparent GIF the value is 0 and for transparent GIF the value is 1. An Optional parameter.

Note: ImageQuality should be between 0 and 100. If the parameter is not set or is set to zero ImageQuality is taken as 80 by default.

Return Value

ImageData – Data type String - Encoded (Base64) Signature image data for storage.

GetImageBytes

Description

Returns Signature image data for storage as a Byte array.

Parameters

nHeight – Data type Integer - Height of the bmp image in pixels.

nWidth – Data type Integer - Width of the bmp image in pixels.

FileType **-** Data type FILETYPE – Indicates type of file (image) storage. (BMP =

0, JPEG =1, GIF=2)

ImageQuality – Data type Integer – For image quality (jpeg). An Optional parameter.

GIFTransparency – Data type Integer – For Non-transparent GIF the value is 0 and for transparent GIF the value is 1. An Optional parameter.

Note: ImageQuality should be between 0 and 100. If the parameter is not set or is set to zero ImageQuality is taken as 80 by default.

Return Value

ImageBytes – Data type ByteArray - Signature image bytes for storage.

SetSignAppearanceOptionsEx

Description

This function lets you specify the information to be displayed and printed along with the signature.

Parameters

PenColor (In) – Data type Long

ShowName (In) – Data type Integer

ShowDate (In) – Data type Integer

ShowLogo (In) – Data type Integer

ShowLocation (In) – Data type Integer

ShowLabels (In) – Data type Integer

For all the above parameters a value of 1 means true and 0 means false.

Return Value

This function does not return any value.

SetBkImage

Description

You can specify the bitmap and the coordinates for ink region (when user scribbles) using SetBkImage method. This method should be called before StartSign method.

Parameters

Filename (In) – Data type String – Complete path of the image file (including name) to be set for display on the ePadInk device.

The selected bitmap should be a monochrome image with 320 X 200 resolution.

LeftX (In) – Data type Integer - Left X co-ordinate of the area

TopY (In) – Data type Integer - Top Y co-ordinate of the area.

RightX (In) – Data type Integer - Right X co-ordinate of the area.

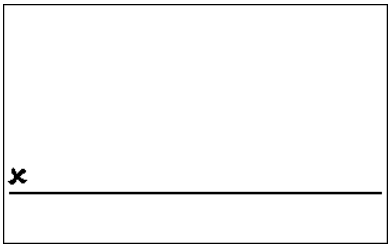
BottomY (In) – Data type Integer - Bottom Y co-ordinate of the area.

Return Value

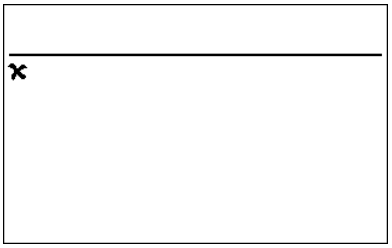
This function does not return any value.

Image Orientation:

As explained above, the image should be a monochrome image with 320 * 200 pixels and it should be flipped vertically. For example, if you want the image to be displayed on the device as



the original image should be as shown below.



NOTE: It is recommended to use the SetBkImageEx method instead of the SetBkImage method. SetBkImage method is maintained to support legacy applications developed using versions of IntegriSign Desktop before 7.5.4.

GetDevices

Description

Enumerates the all the installed device providers and returns an array containing their information i.e. ClassID and Description.

Parameters

No input parameters

Return Value

String Array, containing the list of providers.

setePadDevice

Description

Sets the device for initiating the act-of-signing. The device to be set is selected from the list of installed providers.

Parameters

ClassID of the device as String.

Return Value

No return value.

GenerateImageFromBase64

Description

Generates the signature image from the supplied Base64 image string.

Parameters

ImageStr – Image data as Base64 string.

FilePath – File Path including the name where the image needs to be saved.

Return Value

Long – Indicates whether the image is saved successfully.

0 – Success 1 – Failure

GenerateImageFromBinary

Description

Generates the signature image from the supplied binary data.

Parameters

ImageData – Image data as a byte array.

FilePath – File Path including the name where the image needs to be saved.

Return Value

Long – Indicates whether the image is saved successfully.

0 – Success 1 – Failure

CloseConnection

Description

Closes the connection opened using StartSign method.

Parameters

No Parameters

Return Value

No return value.

SetSignerDetails

Description

The Signer details like Name, Organization etc. can be set using this method. This information is set so that it can be used as signer reference. This method, if implemented, should be called before the **StartSign** method to define 'who' is signing.

Parameters

Name (In) – Data type String - Name of the signer

OrgUnit (In) – Data type String - Organization unit of the signer

OrgName (In) – Data type String - Organization name of the signer

City (In) – Data type String - Location of the signer

State (In) – Data type String – State

Country (In) – Data type String - Country of the signer

eMail (In) – Data type String – Signer eMail address

SignLocation (In) – Data type String - Location from where the signer is signing.

bReserved(In) – Data type String - reserved parameter.

Note: All the Parameters can take NULL values.

Return Value

This function returns nothing.

NOTE: It is recommended to use the SetSignerDetailsEx method instead of the SetSignerDetails method. SetSignerDetails method is maintained to support legacy applications.

GetSignerDetails

Description

This method returns the Signer details stored in the signature component.

Parameters

Name (In/Out) – Data type String - Name of the signer

OrgUnit (In/Out) – Data type String - Organization unit of the signer

OrgName (In/Out) – Data type String - Organization name of the signer

City (In/Out) – Data type String - Location of the signer

State (In/Out) – Data type String – State

Country (In/Out) – Data type String - Country of the signer

eMail (In/Out) – Data type String – Signer eMail address

SignLocation (In/Out) – Data type String - Location from where the signer is signing.

bReserved (In/Out) – Data type String – reserved parameter.

Note: All the Parameters can take NULL values.

NOTE: It is recommended to use the GetSignerDetailsEx method instead of the GetSignerDetails method. GetSignerDetails method is maintained to support legacy applications.

SetSignAppearanceOptions

Description

This function lets you specify the information to be displayed and printed along with the signature.

Parameters

PenColor (In) – Data type Long

ShowName (In) – Data type Integer

ShowDate (In) – Data type Integer

ShowLogo (In) – Data type Integer

sReserved (In) – Data type Integer. A reserved parameter.

ShowLocation (In) – Data type Integer

ShowLabels (In) – Data type Integer

For all the above parameters a value of 1 means true and 0 means false.

Return Value

This function does not return any value.

NOTE: It is recommended to use the SetSignAppearanceOptionsEx method instead of the SetSignAppearanceOptions method. SetSignAppearanceOptions method is maintained to support legacy applications.

SetAffirmationText

Description

Sets the affirmation text to be displayed on the ePadInk device during signature capture.

Parameters

Affirmation Text – Type String

Return Value

Returns the error number as Integer.

0 – No error

11 – Text length is too long

12 – More number of lines are required to display the text than supported 6 lines.

13 – No Data

Note: Number of lines supported for Affirmation Text on the ePadInk device is 6. If affirmation text needs to be displayed in multiple lines as intended by the application, insert "\n" wherever required. If "\n" is not found in the text, the algorithm in the component breaks the data into multiple lines accordingly.

SetUserInfo

Description

Sets the User Information to be displayed below the signing area on the ePadInk device during signature capture.

Parameters

User Info – Type String

Return Value

Returns the error number as Integer.

0 – No error

11 – Text length too long

12 – More number of lines are required to display the text than supported 2 lines.

13 – No Data

Note: Number of lines supported for User Information on the ePad-ink device is 2. If User Information needs to be displayed in multiple lines as intended by the application, insert "\n" wherever required. If "\n" is not found in the text, the algorithm in the component breaks the data into multiple lines accordingly.

SetBkImageEx

Description

Sets the bitmap and the coordinates for ink region (when user scribbles) using SetBkImageEx method. This method should be called before StartSignEx method and after SetAffirmationText, SetUserInfo methods. If the last optional parameter FolderName is not specified, by default the folder is assumed as Windows system folder.

Parameters

Filename (In) – Data type String – Image file name excluding path. Currently only BMP images are supported.

The selected bitmap should be a monochrome image.

StartX (In) – Data type Integer - Starting X co-ordinate of the area- Value can be between 0 and 320.

StartY (In) – Data type Integer – Top Left Y co-ordinate of the area. Value should be in between the values obtained from properties AvailableStartY and AvailableEndY.

EndX (In) – Data type Integer - Ending X co-ordinate of the area- Value can be between 0 and 320.

EndY (In) – Data type Integer – Bottom Right Y co-ordinate of the area. Value should be in between the values obtained from properties AvailableStartY and AvailableEndY.

The above four parameters define the Inking Region on the screen.

FolderName (In) – Optional Parameter – Folder Path of the Image file if it is other than Windows System folder.

Return Value

Returns the error number as Integer.

0 - No error

1 - The specified Image File not found

2 - Invalid Image

3 - The specified image is not a monochrome image.

4 - Image height is greater than the available inking area.

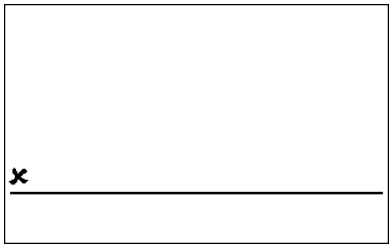
5 - Image Width greater than the available inking area.

6 - Invalid ink region specified.

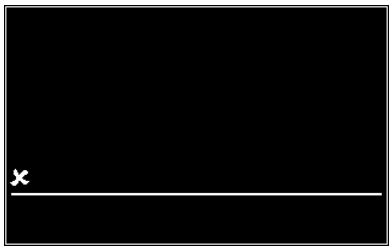
Image Orientation:

ePad-ink:

The image should be a monochrome image. The colors in the image should be inverted. For example, if you want the image to be displayed on the device as



the original image should be as shown below.



ePad-vision:

Currently ePad-vision does not support this method. Even if this method used, the image specified would not be displayed on the device.

StartSignEx

Description

Invokes the "act of signing".

Parameters

BUTTON_TYPE_EX bTypeEx – Type ButtonStyle –

Possible values for this parameter are

BT_One_Left = 0
BT_One_Righ = 1
BT_One_Center = 2
BT_Two_Left = 3
BT_Two_Right = 4
BT_Two_Cente = 5
BT_Three = 6

Captions – String containing the Captions for the buttons. The names of the buttons are to be separated by "~". Example : "Cancel~Clear~Accept"

Return Value

StartSignEx method returns the name of the button that was clicked by the User. For Example, Cancel or Clear or Accept.

SetSignRect

Description

Sets the percentage of the signature object to be used for drawing the signature.

Parameters

WidthPercent – Percentage of Signature Object Width to be used. Default value is 80.

HeightPercent – Percentage of Signature Object Height to be used. Default value is 80.

Return Value

No Return Value.

Note: This method is obsolete and is there only for compatibility purposes.

SaveSigImage

Description

Saves the signature as an image file (bmp, jpeg, gif) in the set location.

Parameters

FileName – Data type String - Complete file Path where the bmp image is to be saved.

nHeight – Data type Integer - Height of the bmp image in pixels.

nWidth – Data type Integer - Width of the bmp image in pixels.

FileType **-** Data type Integer – Indicates type of file (image) storage. (BMP = 0,

JPEG =1, GIF=2)

ImageQuality – Data type Integer – For image quality (jpeg). An Optional parameter.

GIFTransparency – Data type Integer – For Non-transparent GIF the value is 0 and for transparent GIF the value is 1. An Optional parameter.

Note: ImageQuality should be between 0 and 100. If the parameter is not set or is set to zero ImageQuality is taken as 80 by default.

Return Value

This returns nothing.

esCapSvr Component

This component is used to decode the Base64 string into byte array. This component is useful on the web server to decode and write the image data into a file.

Signing Methods/Functions

SaveToFile

Description

Decodes and Saves the signature as an image file (bmp, jpeg) in the set location.

Parameters

ImageData – Data type String – Encoded signature data.

FileName – Data type String - Complete file Path with proper extension where the image (BMP or Jpeg) is to be saved.

Note: All the Parameters can take NULL values.

Return Value

Returns a Short.

0 – Failed to write into the file.

1 – Base64 Image data saved as an Image file.