

**Topaz Systems, Inc.**  
**SigCompare™ Signature Comparison Active-X Control**  
Covered under US Patent 6,307,955 - Electronic Signature Management System  
Version 1.0 - 11/19/2004

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](http://www.topazsystems.com)  
support@topazsystems.com

**Copyright 2004, all rights reserved**

A member of the Topaz SigPlus® family.  
SigCompare™ Signature Comparison ActiveX Control System

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

DESCRIPTION:

SigCompare.ocx is an Active-X control that allows the developer to compare, side-to-side, two Topaz signatures. The primary purpose of SigCompare is to bring robust signature authentication capability into everyday applications. SigCompare is covered by US Patent 6,307,955. In addition to a side-by-side display of signatures, this tool provides 6 additional levels of "Dynamic Shading TM" Dynamic shading is used to redraw the signature with ink thickness based upon capture velocity. Areas where the author signed slower appear thicker. Areas where the velocity of the pen was increased appear thinner. Velocity plays a key role in signature biometrics, and because SigCompare provides visual cues to the signatures velocity, signature comparison speed and accuracy is greatly increased.

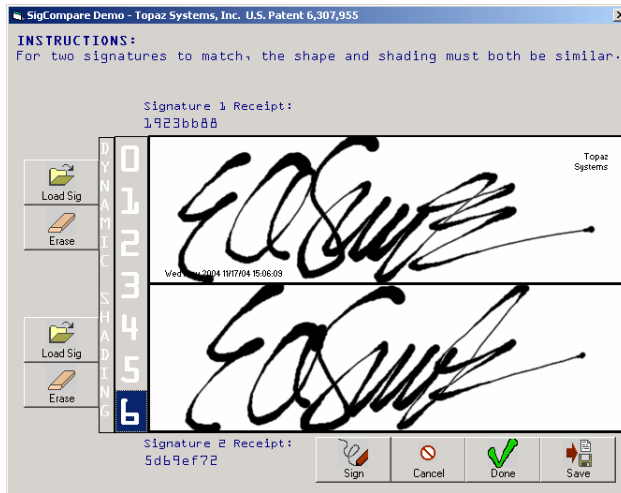
TECH SUPPORT

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 SIGCOMPARE.OCX, PLEASE SEE YOUR  
WIN\SigPlus\SigCompare\Proj DIRECTORY AFTER INSTALLATION.**

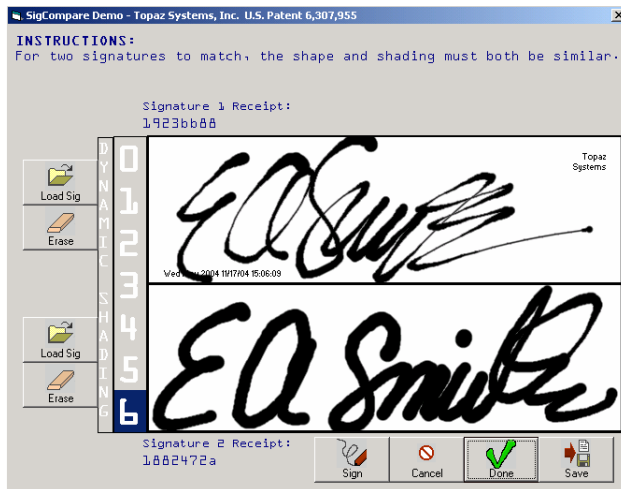
**Covered under US Patent 6,307,955 - Electronic Signature Management System**  
You may review this patent at [www.uspto.gov](http://www.uspto.gov)

## Example 1. True Signer



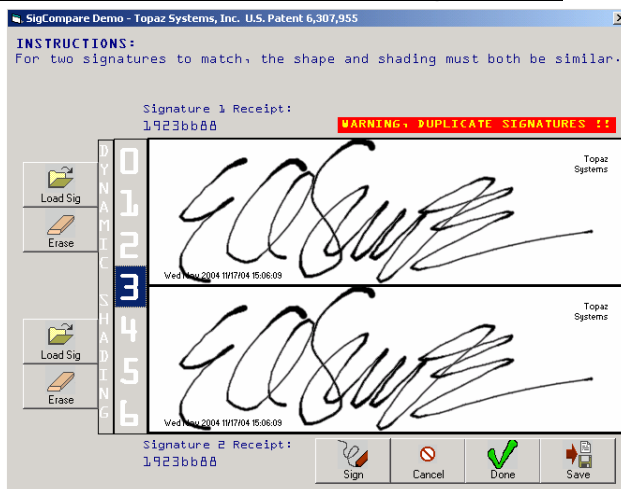
The upper reference signature is used to compare the authenticity of the lower questioned signature. The lower signature is very likely to be true because it is similar in shape and dynamic shading to the reference; but not identical to the reference.

## Example 2. Likely forgery



The upper reference signature is used to compare the authenticity of the lower questioned signature. The lower signature is very likely to be false, because while similar in shape, it exhibits the "slow signing effect" typical of the actions of a forger.

## Example 3. Duplicate Signature



The upper reference signature is used to compare the authenticity of the lower questioned signature. The lower signature is a duplicate or copy and therefore is clearly not representative of a unique act-of-signing event. This copy is detected by comparing signature receipts.

## **Properties:**

### **.DynamicShadingLevel =**

Integer. (0 to 6)

There are 6 dynamic shading levels, and an "off" position (0)

The shading levels 1, 2 and 3 are standard shading. 1 is best for slower-captured signatures, 2 for medium-speed-captured signatures, and 3 if for high-velocity signatures. Levels 4, 5 and 6 are exaggerated; the thickness of slower-velocity portions is increased, thus widening the difference between the thinner and thicker portions of the signatures.. 4 is best for slower-captured signatures, 5 for medium-speed-captured signatures, and 6 if for high-velocity signatures. This should be used if the difference when using levels 1, 2 and 3 are not great enough for useful comparison.

Thicker areas indicate slower pen velocity, and thinner areas indicate higher pen velocity.

DEFAULT: .DynamicShadingLevel = 0

Example:

[SigCompare1.DynamicShadingLevel = 3 'set to level 3](#)

## **Methods:**

### **.ClearTablet(TabletIndex As Integer)**

This method clears the signature from signature 1 or 2, depending upon the TabletIndex argument.

RETURN: None

Example:

[SigCompare1.ClearTablet\(1\) 'clears signature 1](#)

[SigCompare1.ClearTablet\(2\) 'clears signature 2](#)

### **.ExportSigFile(SigPath As String, CompressionMode As Integer, SigIndex As Integer) As Boolean**

This method exports the current signature as a Topaz .SIG file:

Arguments:

SigPath: The path to the location you wish to save the .SIG file

CompressionMode: Compresses the signature at the mode supplied (see SigPlus.doc, SigCompressionMode (0 is no compression)

SigIndex: 1=signature1, 2=signature2 (as displayed in the SigCompare control)

RETURN: Boolean, whether the .SIG file was written successfully

Example:

[Dim blnRet As Boolean](#)

[blnRet = SigCompare1.ExportSigFile \("C:\myfolder\mysignature.sig", 0, 1\)](#)

### **.ExtractSigString(CompressionMode As Integer, SigIndex As Integer) As String**

This method returns a Topaz SigString (see SigPlus.doc, SigString property):

Arguments:

CompressionMode: Compresses the signature at the mode supplied (see SigPlus.doc, SigCompressionMode (0 is no compression)

SigIndex: 1=signature1, 2=signature2 (as displayed in the SigCompare control)

RETURN: String, Topaz SigString

Example:

```
Dim strSigString As String  
strSigString = SigCompare1.ExtractSigString (1, 2) 'mode 1 compression, 2nd signature
```

**.LoadSigFile(SigPath As String, CompressionMode As Integer, EncryptionMode As Integer, EncryptionType As Integer, EncryptionString As String, SigIndex As Integer) As Boolean**

This method imports a Topaz .SIG file into one of the signature blocks in the SigCompare control

Arguments:

SigPath: The path to the location of the .SIG file you wish to import

CompressionMode: If the .SIG file has been compressed, you must pass in the mode at which it has been compressed (see SigPlus.doc, SigCompressionMode)

EncryptionMode: If the .SIG file has been encrypted, you must pass in the mode at which it has been encrypted (see SigPlus.doc, EncryptionMode)

EncryptionType: Whether you have encrypted the .SIG file with an external document, or by passing in string values. This is related to whether you did or did not call

AutoKeyStart() when the .SIG file was encrypted. Calling AutoKeyStart() means you have encrypted the .SIG file to literal string values. Not calling AutoKeyStart() means you have encrypted the .SIG file by passing in a path to a file. If you encrypted using literal string values, this is type 0. If you encrypted by passing in a path to a file, this is type 1.

EncryptionString: The string used to originally encrypt your .SIG file, whether it was a literal string, or a path to a file. If you used multiple strings (calling AutoKeyData more than once) be sure to concatenate your strings to form one string to pass in here.

SigIndex: 1=signature1, 2=signature2 (as displayed in the SigCompare control)

RETURN: Boolean, whether the .SIG file was loaded successfully

Example:

```
Dim strMyEncString As String  
Dim blnRet As Boolean
```

```
strMyEncString = 'obtain this value as necessary
```

```
blnRet = SigCompare1.LoadSigFile("C:\myfolder\mysignature.sig", 1, 2, 0,  
strMyEncString, 1)
```

'this will load a .SIG file from the location specified. This .SIG file was compressed at 'Mode1, Encrypted at Mode2, was encrypted using a literal string. The string used to 'encrypt it originally is kept in the strMyEncString variable, and this .SIG file will be loaded 'into the 1<sup>st</sup>-position signature in SigCompare

### **.LoadSigString(SigString As String, CompressionMode As Integer, EncryptionMode As Integer, EncryptionType As Integer, EncryptionString As String, SigIndex As Integer) As Boolean**

This method imports a Topaz SigString into one of the signature blocks in the SigCompare control

Arguments:

SigString: The original SigString you wish to import (see SigPlus.doc, SigString property)

CompressionMode: If the SigString has been compressed, you must pass in the mode at which it has been compressed (see SigPlus.doc, SigCompressionMode)

EncryptionMode: If the SigString has been encrypted, you must pass in the mode at which it has been encrypted (see SigPlus.doc, EncryptionMode)

EncryptionType: Whether you have encrypted the SigString with an external document, or by passing in string values. This is related to whether you did or did not call

AutoKeyStart() when the SigString was encrypted. Calling AutoKeyStart() means you have encrypted the SigString to literal string values. Not calling AutoKeyStart() means you have encrypted the SigString by passing in a path to a file. If you encrypted using literal string values, this is type 0. If you encrypted by passing in a path to a file, this is type 1.

EncryptionString: The string used to originally encrypt your SigString, whether it was a literal string, or a path to a file. If you used multiple strings (calling AutoKeyData more than once) be sure to concatenate your strings to form one string to pass in here.

SigIndex: 1=signature1, 2=signature2 (as displayed in the SigCompare control)

RETURN: Boolean, whether the SigString was imported successfully

Example:

```
Dim mySigString As String  
Dim blnRet As Boolean
```

```
strSigString = 'your original SigString  
blnRet = SigCompare1.LoadSigFile(strSigString, 0, 0, 0, "", 2)
```

'this will load the SIG string from the mySigString variable. This SigString was uncompressed and unencrypted. This SigString will be loaded 'into the 2<sup>nd</sup>-position signature in SigCompare

### **.RefreshSignature(SigIndex As Integer)**

This method forces a refresh of the specified signature; useful for refreshing the signature after a new dynamic shading mode is set.

Arguments:

SigIndex: 1=signature1, 2=signature2 (as displayed in the SigCompare control)

RETURN: None

Example:

```
SigCompare1.RefreshSignature 1 'refreshes the first signature
```

### **.GetSigReceipt (SigIndex As Integer) As String**

This method returns an 8-character hash of the signature, used for easy comparison purposes. If the SigReceipt for both displayed signatures match, you have identical signatures loaded.

Arguments:

SigIndex: 1=signature1, 2=signature2 (as displayed in the SigCompare control)

RETURN: String – 8-character hash of the signature

Example:

```
Dim strSigReceipt As String  
strSigReceipt = SigCompare1.GetSigReceipt
```

### **.SetTabletState(SigIndex As Integer, State As Integer)**

This method opens/closes the port (as set in the SigPlus.ini) for signature capture.

Arguments:

SigIndex: 1=signature1, 2=signature2 (as displayed in the SigCompare control)

State – 0=close, 1=open

RETURN: None

Example:

```
SigCompare1.SetTabletState(2,1) 'open signature 2 port for capture
```

### **.GetTabletState(SigIndex As Integer) As Integer**

This method returns the current state of the port.

Arguments:

SigIndex: 1=signature1, 2=signature2 (as displayed in the SigCompare control)

RETURN: Integer 0=closed, 1=open

Example:

```
SigCompare1.GetTabletState(2) 'returns the current tablet state of signature 2
```

### **.GetTabletComPort(SigIndex As Integer) As Integer**

This method returns the currently-assigned com port..

Arguments:

SigIndex: 1=signature1, 2=signature2 (as displayed in the SigCompare control)

RETURN: Integer: com port assignment

Example:

```
Dim intComPort As Integer  
strComPort = SigCompare1.GetComPort(2) 'return com port assignment for signature 2
```

### **.SetTabletComPort(SigIndex As Integer, ComPort As Integer)**

This method sets the com port.

Arguments:

SigIndex: 1=signature1, 2=signature2 (as displayed in the SigCompare control)

ComPort: The com port assignment

RETURN: None

Example:

`SigCompare1.SetComPort(1, 1)` 'sets signature 1 com port to COM1

### **.GetTabletType(SigIndex As Integer) As Integer**

This method returns the current tablet connection type (0=serial, 2=USB, 6=HSB)

Arguments:

SigIndex: 1=signature1, 2=signature2 (as displayed in the SigCompare control)

RETURN: Integer: Tablet connection type

Example:

`Dim intTabType As Integer`  
`strTabType = SigCompare1.TabType(1)` 'return tablet connection type for signature 1

### **.SetTabletType(SigIndex As Integer, TabletType As Integer)**

This method sets the tablet connection type

Arguments:

SigIndex: 1=signature1, 2=signature2 (as displayed in the SigCompare control)

TabletType: The tablet type to set (0=serial, 2=USB, 6=HSB)

RETURN: None

Example:

`SigCompare1.SetTabletType(2, 6)` 'sets signature 2 to a Topaz HSB tablet

## **Installation notes**

Topaz Systems SigCompare post-installation directory structure is:

**WIN\**

**\SigPlus\**

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

**\SigCompare\**

[SigCompare.exe](#) (The SigCompare demo using SigCompare.ocx)

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

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

## **SigCompare Dependency List**

VBRuntime files (for SigCompare.exe only)

## **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 it's 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 it Topaz or it's 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 it's 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 it's 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 it's 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 6,307,955 and 5,120,908 and 5,122,623. Patent work is ongoing. 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. 5,120,906; 5,195,133; 5,227,590; 5,297,202; 5,322,978; 5,544,255; 5,647,017; 5,818,955; 6,064,751; 6,091,835; 6,381,344; 6,539,363 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 this document-related 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 SigPlus 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 SigPlus 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 SigPlus software or the SigPlus control in any specific use or implementation. To the extent that SigPlus 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.