kiSHA2.dll (tm)   Kenneth Ives kenaso@tx.rr.com

I am open to ways to improve this application, please email me.

Visual Basic 6.0 with Service Pack 6 runtime files required.
To obtain required files (VBRun60sp6.exe):
http://www.microsoft.com/downloads/details.aspx?FamilyId=7B9BA261-7A9C-43E7-9117-F6730
77FFB3C

VBRun60sp6.exe installs Visual Basic 6.0 SP6 run-time files.
http://support.microsoft.com/kb/290887

This software has been tested on Windows XP through Windows 7.
Windows 9x, 2000 and NT4 are no longer supported.

=
*** WARNING *** WARNING *** WARNING *** WARNING *** WARNING *** WARNING ***
=

You acknowledge that this software is subject to the export control laws and regulations of the United States ("U.S.") and agree to abide by those laws and regulations. Under U.S. law, this software may not be downloaded or otherwise exported, reexported, or transferred to restricted countries, restricted end-users, or for restricted end-uses. The U.S. currently has embargo restrictions against Cuba, Iran, Iraq, Libya, North Korea, Sudan, and Syria. The lists of restricted end-users are maintained on the U.S. Commerce Department's Denied Persons List, the Commerce Department's Entity List, the Commerce Department's List of Unverified Persons, and the U.S. Treasury Department's List of Specially Designated Nationals and Blocked Persons. In addition, this software may not be downloaded or otherwise exported, reexported, or transferred to an end-user engaged in activities related to weapons of mass destruction.

=

REFERENCE:

NIST (National Institute of Standards and Technology)
FIPS (Federal Information Processing Standards Publication)
SP (Special Publications)
http://csrc.nist.gov/publications/PubsFIPS.html

FIPS 180-2 (Federal Information Processing Standards Publication)
dated 1-Aug-2002, with Change Notice 1, dated 25-Feb-2004
http://csrc.nist.gov/publications/fips/fips180-2/FIPS180-2_changenotice.pdf

FIPS 180-3 (Federal Information Processing Standards Publication)
dated Oct-2008 (supercedes FIPS 180-2)
http://csrc.nist.gov/publications/fips/fips180-3/fips180-3_final.pdf

FIPS 180-4 (Federal Information Processing Standards Publication)
dated Mar-2012 (Supercedes FIPS-180-3)
http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf

Examples of the implementation of the secure hash algorithms
SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224 and
SHA-512/256, can be found at:
http://csrc.nist.gov/groups/ST/toolkit/examples.html
http://csrc.nist.gov/groups/ST/toolkit/documents/Examples/SHA2_Additional.pdf

Aaron Gifford's additional test vectors

http://www.adg.us/computers/sha.html

Guidelines for Media Sanitization (SP800-88)
http://csrc.nist.gov/publications/nistpubs/800-88/NISTSP800-88_rev1.pdf

Feb-2005: SHA-1 has been compromised. Recommended that
you do not use for password or document authentication.
http://www.schneier.com/blog/archives/2005/02/sha1_broken.html
http://csrc.nist.gov/groups/ST/toolkit/documents/shs/NISTHashComments-final.pdf

March 15, 2006: The SHA-2 family of hash functions
(i.e., SHA-224, SHA-256, SHA-384 and SHA-512) may be used
by Federal agencies for all applications using secure hash
algorithms. Federal agencies should stop using SHA-1 for
digital signatures, digital time stamping and other
applications that require collision resistance as soon as
practical, and must use the SHA-2 family of hash functions
for these applications after 2010. After 2010, Federal
agencies may use SHA-1 only for the following applications:

- hash-based message authentication codes (HMACs)
- key derivation functions (KDFs)
- random number generators (RNGs)

Regardless of use, NIST encourages application and protocol
designers to use the SHA-2 family of hash functions for all
new applications and protocols.
http://csrc.nist.gov/groups/ST/hash/policy.html

Export Control: Certain cryptographic devices and technical
data regarding them are subject to Federal export controls.
Exports of cryptographic modules implementing this standard
and technical data regarding them must comply with these
Federal regulations and be licensed by the Bureau of Export
Administration of the U.S. Department of Commerce.
Information about export regulations is available at:
http://www.bis.doc.gov/index.htm

SHA-2 support on MS Windows
Paraphrasing: Regarding SHA-224 support, SHA-224 offers less security
than SHA-256 but takes the same amount of resources. Also SHA-224 is
not generally used by protocols and applications. The NSA's (National
Security Agency) Suite B standards also does not include it. Microsoft
has no plans to add it to future versions of their Cryptographic
Service Providers (CSP).

http://blogs.msdn.com/b/alejacma/archive/2009/01/23/sha-2-support-on-windows-xp.aspx

****

How to use:

For a simple example, execute the SHA_Demo application. The demo converts
the data to a byte array prior to passing it to the DLL to be hashed.

[STRING DATA]
Convert string data to byte array prior to passing to the HashString function.

Example: abytData() = StrConv("abc", vbFromUnicode)

[FILE DATA]
Just the path and filename are passed in the byte array. Convert the
path\filename data to byte array prior to passing to the HashFile function.
The HashFile routine will open and read the file into an internal byte array.

Example:
```
  abytData() = StrConv("C:\Files\Test Folder\Testfile.doc", vbFromUnicode)
```

Both will create a hashed output string based on file data input.

****

Secure Hash Algorithm

DESCRIPTION
The Secure Hash Algorithm (SHA) is required for use with the Digital Signature Algorithm (DSA) as specified in the Digital Signature Standard (DSS) and whenever a secure hash algorithm is required for federal applications. For a message of length < 2^64 bits, this algorithm produces a condensed representation of the message called a message digest. The message digest is used during generation of a signature for the message. This also used to compute a message digest for the received version of the message during the process of verifying the signature. Any change to the message in transit will, with very high probability, result in a different message digest, and the signature will fail to verify.

These algorithms have been tested to be accurate in accordance with FIPS-180-2 publication.

According to FIPS 180-3 there are only two differences between SHA-224 and SHA-256.

1. The initializing values are different
2. Just the left most 224 bits (28 bytes) are saved

According to FIPS 180-3 there are only two differences between SHA-384 and SHA-512.

1. The initializing values are different
2. SHA-384 only uses the first six elements for the output.
   SHA-512 uses all eight elements for the output.

```
==============================================================================
Located in Hash (clsHash.cls)
==============================================================================

' ****************************************************************************
' Enumerations
' ****************************************************************************
  Public Enum enumHASH_ALGORITHM
      eHASH_SHA1        ' 0
      eHASH_SHA224      ' 1
      eHASH_SHA256      ' 2  Default
      eHASH_SHA384      ' 3
      eHASH_SHA512      ' 4
      eHASH_SHA512_224  ' 5  As per FIPS 180-4 (dtd March-2012)
      eHASH_SHA512_256  ' 6  As per FIPS 180-4 (dtd March-2012)
      eHASH_SHA512_320  ' 7  My creation as per FIPS 180-4 (dtd March-2012)
  End Enum

' ****************************************************************************
' ****                        Properties                                 ****
' ****************************************************************************
Version - Output - String - Name of DLL and version information

StopProcessing - Input/Output - Boolean - True if user wants to stop
     processing.

HashMethod - Input only - [OPTIONAL] Long integer (0-6) designating what
     hash algorithim to use. See enumHASH_ALGORITHM

HashRounds - Input only - [OPTIONAL] Long integer (1-10) designating number
     of hash iterations to use.

ReturnLowercase - Input only - [OPTIONAL] Boolean - Return hashed data in
     upper or lower case format.  Default = TRUE

' ****************************************************************************
' ****                        Methods                                    ****
' ****************************************************************************
' Creates a hash output string based on string data input
' in byte array format.
Function HashString(ByRef abytInput() As Byte) As Variant

' Creates a hash output string based on file data input.
' Input is the Path/File location only in byte array format.
Function HashFile(ByRef abytInput() As Byte) As Variant
```

```
================================================================================
                              Overview
================================================================================

Message padding

All data is stored in Big_Endian format with the Most Significant Bit (MSB)
first.

The message 'M' shall be padded before hash computation begins. The purpose
of this padding is to ensure that the padded message is a multiple of 512 or
1024 bits, depending on the algorithm.

32-Bit Format (160, 224, 256)
   Suppose the length of the message 'M', in bits, is 'l' bits. Append the
   bit '1' to the end of the message, followed by 'k' zero bits, where 'k'
   is the smallest non-negative solution to the equation l+1+k=448 mod 512.
   Then append the 64-bit block that is equal to the number 'l' expressed
   using a binary representation. The length of the padded message should
   now be a multiple of 512 bits.

64-Bit Format (384, 512)
   Suppose the length of the message 'M', in bits, is 'l' bits. Append the
   bit '1' to the end of the message, followed by 'k' zero bits, where 'k'
   is the smallest non-negative solution to the equation l+1+k=896 mod 1024.
   Then append the 128-bit block that is equal to the number 'l' expressed
   using a binary representation. The length of the padded message should
   now be a multiple of 1024 bits.


Constant and work arrays

The definition of the initial hash value H(0) allows its implementation to
be shared between the algorithms.  That is, the left halves of the SHA-512
words of H(0) are the words of H(0) for SHA-256. Similarly for SHA-384, the
right halves of the words of H(0) are the words for H(0) for SHA-224.  For
example for H(0):

    SHA-224 H(0) =         c1059ed8       Right half of SHA-384
    SHA-384 H(0) = cbbb9d5dc1059ed8

    SHA-256 H(0) = 6a09e667               Left half of SHA-512
    SHA-512 H(0) = 6a09e667f3bbc908

A similar implementation holds true for the constants arrays.  This uses
the left half of SHA-512.

    SHA-224, SHA-256 = 428a2f98           Left half of SHA-512
             SHA-512 = 428a2f98d728ae22


Calculating the Square and Cube Roots

Got this explanation from Ask Dr. Math web site.
http://mathforum.org/dr.math/

There is a simple process for converting a base 10 decimal fraction to a
"decimal" fraction in another base.  We repeatedly multiply (in base 10) our
decimal fraction by the new base, picking off the whole number part each time
as the next digit of the final output.

Example:   1st Prime number = 2
           Cube root of 2 = 1.2599210498948731647672106073
```

```
0.259921049894873164767210607 * 16 =  4.15873679831797063627536971 68
```

```
0.2599210498948731647672106073 * 16 =  4.1587367983179706362753697168
0.1587367983179706362753697168 * 16 =  2.5397887730875301804059154688
0.5397887730875301804059154688 * 16 =  8.6366203694004828864946475 01
0.636620369400482886494647501  * 16 = 10.185925910407726183914360016
0.185925910407726183914360016  * 16 =  2.974814566523618942629760256
0.974814566523618942629760256  * 16 = 15.597033064377903082076164096
0.597033064377903082076164096  * 16 =  9.552529030046449313218625536
0.552529030046449313218625536  * 16 =  8.840464480743189011498008576
0.840464480743189011498008576  * 16 = 13.447431691891024183968137216
0.447431691891024183968137216  * 16 =  7.158907070256386943490195456
0.158907070256386943490195456  * 16 =  2.542513124102191095843127296
0.542513124102191095843127296  * 16 =  8.680209985635057533490036736
0.680209985635057533490036736  * 16 = 10.883359770160920535840587776
0.883359770160920535840587776  * 16 = 14.133756322574728573449404416
0.133756322574728573449404416  * 16 =  2.140101161195657175190470656
0.140101161195657175190470656  * 16 =  2.241618579130514803047530496
```

The hex representation of the fractional part of the
CUBE ROOTS of 2 is: 428a2f98d728ae22

Example: 1st Prime number = 2
Square root of 2 = 1.4142135623730950488016887242

```
0.4142135623730950488016887242 * 16 =  6.6274169979695207808270195872
0.6274169979695207808270195872 * 16 = 10.0386719675123324932323133395
0.038671967512332493232313395  * 16 =  0.618751480197319891717014 32
0.61875148019731989171701432   * 16 =  9.90002368315711826747222912
0.90002368315711826747222912   * 16 = 14.40037893051389227955566592
0.40037893051389227955566592   * 16 =  6.40606288822227647289065472
0.40606288822227647289065472   * 16 =  6.49700621155642356625047552
0.49700621155642356625047552   * 16 =  7.95209938490277706000760832
0.95209938490277706000760832   * 16 = 15.23359015844443296012173312
0.23359015844443296012173312   * 16 =  3.73744253511092736194772992
0.73744253511092736194772992   * 16 = 11.79908056177483779116367872
0.79908056177483779116367872   * 16 = 12.78528898839740465861885952
0.78528898839740465861885952   * 16 = 12.56462381435847453790175232
0.56462381435847453790175232   * 16 =  9.03398102973559260642803712
0.03398102973559260642803712   * 16 =  0.54369647576948170284859392
0.54369647576948170284859392   * 16 =  8.69914361231170724557750272
```

The hex representation of the fractional part of the
SQUARE ROOTS of 2 is: 6a09e667f3bcc908

SHA-512 Work array:

These words were obtained by taking the first sixty-four bits of the fractional parts of the SQUARE ROOTS of the first 8 prime numbers.

SHA-384 Work array:

These words were obtained by taking the first sixty-four bits of the fractional parts of the SQUARE ROOTS of the 9th through 16th prime numbers.

Data for constants array:

SHA-384 and SHA-512 use the first sixty-four bits of the fractional part of the CUBE ROOTS of the first 80 prime numbers.

Test data was obtained from the following:

FIPS 180-2 (Federal Information Processing Standards Publication)
dated 1-Aug-2002, with Change Notice 1, dated 25-Feb-2004
http://csrc.nist.gov/publications/fips/fips180-2/FIPS180-2_changenotice.pdf

FIPS 180-3 (Federal Information Processing Standards
Publication) dated Oct-2008 (supercedes FIPS 180-2)
http://csrc.nist.gov/publications/fips/fips180-3/fips180-3_final.pdf

FIPS 180-4 (Federal Information Processing Standards Publication)
dated Mar-2012 (Supercedes FIPS-180-3)
http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf

Aaron Gifford's additional test vectors
http://www.adg.us/computers/sha.html

Examples of the implementation of the secure hash algorithms
SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224 and
SHA-512/256, can be found at:
http://www.nist.gov/CryptoToolkitExamples

All other test data is of my choosing.

===
License Kenneth Ives
kenaso@tx.rr.com
===

Preamble

This License governs Your use of the Work. This License is intended to allow developers to use the Source Code and Executable Files provided as part of the Work in any application in any form.

The main points subject to the terms of the License are:

  Source Code and Executable Files can be used in commercial applications;
  Source Code and Executable Files can be redistributed; and
  Source Code can be modified to create derivative works.

No claim of suitability, guarantee, or any warranty whatsoever is provided. The software is provided "as-is".

This License is entered between You, the individual or other entity reading or otherwise making use of the Work licensed pursuant to this License and the individual or other entity which offers the Work under the terms of this License ("Author").

License

THE WORK (AS DEFINED BELOW) IS PROVIDED UNDER THE TERMS OF THIS CODE PROJECT OPEN LICENSE ("LICENSE"). THE WORK IS PROTECTED BY COPYRIGHT AND/OR OTHER APPLICABLE LAW. ANY USE OF THE WORK OTHER THAN AS AUTHORIZED UNDER THIS LICENSE OR COPYRIGHT LAW IS PROHIBITED.

BY EXERCISING ANY RIGHTS TO THE WORK PROVIDED HEREIN, YOU ACCEPT AND AGREE TO BE BOUND BY THE TERMS OF THIS LICENSE. THE AUTHOR GRANTS YOU THE RIGHTS CONTAINED HEREIN IN CONSIDERATION OF YOUR ACCEPTANCE OF SUCH TERMS AND CONDITIONS. IF YOU DO NOT AGREE TO ACCEPT AND BE BOUND BY THE TERMS OF THIS LICENSE, YOU CANNOT MAKE ANY USE OF THE WORK.

Definitions.

"Articles" means, collectively, all articles written by Author which describes how the Source Code and Executable Files for the Work may be used by a user.

"Author" means the individual or entity that offers the Work under the terms of this License.

"Derivative Work" means a work based upon the Work or upon the Work and other pre-existing works.

"Executable Files" refer to the executables, binary files, configuration and any required data files included in the Work.

"Publisher" means the provider of the website, magazine, CD-ROM, DVD or other medium from or by which the Work is obtained by You.

"Source Code" refers to the collection of source code and configuration files used to create the Executable Files.

"Standard Version" refers to such a Work if it has not been modified, or has been modified in accordance with the consent of the Author, such consent being in the full discretion of the Author.

"Work" refers to the collection of files distributed by the Publisher, including the Source Code, Executable Files, binaries, data files, documentation, whitepapers and the Articles.

"You" is you, an individual or entity wishing to use the Work and exercise your rights under this License.

Fair Use/Fair Use Rights. Nothing in this License is intended to reduce, limit, or restrict any rights arising from fair use, fair dealing, first sale or other limitations on the exclusive rights of the copyright owner under copyright law or other applicable laws.

License Grant. Subject to the terms and conditions of this License, the Author hereby grants You a worldwide, royalty-free, non-exclusive, perpetual (for the duration of the applicable copyright) license to exercise the rights in the Work as stated below:

You may use the standard version of the Source Code or Executable Files in Your own applications.

You may apply bug fixes, portability fixes and other modifications obtained from the Public Domain or from the Author. A Work modified in such a way shall still be considered the standard version and will be subject to this License.

You may otherwise modify Your copy of this Work (excluding the Articles) in any way to create a Derivative Work, provided that You insert a prominent notice in each changed file stating how, when and where You changed that file.

You may distribute the standard version of the Executable Files and Source Code or Derivative Work in aggregate with other (possibly commercial) programs as part of a larger (possibly commercial) software distribution.

The Articles discussing the Work published in any form by the author may not be distributed or republished without the Author's consent. The author retains copyright to any such Articles. You may use the Executable Files and Source Code pursuant to this License but you may not repost or republish or otherwise distribute or make available the Articles, without the prior written consent of the Author.

Any subroutines or modules supplied by You and linked into the Source Code or Executable Files this Work shall not be considered part of this Work and will not be subject to the terms of this License.

Patent License. Subject to the terms and conditions of this License, each Author hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, import, and otherwise transfer the Work.

Restrictions. The license granted in Section 3 above is expressly made subject to and limited by the following restrictions:

You agree not to remove any of the original copyright, patent, trademark, and attribution notices and associated disclaimers that may appear in the Source Code or Executable Files.

You agree not to advertise or in any way imply that this Work is a product of Your own.

The name of the Author may not be used to endorse or promote products derived from the Work without the prior written consent of the Author.

You agree not to sell, lease, or rent any part of the Work.

You may distribute the Executable Files and Source Code only under the terms of this License, and You must include a copy of, or the Uniform Resource Identifier for, this License with every copy of the Executable Files or Source Code You distribute and ensure that anyone receiving such Executable Files and Source Code agrees that the terms of this License apply to such Executable Files and/or Source Code. You may not offer or impose any terms on the Work that alter or restrict the terms of this License or the recipients' exercise of the rights granted hereunder. You may not sublicense the Work. You must keep intact all notices that refer to this License and to the disclaimer of warranties. You may not distribute the Executable Files or Source Code with any technological measures that control access or use of the Work in a manner inconsistent with the terms of this License.

You agree not to use the Work for illegal, immoral or improper purposes, or on pages containing illegal, immoral or improper material. The Work is subject to applicable export laws. You agree to comply with all such laws and regulations that may apply to the Work after Your receipt of the Work.

Representations, Warranties and Disclaimer. THIS WORK IS PROVIDED "AS IS", "WHERE IS" AND "AS AVAILABLE", WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES OR CONDITIONS OR GUARANTEES. YOU, THE USER, ASSUME ALL RISK IN ITS USE, INCLUDING COPYRIGHT INFRINGEMENT, PATENT INFRINGEMENT, SUITABILITY, ETC. AUTHOR EXPRESSLY DISCLAIMS ALL EXPRESS, IMPLIED OR STATUTORY WARRANTIES OR CONDITIONS, INCLUDING WITHOUT LIMITATION, WARRANTIES OR CONDITIONS OF MERCHANTABILITY, MERCHANTABLE QUALITY OR FITNESS FOR A PARTICULAR PURPOSE, OR ANY WARRANTY OF TITLE OR NON-INFRINGEMENT, OR THAT THE WORK (OR ANY PORTION THEREOF) IS CORRECT, USEFUL, BUG-FREE OR FREE OF VIRUSES. YOU MUST PASS THIS DISCLAIMER ON WHENEVER YOU DISTRIBUTE THE WORK OR DERIVATIVE WORKS.

Indemnity. You agree to defend, indemnify and hold harmless the Author and the Publisher from and against any claims, suits, losses, damages, liabilities, costs, and expenses (including reasonable legal or attorneys’ fees) resulting from or relating to any use of the Work by You.

Limitation on Liability. EXCEPT TO THE EXTENT REQUIRED BY APPLICABLE LAW, IN NO EVENT WILL THE AUTHOR OR THE PUBLISHER BE LIABLE TO YOU ON ANY LEGAL THEORY FOR ANY SPECIAL, INCIDENTAL, CONSEQUENTIAL, PUNITIVE OR EXEMPLARY DAMAGES ARISING OUT OF THIS LICENSE OR THE USE OF THE WORK OR OTHERWISE, EVEN IF THE AUTHOR OR THE PUBLISHER HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Kenneth Ives kenaso@tx.rr.com