

Topaz Systems, Inc.
SigID™ Fingerprint Capture and Validation Active-X Control
Version 1.0.0.1 - 4/16/2004

Topaz Systems, Inc.
650 Cochran Street, Suite 6
Simi Valley, CA, 93065
805 520-8282 (Main)
805 520-0867 fax

Tech support:

805 520-8286

805 520-8282

www.topazsystems.com

support@topazsystems.com

Copyright 2004, all rights reserved

SigID™ Fingerprint Capture and Comparison System
A member of the GemTools family.

INSTALLATION NOTE: The control should be properly setup and registered when running the setup program. To register the control manually, copy SigID.ocx into the c:\windows\sigplus directory. Then run regsvr32 c:\windows\sigplus\SigID.ocx. Or, copy SigID.ocx into the c:\winnt\sigplus directory. Then run regsvr32 c:\winnt\sigplus\SigID.ocx

DESCRIPTION:

SigID.ocx is an Active-X control that allows the developer to capture encrypted fingerprint templates, store them, and use them later to compare against newly-captured fingerprints on-the-fly. Each fingerprint template is automatically encrypted prior to its delivery to the developer.

NOTE: It's important to understand there are two possible ways to use the SigID ActiveX control:

1. The developer is handed back the fingerprint template with which to compare later fingerprints in a portable file format (.TFT) or ASCII hex string (for easy database storage). This allows maximum flexibility in terms of storing fingerprint templates, but certain fingerprint comparisons are not possible using this approach (namely one to many searches).
2. A proprietary, encrypted database (essentially a DAT file) is used to store the fingerprint templates. Whereas this approach limits the storage flexibility of approach 1, it maximizes fingerprint comparisons in terms of speed and possible search options (namely one to many searches).

These approaches are mutually exclusive, and the developer must decide which approach best suits the application. Methods beginning with the prefix **pr (portable)** signify methods falling under approach #1, the most portable option. Methods beginning with the prefix **db (database)** signify methods falling under approach #2, the maximized proprietary database option. Methods that begin with neither prefix are neutral methods, and are not dependent upon one approach or another.

TECH SUPPORT

Call (805) 520-8286 or (805) 520-8282 for Technical support

Be sure to see <http://www.topazsystems.com/Software/faq.htm> for the most frequently asked questions and answers to software and hardware related issues.

**FOR A VB6 DEMO USING SIGID.OCX, PLEASE SEE YOUR Win\SigPlus\SigID\Proj
DIRECTORY AFTER INSTALLATION.**

Properties:

General Methods:

The following methods apply to both fingerprint template storage methods (see p.1 for details)

.SensorState(intState) As Integer

This method opens or closes the USB port for the fingerprint sensor based upon the integer passed in. 0=close sensor, 1=open sensor.

RETURN: Integer. 0=success, 1=unhandled error, 2=cannot find device

```
Dim intRet As Integer
intRet = SigID1.SensorState(1)

If intRet = 0 Then
    'success
ElseIf intRet = 1 Then
    'error occurred
ElseIf intRet = 3 Then
    'cannot locate device...not plugged in
End If
```

.SetResultDelay(long milliseconds)

This method sets the amount of delay (in milliseconds) that the guided dialogs will sleep after completing a task (applies to the following methods):

DEFAULT (when not set): 3000 milliseconds

Example:

```
SigID1.SetResultDelay 1500 'sets 1.5 second delay
```

.GetResultDelay() As Long

This method returns the current amount of delay (in milliseconds) that the guided dialogs will sleep after completing a task.

RETURN: Long: current value (in milliseconds) of the delay

Example:

```
Dim intRet As Integer
intRet = SigID1.GetResultDelay
'intRet now holds value for current millisecond delay
```

Portable Template (PR) Methods:

The following methods apply to the “portable” approach to fingerprint template storage (see p.1 for details)

.prExportTempFile(strPath, intVal) As Integer

This method enrolls a user's fingerprint and saves it to a file. The user is guided by various dialogs. The first argument is the path, filename and .dat extension of the file to save as. The second argument is currently unused; please pass a 0.

RETURN: Integer. 0=save successful, 1=save unsuccessful, 2=user canceled

Example:

```
intVal = SigID1.ExportTempFile("C:\somefolder\somefile.dat", 0)
```

```
If intVal = 0 Then
    'saved to file successful
ElseIf intVal = 1 Then
    'unsuccessful save
ElseIf intVal = 2 Then
    'user canceled dialog
End If
```

.prValidateTemplateFile(strPath) As Integer

This method captures the user's fingerprint and compares it to a saved fingerprint template. It takes one argument, the path to the saved fingerprint file to use for comparison. The user is guided by automatic dialogs.

RETURN: Integer: 0=validation success, 1=validation fail, 2=user canceled

Example:

```
intRet = SigID1.ValidateTemplateFile("C:\somefolder\somefile.dat")
If intRet = 0 Then
    'MATCH SUCCESSFUL
ElseIf intRet = 1 Then
    'NO MATCH
ElseIf intRet = 2 Then
    'USER CANCELED DIALOG
End If
```

.prGetFingerprintString() As String

This method captures the user's fingerprint file, returning the fingerprint file converted to a long ASCII hex string. This string can be stored as necessary (for example in a database) and returned later for comparison. The user is guided by automatic dialogs.

RETURN: String: ASCII hex representation of the fingerprint template file

Example:

```
Dim strFingerprint As String
strFingerprint = SigID1.GetFingerprintString
If strFingerprint <> "" Then
    'fingerprint string captured ok
Else
    'user canceled dialog
End If
```

.prValidateFingerprintString(strFingerprint) As Integer

This method takes the result of a call to GetStringFingerprint and compares it to a newly captured fingerprint. The user is guided by automatic dialogs.

RETURN: Integer: 0=validation success, 1=validation failure, 2=user canceled

Example:

```
Dim intAns As Integer
intAns = SigID1.ValidateFingerprintString(strFingerprint)

If intAns = 0 Then
    'MATCH SUCCESSFUL
Elseif intAns = 1 Then
    'NO MATCH
Elseif intAns = 2 Then
    'USER CANCELED DIALOG
End If
```

Maximized Speed/Search Template (DB) Methods:

The following methods apply to the “maximized” approach to fingerprint template storage (see p.1 for details)

.dbInit(string dbfilepath) As Integer

This method initializes the database. The user must pass the path to the DAT file to use. If the DAT file does not exist, SigID will create a new DAT file at the location specified.

RETURN: Integer: 0=successful open, 2=cannot locate and initialize fingerprint sensor

Example:

```
Dim RetVal As Integer
RetVal = SigID1.Init("C:\mydatabase.dat")

If RetVal = 0 Then
    'sensor initialized OK
Elseif RetVal = 2 Then
    'cannot locate fingerprint sensor
Exit Sub
End If
```

.dbListAllUsers() As String

This method returns all of the users in the currently-open database, in one comma delineated string. The developer will have to parse this string to separate the users.

```
Dim strUser As String
strUsers = SigID1.dbListAllUsers
'strUsers can now be parsed
```

.dbEnroll(string user)

This method enrolls a user's fingerprint and saves it to the currently-open DAT database. The user is guided by various dialogs. The method takes one string argument, the name of the user to enroll.

RETURN: Integer: 0=success, 1= failure, 2=User canceled operation

```
Dim retVal As Integer
retVal = SigID1..Enroll ("Some Name")
If retVal = 0 Then
    'success
ElseIf retVal = 1 Then
    'failure
ElseIf retVal = 2 Then
    'user canceled
End If
```

.dbValidateID() As String

This method takes a string argument, the name of a user in the database, captures a fingerprint based on automatic dialogs, and compares the new fingerprint against the user in the database to validate their identity.

RETURN: Integer: 0= user validated successfully, 1= failed to validate user, 2=User canceled operation, 3=unhandled exception

```
Dim retVal As Integer
retVal = SigID1.dbValidateID(strUserName)
If retVal = 0 Then
    'failed to validate
ElseIf retVal = 1 Then
    'validation success
End If
```

.dbIdentify() As String

This method takes a string argument, the name of a user in the database, captures a fingerprint based on automatic dialogs, and compares the new fingerprint against the user in the database to validate their identity. This method is tied to the **dbGetIdentifiedUser()** method, which returns the identified user (as a string) when dbIdentify() returns a 0 (successful identification). (See example below)

RETURN: Integer: 0=identified the user, 1=no match, 2=user cancelled operation

```
Dim retVal As Integer
retVal = SigID1.dbIdentify()
If retVal = 0 Then
    MsgBox "user is identified as " & SigID1.dbGetIdentifiedUser(True)
ElseIf retVal = 1 Then
    'no match in the database
ElseIf retVal = 2 Then
    'user closed (cancelled) the capture window
End If
```

.dbGetIdentifiedUser() As String

This method is called as a result of a successful return from **dbIdentify()**, in order to retrieve the successfully identified user. This method takes a Boolean argument, which

allows the return identified user value to persist until the next successful return from calling **dbIdentify()**, or deletes the value once it has been returned.

ARGUMENT: TRUE: persist identified user value, FALSE: eliminate value after call

RETURN: String: IdentifiedUser

[See dbIdentify\(\) for code example](#)

.dbDeleteUser(string username)

This method deletes the user specified in the string argument from the currently-open database.

[SigID1.dbDeleteUser strUserName](#)

.dbDeleteAllUsers()

This method deletes all users from the currently-open database.

[SigID1.dbDeleteAllUsers](#)

Fingerprint Imaging:

The following methods apply to the creation of fingerprint images (bmp, tif, jpg) only, and are not used for verification purposes

.BmpResolution(int resolutionsetting)

Sets the px width of the resulting bmp when WriteBmpFile() is called. There are three settings (at screen resolution):

0 = 96x96 px (DEFAULT)

1 = 144x144 px

2 = 192x192 px

Example:

[SigID1.BmpResolution 1 'will write bmp at 144x144](#)

[SigID1. WriteBmpFile\("c:\myfolder\myimage.bmp"\)](#)

.JpgResolution(int resolutionsetting)

Sets the px width of the resulting bmp when WriteJppFile() is called. There are three settings (at screen resolution):

0 = 96x96 px (DEFAULT)

1 = 144x144 px

2 = 192x192 px

Example:

[SigID1.JpgResolution 2 'will write bmp at 192x192](#)

[SigID1. WriteJpgFile\("c:\myfolder\myimage.jpg"\)](#)

.TifResolution(int resolutionsetting)

Sets the px width of the resulting tif when WriteTifFile() is called. There are three settings (at screen resolution):

0 = 96x96 px (DEFAULT)

1 = 144x144 px

2 = 192x192 px

Example:

```
SigID1.TifResolution 0 'will write bmp at 96x96, also the default  
SigID1. WriteTifFile("c:\myfolder\myimage.tif")
```

.WriteBmp(strFilePath) As Integer

This method creates a bitmap image of a fingerprint at the location specified through its argument, the path to a file. The user is guided by automatic dialogs. It takes a string argument, the path to the file you wish to save to.

RETURN: integer

0 - Success

1 - Unhandled Exception

2 - User closed dialog window (pressed the 'X')

3 - Poor image captured; image cannot be created

Example:

```
Dim intRetVal As Integer  
SigID1.BmpResolution 1  
intRetVal = SigID1.WriteBmp("C:\myimage.bmp")  
  
If intRetVal = 0 Then  
    'saved to file ok  
Elseif intRetVal = 1 Then  
    'unhandled exception  
Elseif intRetVal = 2 Then  
    MsgBox "You Pressed 'X'", vbOKOnly + vbExclamation, "User Canceled"  
Elseif intRetVal = 3 Then  
    MsgBox "Be sure to press finger firmly on the fingerprint scanner", vbOKOnly +  
        vbExclamation, "Poor Image Collected"  
End If
```

.WriteJpg(strFilePath) As Integer

This method creates a jpg image of a fingerprint at the location specified through its argument, the path to a file. The user is guided by automatic dialogs. It takes a string argument, the path to the file you wish to save to.

RETURN: integer

0 - Success

1 - Unhandled Exception

2 - User closed dialog window (pressed the 'X')

3 - Poor image captured; image cannot be created (it is suggested to close the USB port and reopen it before continuing (using **SensorState()**)

Example:

```
Dim intRetVal As Integer
SigID1.JpgResolution 0
intRetVal = SigID1.WriteJpg("C:\myimage.jpg")

If intRetVal = 0 Then
    'saved to file ok
ElseIf intRetVal = 1 Then
    'unhandled exception
ElseIf intRetVal = 2 Then
    MsgBox "You Pressed 'X'", vbOKOnly + vbExclamation, "User Canceled"
ElseIf intRetVal = 3 Then
    MsgBox "Be sure to press finger firmly on the fingerprint scanner", vbOKOnly +
        vbExclamation, "Poor Image Collected"
End If
```

.WriteTif(strFilePath) As Integer

This method creates a tif image of a fingerprint at the location specified through its argument, the path to a file. The user is guided by automatic dialogs. It takes a string argument, the path to the file you wish to save to.

RETURN: integer

0 - Success

1 - Unhandled Exception

2 - User closed dialog window (pressed the 'X')

3 - Poor image captured; image cannot be created

Example:

```
Dim intRetVal As Integer
SigID1.TifResolution 1
intRetVal = SigID1.WriteTif("C:\myimage.tif")

If intRetVal = 0 Then
    'saved to file ok
ElseIf intRetVal = 1 Then
    'unhandled exception
ElseIf intRetVal = 2 Then
    MsgBox "You Pressed 'X'", vbOKOnly + vbExclamation, "User Canceled"
ElseIf intRetVal = 3 Then
    MsgBox "Be sure to press finger firmly on the fingerprint scanner", vbOKOnly +
        vbExclamation, "Poor Image Collected"
End If
```

.WriteBmp(strFilePath) As Boolean

Not currently implemented.

.WriteJpg(strFilePath) As Boolean

Not currently implemented.

.WriteTif(strFilePath) As Boolean

Not currently implemented.

.GetBitmapBufferBytes(strFilePath) As Byte()

This method returns a buffered bitmap byte array, based on the size as set by the call to BMPResolution(). (Default is 96px X 96px).

Example:

```
Dim ByteValue() As Byte
```

```
SigID1.BmpResolution 0
```

```
ByteValue = SigID1.GetBitmapBufferBytes
```

```
'ByteValue now holds the buffered fingerprint bitmap
```

.AutoKeyStart(), .AutoKeyData, AutoKeyFinish(),EncryptionMode(), SecurityLevel() are currently not implemented.

Installation notes

Topaz Systems SigID post-installation directory structure is:

WIN

\SigPlus

[SigID.ocx \(the Fingerprint Capture and Validation Active-X Control\)](#)

\SigID

[SigID.exe \(The SigID demo using SigID.ocx\)](#)

\Docs [SigID.doc \(this development documentation\)](#)

\Proj [SigID.zip\(VB6 demo using SigID ActiveX control\)](#)

SigID DLL AND OCX List

VBRuntime files (for SigID.exe only)

ComDlg32.ocx

ATSC51.dll

ATInsmg.dll

FLMckusb.sys

FLMckusb.inf

Important Notices

License Agreement and Limited Warranty

IMPORTANT: Please read this document before continuing the software load procedure. By loading the software enclosed with this agreement, you are indicating acceptance of the terms of this legal agreement between you (herein call Licensee) and Topaz Systems, Inc. (herein called

Topaz). If you do not agree to the terms of this agreement, do not load the enclosed software and promptly return the product.

1. Limited use License: Topaz and its suppliers (if any) grant you the right to use the software for use with Topaz Gem-Series tablets only. The software is owned for distribution exclusively by Topaz and is protected by the United States Patent and Trademark laws and international treaties.

2. Governing Law, Jurisdiction, and Forum: This agreement is governed by the laws of the State of California, County of Ventura.

3. YOU MAY:

(a) Freely use, copy, and distribute the enclosed software only for use with Topaz Gem-Series tablets.

4. YOU MAY NOT:

(a) Use, distribute or modify the Topaz software for use with tablets not supplied by Topaz Systems, Inc.

(b) Reverse engineer, decompile, or dis-assemble, the software.

5. LIMITED WARRANTY:

Topaz Systems warrants to the original buyer only, that the media upon which the Software is recorded is free from defects in workmanship and material under normal use and service for a period of 30 days.

6. EXCLUSIVE REMEDY:

Topaz's entire liability and your exclusive remedy shall be, at Topaz's option, either (a) the repair or replacement of the software or (b) the refund of the price that was paid for the software package. The defective software, along with proof of payment, must be returned to Topaz within the terms of the warranty as set forth in this agreement.

7. LIMITATIONS ON DAMAGES:

In no event shall Topaz be liable for damages whatsoever, (including without limitation, damages for loss of profits, business interruption, loss of information, or other pecuniary loss) arising out of the use of or inability to use the software even if Topaz or its suppliers, if any, have been advised have been advise of the possibility of such damages. In no event will Topaz's liability for any reason exceed the actual price paid for the license to use the specific program.

8. NO OTHER WARRANTIES:

With respect to the software, media, and written material, Topaz and its suppliers, if any, disclaim all warranties, other than the above Topaz warranty, including but not limited to warranties merchantability or fitness for a particular use. Topaz does not warranty the software will meet or requirements or that the operations of the software will be uninterrupted or error free.

9. GOVERNING LAW AND GENERAL PROVISIONS:

If any part of this agreement is found void and unenforceable, it will not effect the validity of the balance of the agreement, which shall remain valid and enforceable according to its terms. You agree that the software will not be shipped, transferred, or exported into any country or used in any manner prohibited by the United States Export Administration Act or any other export laws, restrictions or regulations. This agreement shall automatically terminate upon failure of you to comply with its terms. This agreement may only be modified in writing signed by an officer of Topaz Systems, Inc. This agreement will not be governed by the United Nations convention on contracts for the International sale of goods, the application of which is expressly excluded.

Statement regarding patents, and warning against modification of Topaz products:

Topaz products are covered under US patents 5,120,908 and 5,122,623. Patent work is ongoing and pending. Use of Topaz's products in accordance with our instructions, to the best of our knowledge, does not infringe any patents. However, be aware that there are many patents out there, and modification of our products, or use of our products for other than their intended purpose, could run afoul of these patents. It is the responsibility of Topaz customers to honor relevant patents. U. S. Patent Nos. 5,544,255; 5,647,017; 5,818,955 and others are patents that users of Topaz products should consider if modifications are to be made to our products or if our products are to be used for other than their intended purpose. These patents may be obtained at <http://www.uspto.gov>. Only as examples, without first considering these patents, you should refrain from storing within the signature information (sig file or sig data):

1. A reason for signing or a statement of importance of the signature together with a set of measurements relating to the handwritten signature and either an indication of the signatory or means for comparing said measurements with a set of statistics of a genuine signature to obtain a similarity score.
2. A visual representation of a signature together with a document checksum, hash, or receipt, and the claimed identity of the signatory.
3. A set of measurements relating to biometric information together with a document checksum, hash, or receipt, and the claimed identity of the user.

The foregoing statement is not intended to be an interpretation of the patents or their application to any particular product or implementation.

It may be possible that Federal (FDA) and state digital signature regulations become violated if techniques in 1, 2, and 3 above are employed, so rest assured, Topaz software does not do any of them. Note the information below:

1. The FDA regulations state that "Signed electronic records shall contain information associated with the signing that clearly indicates all of the following ...printed name ... date and time ... The meaning ...". (see FDA guidelines section 11.50) The FDA regulations treat electronic records and electronic signatures as different entities (Section 11.70). Therefore, you should refrain from modifying the Topaz system on your own to store any of the name, date and meaning data in the signature file.
2. State regulations such as CA code of Regulations, Title 2, Division 7, Chapter 10 for example, require that the signature be bound to only the single message that is signed and not to any other message. Therefore you should refrain from modifying the Topaz system on your own to place a checksum in with the signature data and then adding other information in with the signature, and then encrypting them to a secret key. If you were to modify the Topaz system to do this, you would be binding the signature to multiple-messages which is prohibited by the state regulations.
3. State regulations require that the signature data be capable of verification by a forensic document examiner. (CA regulations paragraph 220003(b)(3)(B) for example). Some of the techniques in 1-3 above appear at odds with "lengthy process of handwriting analysis" required by state regulation (see state of CA FAQ section). With the Topaz system, you are assured of the most accurate forensic handwriting analysis results, since the Topaz .sig file is saving all of the original signature data.

In addition, there are potential security and refutability issues if too many things are crammed in with the signature. Leave the document data, reason for signing, receipt, and identity information in the document as GemTools is designed to work, and as complies with FDA and state regulations. That way, the signature is bound to all of the data, not just to part of the data. It is very easy to create all kinds of standard image files of the signature using GemTools software. Topaz signature & document security results from our patent pending methods of encrypting the .sig data directly to a document hash (not by putting the hash or receipt into the .sig file), and non-repudiation results from the proper use of (again patent pending) Topaz signature and document receipts. Please contact the factory if you have any additional questions or would like more information about your rights concerning patents as a customer or developer using Topaz products.

Important Notice:

This software or any or all additional documentation, guidelines, or examples do not constitute a warranty about the performance, security, or legal acceptability of GemTools software or the SigPlus control in any specific use or implementation. To the extent that GemTools or SigPlus are used to achieve regulatory or other specific objectives within an industry, you must consult competent experts or regulatory officials together with your own plan to achieve your desired business objectives using the Topaz tools.