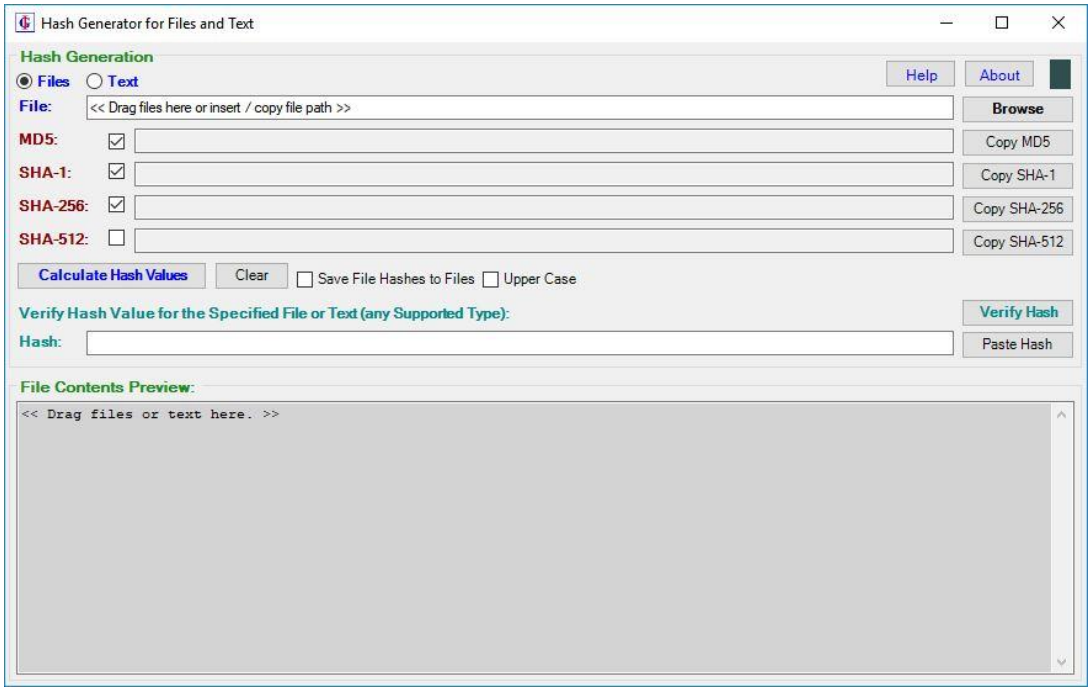
HashForm

Handy Tool for Calculation and Verification of Checksums of Files and Text

User Manual

Version 1.6, May 2017.

by Igor Grešovnik



This software is based on the *Investigative Generic Library* (*IGLib*).

Contents:

1 INTRODUCTION

***HashForm* is a handy GUI application for quickly calculating or verifying various types of checksums (hashes) of arbitrary files or provided text**.

Checksums are commonly used in data integrity checks, i.e., to verify that file content or any other data in digital form has not been modified with respect to original or expected content. Whether you store or transmit (e.g. send by mail or download from the internet) the data, by calculating the checksums of that the data at two point in time and comparing these checksums, you can confirm that the data has not been changed between these two points if the checksums match.

***HashForm* has the following features**:

- Calculation and verification of standard checksums (hash values):
  - MD5
  - SHA-1
  - SHA-256
  - SHA-512
  - User can select which checksums are calculated (important for large files)
  - Possibility of saving calculated checksums to a file
- Calculation of checksums (hash values) of arbitrary files
  - Specify the file by browsing or drag & drop a file from a file manager or by specifying file path
  - Directory of last selected file is remembered
  - Immediate calculation of checksums after selection of a file
- Calculation of checksum of arbitrary text
  - Text can be typed in by user
  - Text can also be dragged & dropped from a text editor, office application, browser, or any other application that supports selection and drag & drop of text
    - Immediate hash calculation when text is dragged into the working area
- Easy switching between calculation of checksums for files or for text
  - Switching is automatic:
    - when text or file is dragged & dropped into the working area
    - when a file is selected either by browsing or by specifying file path
- Easy to use user interface, optimized to speed up work and reduce possibility of mistakes
  - Help file – user manual can be displayed by clicking a button
  - Responsive user interface in case of large files
  - Possibility to cancel calculation of checksums (important for very large files)
  - Automatic calculation of checksums when possible, no need to click a button:

        - After drag & drop of file or text
        - After selection of a new file
    - Copying of calculated checksums by clicking a button to minimize possibility of errors
    - Pasting checksums to be verified from clipboard by clicking a button
    - Indicator lights indicates status: no results provided, results calculated, busy, error occurred.
- Cross-platform, runs on Windows, Lunux or Mac. On non-Windows systems, the Mono framework must be installed in order to run HashForm. The same executable is run on different platforms.

2 USER MANUAL

2.1 Installation and Running

HashForm is a portable application, therefore installation or setup is not needed to use it. You can run it from a disk or even from a USB stick or any other internal or external storage device. You simply download the executable file (the .exe extension) and run it. The only requirement is that you have .NET framework (on Windows) or Mono Framework (on Windows or Linux) installed. Most modern Windows computer have .NET Framework installed by default, so you don't need to install anything in order to run the application.

Instead of packed single executable, you can also download a compressed directory containing executable and some dynamic libraries (in a single file executable, the libraries and executable are bundled in one file). This variant also contains help file (the manual), which can be viewed when offline. With single file executable, the help files can be viewed only when online, unless you download the files (either HTML or PDF or both) to the same directory that contains the executable.

When executable is downloaded, you can double click it from any file manager such as Windows Explorer, Total Commander, Krusader (on Linux) or whatever tool you use. You can also run it from command prompt - in this case it is convenient to put the executable (and the dll files and help files, if you download the multi-file variant) into a directory included in *system PATH* variable.

2.2 Use

Use of software is very intuitive and you will probably not need much instructions in order to discover most of its features (Figure 1). In order to calculate various checksums (hashes) of a file, wither drag&drop the file (e.g. from a file manager application) into the larger text box below, or browse to the file using the *Browse button*. All activated types of checksums are calculated

immediately. Use checkboxes on the left in order to switch calculation of specific types of checksums.

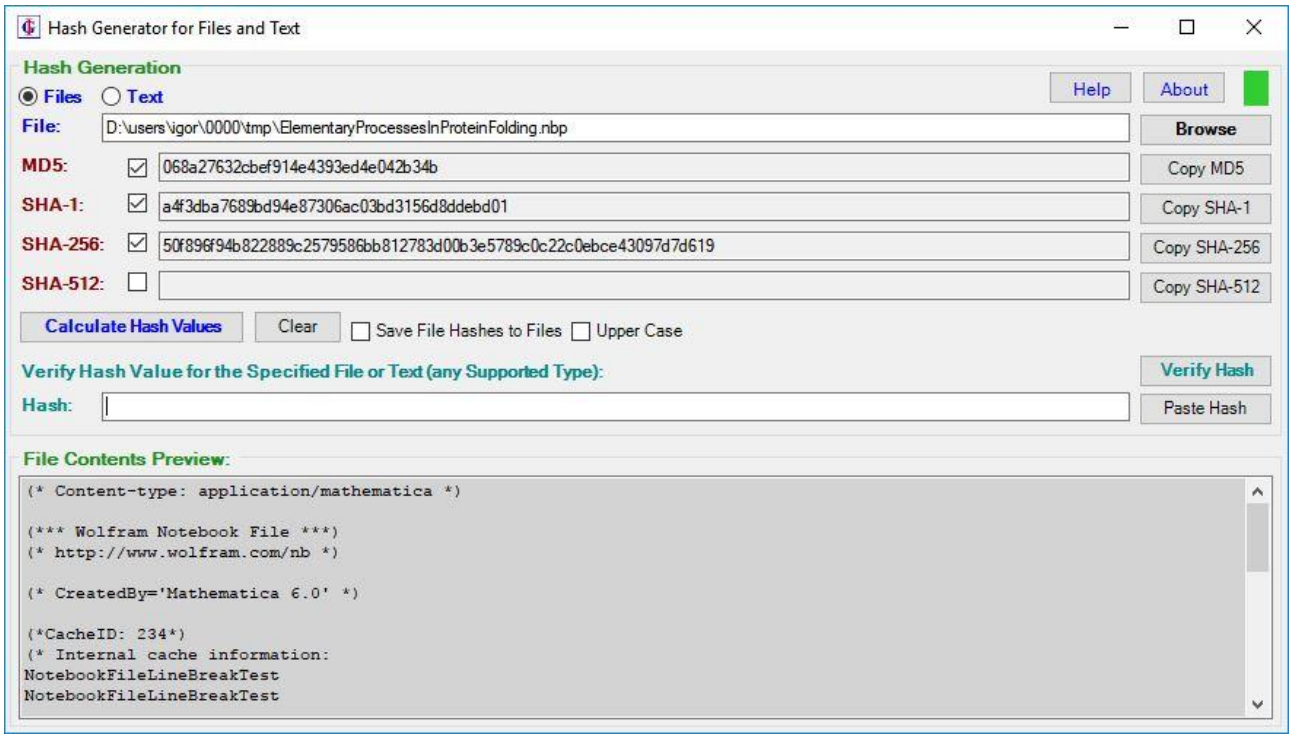


Figure 1: HashForm user interface, with hashes for a selected file calculated.

To calculate checksums of arbitrary text, click the *Text radio button*, inseert text into the box at the bottom, and click the *"Calculate Hash Values" button*. You can also drag selected text from a text editor, office application, web browser, or any other application that allows celection and drag&drop of text, into the box at the bottom.

When a file or text is dragged into the box at the bottom, the file / text mode is switched automatically as necessary (the current mode is established from whether the *Files* or *Text* radio buttn is checked).

Checksums (hash values) can be auomatically saved to a file by checking the *"Save File Hashes to Files"*.

At any time, checksums can be re-calculated by clicking the "*Calculate Hash Values*" button. Most of the time, this is not necessary for files because calculation starts automatically as soon as the file is selected - either by dragind a file into the box at the bottom, browsing to a file by clicking the *"Browse" button*, or inserting file path into the text box labeled *"File:"* and pressing Enter.

Calculated checksums (hashes) are displayed in standard hexadecimal form. By checking the *"Upper Case" checkbox*, upper case 'A' to 'F' letters are used to represent digits 10 to 15 (switching upper/lower case may be useful if you need to compare with values written on paper). Calculation of any type of checksum is switched on or off by the corresponding checkbox next to its type name (*"MD5"*, *"SHA-1"*, etc.).

At any time you can verify whether actual file or text hash corresponds to the specific value provided elsewhere. Just paste the value provided by your source into the text box beside *"Hash:"* and click the *"Verify Hash" button* above the text box. If the provided hexadecimal value correspond to actual hash of your text or file of any given type that is considered by the application, an informative dialog box is launched stating which type of actual hash was provided. It the provided value does not correspond to any considered type of hash of the current file or text, then an error dialog box is launched informing user of the situation.

Please note that provided value is checked only for those types of checksums listed in the user interface. If a valid checksum of any other type is provided, it is not be recognized. However, the check is not performed only against active types (whose checkboxes are checked) but against all types.

More detailed specification of the user interface is found below.

2.3 Detailed Functionality

2.3.1 Calculating File Checksums

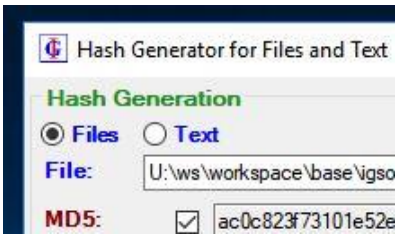


Figure 2: The "Files" radio button is on when file hashes are calculated or verified.

When File checksums (hashes) are calculated, the "Files" radio button is switched on (Figure 2). The file whose hashes are calculated can be specified in several ways:

- A file is dragged and dropped into the textbox denoted "File", or alternatively into the larger textbox at the bottom of HashForm's window. File can be dragged from a file explorer such as Windows Explorer, Total Commander (Windows), Midnight Commander, Dolphin, Krusader, Konqueror, GNOME Commander (Linux), etc.
- File path is inserted or copied into the above mentioned text box.
- By clicking the "Browse" button, file can be specified by the file selection dialog.

Hashes are calculated immediately after a new file is specified in one of the above mentioned ways. Only those hashes that are checked are calculated. Hashes can always be re-calculated by clicking the **"Calculate Hash Values" button**, or cleared by clicking the **"Clear" button**.

By checking additional **checkboxes for different types of hashes**, hash values are immediately calculated for these values. If any of these checkboxes is unchecked then the hash value remains visible until a new file is selected or re-calculation is performed or the "Clear" button

is clicked. Each of the supported hash type is denoted by an appropriate label followed by a checkbox where calculation of this type of hash values is switched on or off. A text box follows where hash value is written after it is calculated.

Each text box is followed by a "Copy …" button. Clicking this button **copies** the corresponding **hash value to clipboard**. After clicking the button, hash value can be pasted from clipboard into every part that supports pasting text. The appropriate copy button can also be clicked for those types of file hashes that were not calculated (i.e., whose checkboxes are unchecked). In this case, the hash value is first calculated, then written to the appropriate textbox and finally copied to clipboard. The corresponding checkbox remains uncheck, and when another file is specified, the corresponding hash is not calculated.

Calculated hash values are represented in hexadecimal form where letters 'a' thru 'f' are used to represent hexadecimal digits from 10 to 15. By convention, both **lower and upper case letters** are legal, and user can switch between one and another by checking or unchecking the checkbox labeled "Upper Case". When hash values are verified, using lower or upper case letters does not affect the result. Also, in the pasted hash value to be verified, spaces before or after the string are ignored and don't affect the result.

2.3.1.1 Verifying File Checksums

The most important functionality provided by the software is the ability to **verify whether file hash**, which is calculated by the software, **corresponds to expected file hash** provided by some other source.

For example, in a typical scenario you may download a file (such as software installer or compressed archive) from the internet and the author (creator, originator, or source) of that file provides one or more hash values of different types for that file. When you download a file form the interned, it may not actually be the file you believe it is – somebody may have uploaded a different file on the server (which is often not controlled by the original creator of the file) or even infect the original file by a virus or some other malicious software. By calculating the actual hash value of the downloaded file and comparing it to the hash value provided by the original author of the file, you can check whether you have actually downloaded the unmodified original files that you expected to download.

Verifying files in this way is primary intention of calculated hash values. Hash value is calculated from the complete contents of the file by applying of the *hash function* to file contents. Hash functions have special properties that ensure that if two files have the same calculated hash value then their contents are for sure the same. More precisely, it is extremely unlikely that two different files would produce the same hash value. Hash functions have the property that any small change in the file (even if a single bit of a long file is changed) results in completely different hash value that is uncorrelated with hash value of the original file. It is possible (but very unlikely) that two files that differ in contents would have the same hash value. For the specified hash value, it is also extremely difficult to construct a file that would produce that hash value. Because of these properties, if the checked file has the same hash as the original one, we can assume with great certainty that the check file is the same as the original one.

Some algorithms for calculating hash values are less safe, meaning that it is less difficult to generate a file that has exactly the same hash as the specified (provided) hash. The MD5 and SHA-1 algorithms are known to have this kind of vulnerability. Therefore, it is advised to use better (and slower) algorithms such as SHA-256 or SHA-1. HashForm can calculate MD5, SHA-1, SHA-256

and SHA-512 hashes. These are most common and standard types used in practice and cover vast majority of cases where file hashes are needed.

It would be cumbersome and error prone to manually (visually) compare the calculated hash of the file to the hash provided by the original source of the file. For this reason, HashForm provides a comfortable tool to check whether a provided hash value (checksum) is the same as a given checksum calculated for the specified file.

In order to **verify file hash**, simply copy and paste the provided hash into a text box under "Verify Hash Value…". This can be done by clicking in the text box (beside the "Hash:" label) and pressing *Ctrl-V* on the keyboard, or by right-clicking in the text box and selecting "Paste" from the context menu. The same is achieved by simply clicking the "Paste Hash" button, which copies contents of the clipboard into the textbox.

After the reference (provided) hash value is pasted into the text box, you can verify whether this hash is the same as the actual (calculated) file hash by clicking the "Verify Hash" button. This compares the provided hash (pasted into the text box) with each of the calculated file hashes and checks whether the provided hash matches any of the calculated hashes. If it does then a message box is launched telling which type of the calculated hash value corresponds to the provided hash value (Figure 10).

If no match is found then an error dialog is shown informing you that the provided hash value does not correspond to any of the calculated hash values (Figure 4).

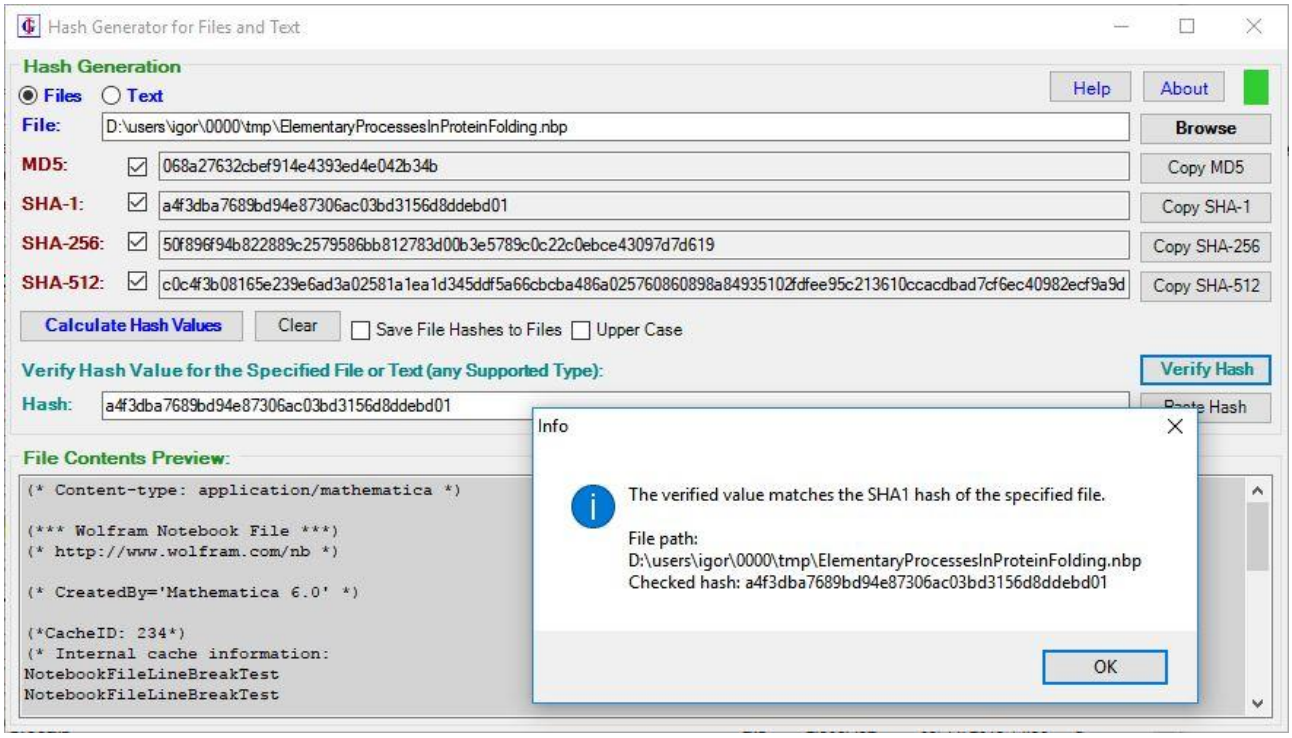


Figure 3: Successful verification of file hash. Message box informs the user that the provided hash value copied into textbox beside the "Hash:" label corresponds to the calculated SHA-1 hash of the specified file that is being verified. This means that the verified file is almost certainly the same as the original file for which the hash was provided (typically by the original creator or source of the file).

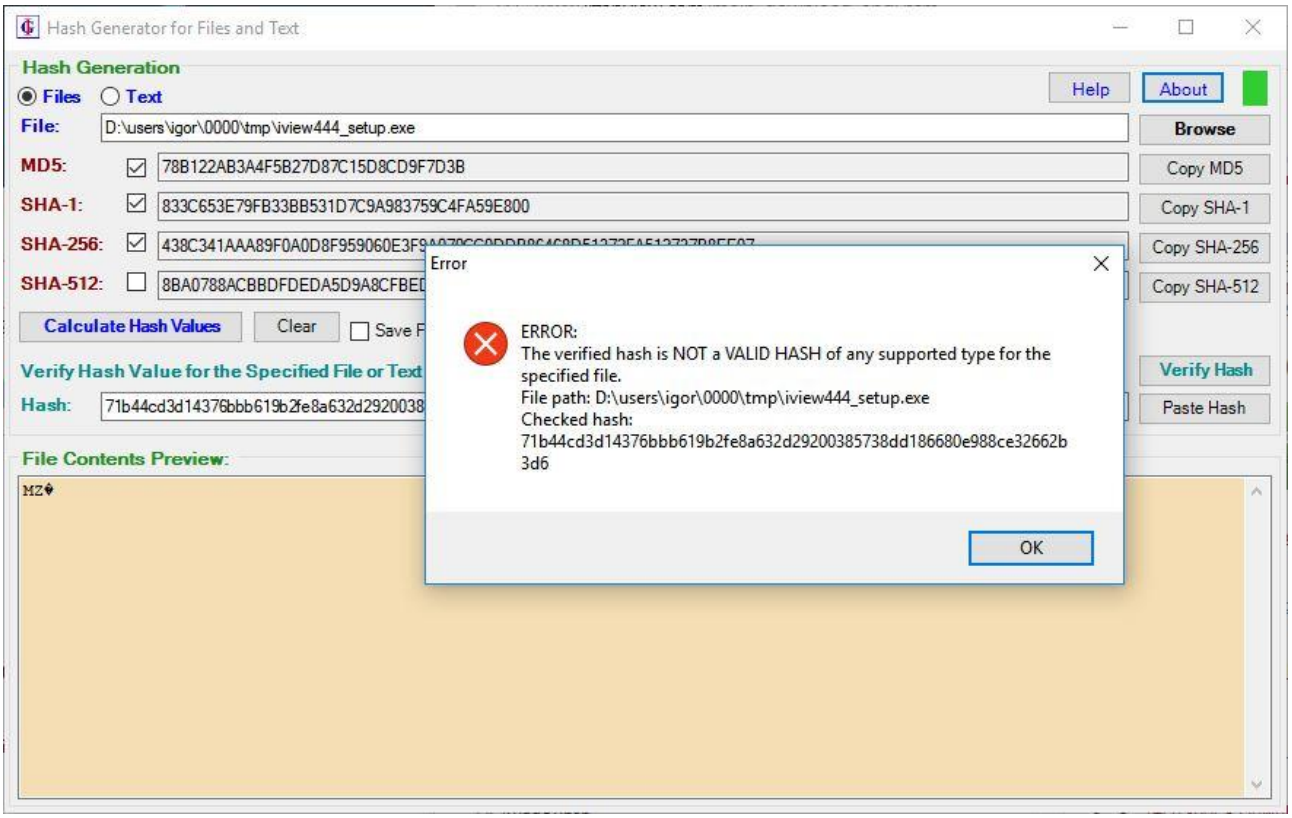


Figure 4: Unsuccessful verification of file hash. The provided hash value does not correspond to any of the hash values calculated for the specified file that the user wishes to verify. This means that the verified file is not the same as the original file for which hashes were provided. Maybe the file was corrupted during transfer or storage, or somebody intentionally modified the original file, possibly to attach malware or unwanted programs to the file.

2.3.1.2 Saving File Checksums to a File

After file hash values are calculated, they can be saved to a file by checking the ”Save File Hashes to Files” checkbox. Hashes are saved to the file with the same name as the verified file, but with the “.chk” extension added. Hashes are saved to a text file whose contents can always be retrieved later. File contents look like this:

```
File:   iview444_setup.exe.chk
Length: 397

Hash values:

  MD5:
D569B2D84E1254DB923070C426327E57
  SHA1:
57EE0A7364FD2E089B69869F9AF8E64E34C3A204
  SHA256:
```

```
54459367952567DB5AB5624F88B4BD373123562DC5B4D2DC5CA9A670518FA9EE
```

Figure 5: Contents of the file with extension ".chk" where calculated hash values for a given file are stored.

2.3.1.3 Calculation of Checksums for Large Files

Calculation of checksums for small files (up to several megabytes) is almost instantaneous on modern personal computers. The situation is different with large files. For example, on my computer with the i7-6700HQ processor, calculation of a SHA-512 checksum takes about 4 seconds for a file of size 200 MB. Times are proportionally larger for larger files, and today it can easily happen that a user has to deal with files whose sizes are several gigabytes.

HashForm has several features that make working with large files easier. There is an indicator light in the top-right corner (Figure 6), which indicates status of the application. Only when all required checksums of a file are successfully calculated, the color of the indicator light turns green. Another color indicates when the application is busy with calculation. If for whatever reason all required checksums could not be calculated then the indicator light turns red and remains read until another calculation is successfully carried out.

When calculating checksums of large files, the "**Cancel Calculation**" button appears. If user clicks this button, any calculation that is currently performed is immediately aborted, and the indicator light indicates that the required checksums were not calculated (Figure 6).

Figure 6: Calculation of checksums of large files: indicator light indicates that the application is busy with calculation, and the "Cancel Calculation" button appears.

2.3.2 Calculating File Checksums of Arbitrary Text

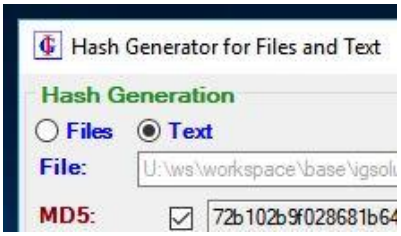


Figure 7: The "Text" radio button is on when text hashes are calculated or verified.

When text checksums (hashes) are calculated, the "Text" radio button is switched on (Figure 7). When clicking the "Calculated Hash Values" button, checksums are calculated for text that is entered in the textbox at the bottom (labeled "Text whose hash value is calculated").

Calculation and verification of checksums for text is done in a similar manner as in the case of files. After application switches to text mode (e.g. by clicking the "Text" radio button, see Figure 8), user can type the text in the textbox at the bottom and calculate hashes by clicking the "Calculated Hash Values" button. Hashes are always calculated for all text that is currently in the text box. After typing additional text, calculated hashes are cleared and the "Calculate Hash Values" button must be clicked again in order to re-calculate hash values for the new text and display them in the appropriate text boxes.

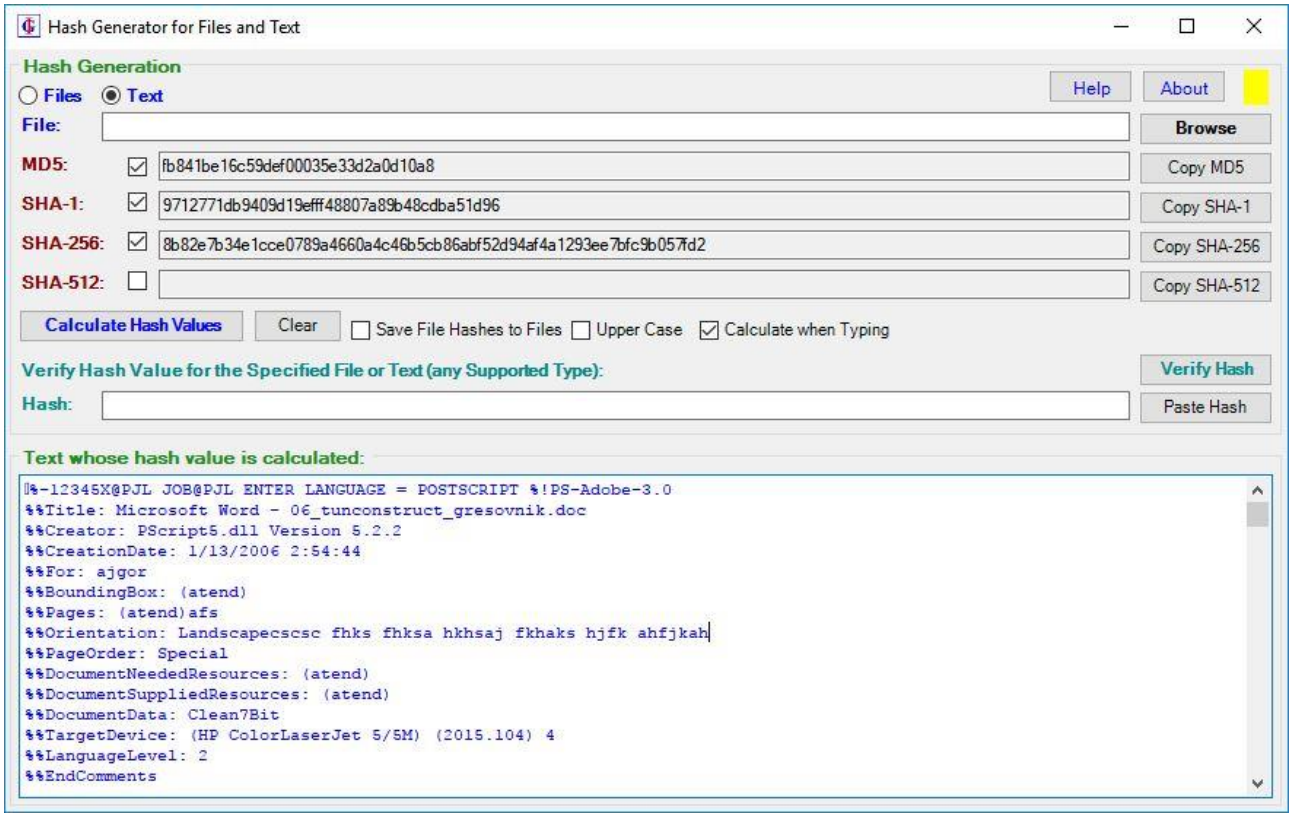


Figure 8: Calculation of hash values of the text in the text box. Since the "Calculate when Typing" is checked, calculation is performed immediately when the user types in the text box and thus the text changes. Yellow indicator light (top right corner) indicates that calculation is going on for the new text and so the hash values shown are not consistent with the current state of the text including the very last characters typed in. If typing is paused for a period long enough that hashes can be calculated for the last state of the text, calculation will catch up and the indicator light will turn green.

If "**Calculate when Typing**" is checked, hash values are calculated immediately as the user types in the text box. This also works where the text box contains very large text, such that

calculation of a hash value takes much longer than the length of time interval between two key strokes. In this case, hash values are not calculated for all intermediate values of inserted text, such that calculation can catch up. This means that hash values displayed do not necessarily correspond to the current text in the text box as they may correspond to the text as it was a few key strokes ago. If the user stops typing for a while, hashes will be calculated for the last state of text in the text box. Indicator light at the top right corner indicates when hashes are being calculated and when they don't correspond to the latest state of text in the text box. Only when calculation catches up with the user typing in the text, the indicator light turns light green.

Text can be **pasted into the text box** from clipboard, therefore it can be copy-paste from any application that supports selection and copying of text (such as browsers, office applications, word processors, text editors, etc.). In order to copy-paste the text whose hash values are calculated, the application must be in the text mode. If in the file mode, user must first click the "Text" radio button to switch to text mode.

Beside that, text can also be dragged from other applications and dropped into text box. In this case, contents of the text box is replaced by the dragged and dropped text (i.e., dropped text is not just pasted between the existent text but it completely replaces the eventual text already contained in the text box – use copy / paste in order to achieve that behavior). If the application is in file mode, it automatically switches to text mode when text is dragged and dropped into the text box. Figure 9 shows a situation when text is dragged and dropped into the text box from a web browser.

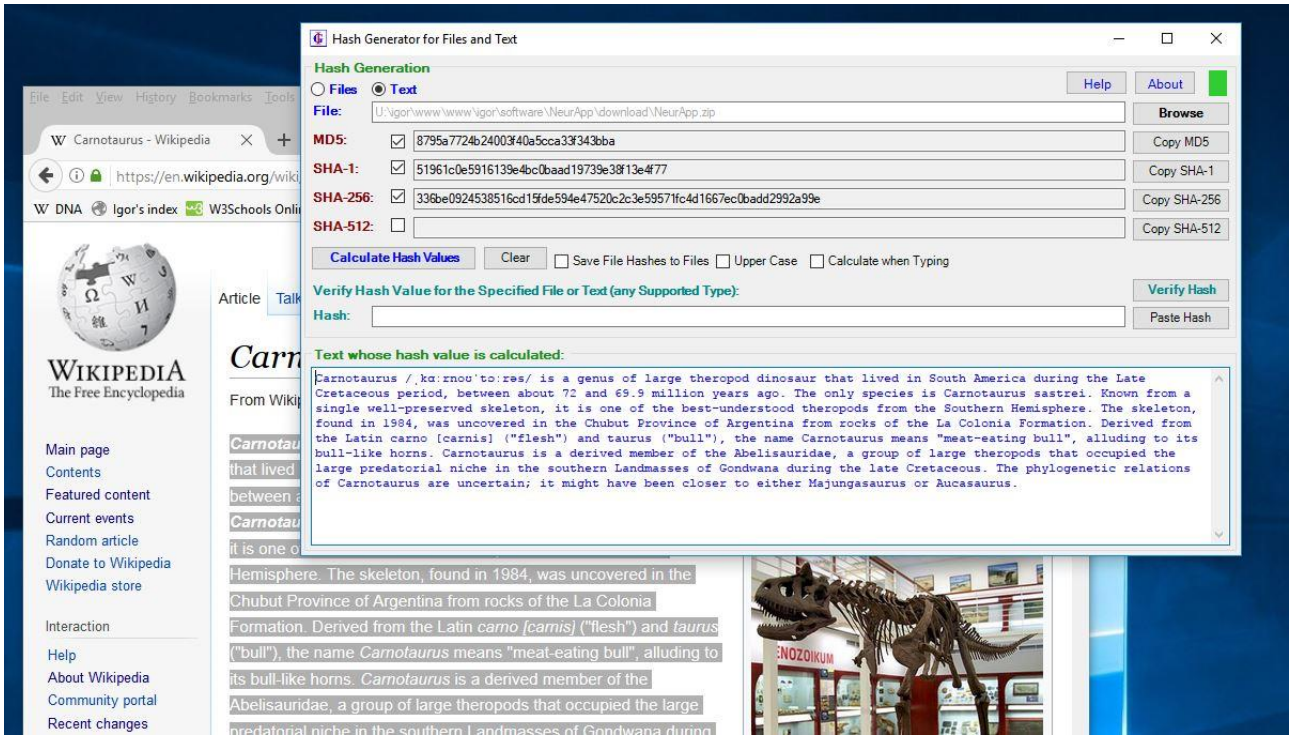


Figure 9: Text can be dragged and dropped into text box at the bottom. In any application that supports selection and dragging of text (browsers, office applications, word processors, text editors, etc.), select the text whose hashes need to be calculated, and simply drag the text into the textbox at the bottom. If in file mode, application automatically switches to text mode.

In practice, calculating checksums of text is not as common as calculation of file checksums. There still exist applications of this, e.g. one may want to verify the integrity of messages sent by another party. The originator may send a large text message via a common communication channel such as e-mail, and then checksum of the text that was sent via another communication channel such as SMS. The recipient can check whether checksum of the actually received text corresponds to the checksum sent by the originator of the message via a different communication channel. If the message was changed during transmission, the checksums would not match.

2.4 Accessing Help and Web Pages

HashForm is a very intuitive and simple to use software. Most of the time, it can be used without special instruction or referring to documentation.

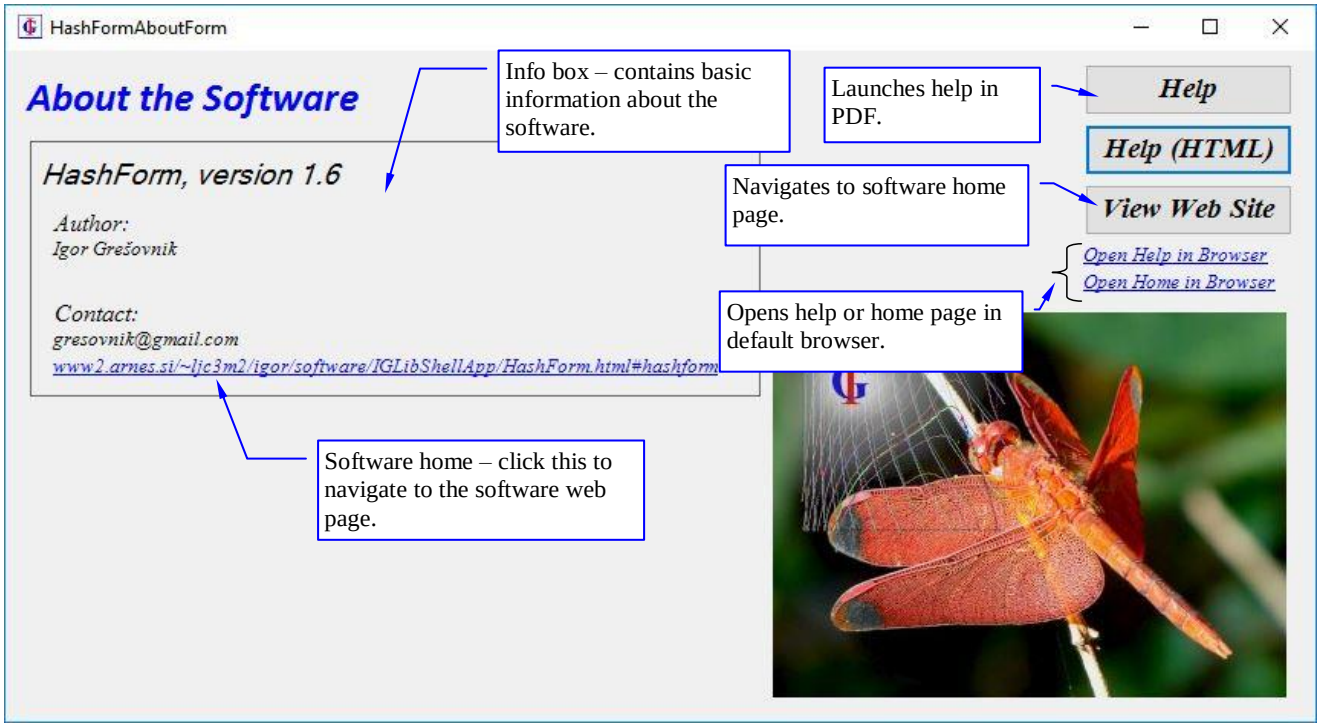


Figure 10: Accessing help and other information. This window is opened by clicking the "About" button.

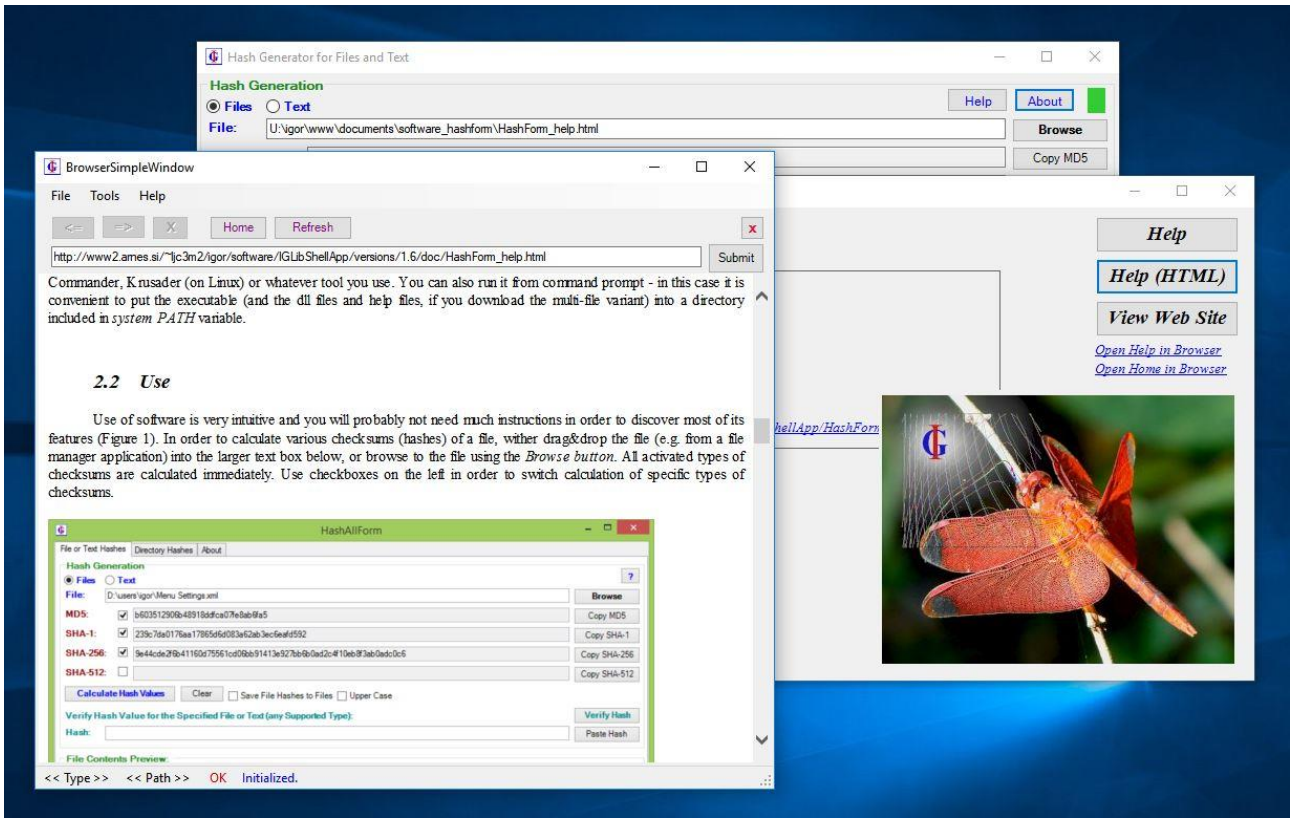


Figure 11: Information about the software can is accessed by clicking the "About" button. This opens a box with some basic information such as software version. From there, you can open documentation or home page either in a built-in browser or in system's web browser or PDF viewer. Clicking the "Help" button in the main application window opens the user manual directly.

2.5 Platforms and System Requirements

HashForm can be run on different platforms. It is based on the .NET framework, therefore, the framework or one of its variants must be installed on computer in order to run the software. .NET is usually pre-installed on Windows systems, therefore no additional installations are required in order to run the software. On other platforms, Mono framework must be installed on computer. Mono framework is freely available for Windows, Linux and Mac OS X.

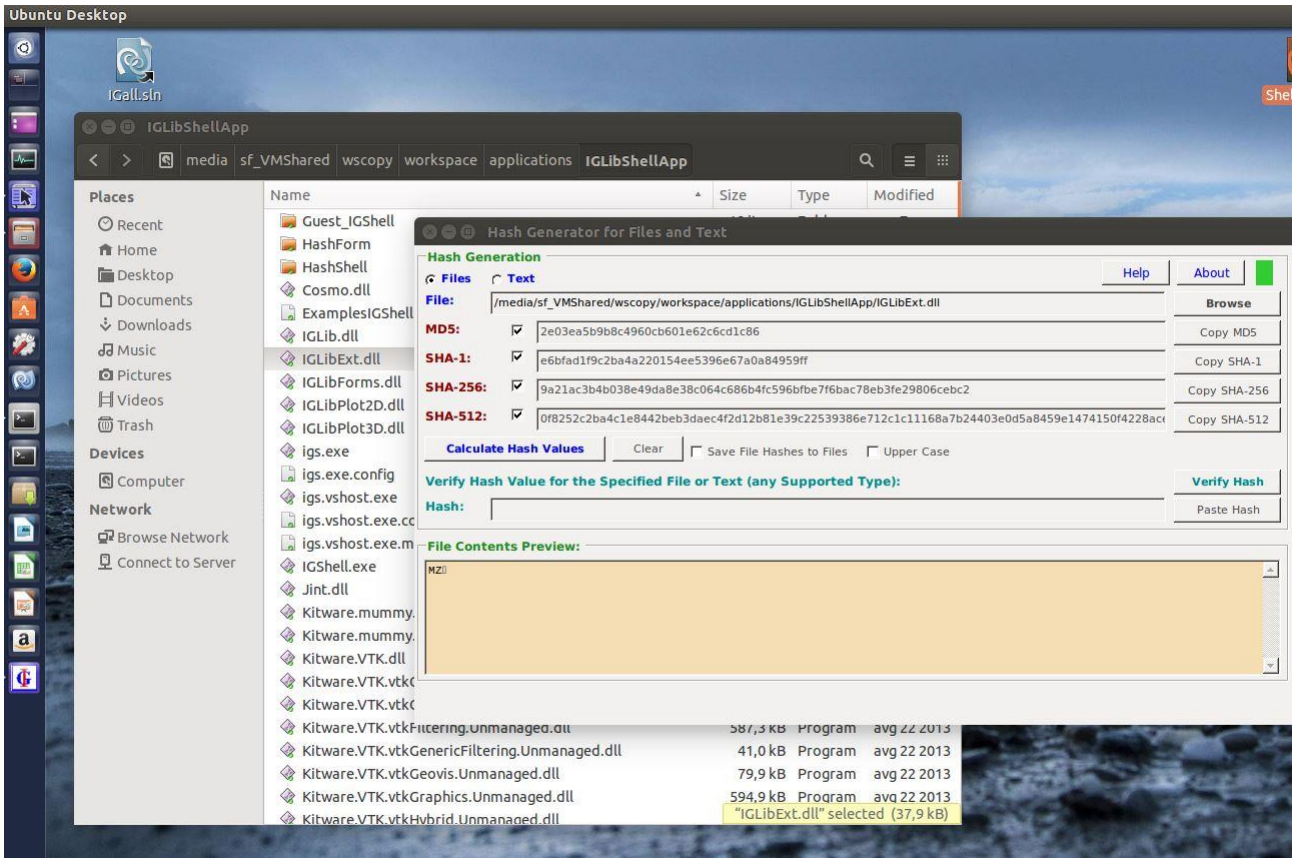


Figure 12: HashForm running on Ubuntu Linux.

3 ABOUT THE SOFTWARE

Back in 2008, I was taking care of securing a couple of multitier business applications built on the .NET platform. At that time, I created a simple cryptographic command-line application for my personal use - for encrypting sensitive data stored on my computer or the internet (on public sites or services such as Dropbox) and for data integrity checks.

While command-line application was very convenient for many tasks, I found it would be more comfortable to have a simple user interface for calculating and verifying file checksums. I created the first version of *HashForm* in 2008 and made it available for public. The emphasis in design was on good integration with the operating system and file managers and on being able to work really quickly when only calculation and verification of standard hash functions was needed file by file.

In April 2017, I corrected some bugs and added this user manual.

For more many operations I continue to use the command-line application (*HashShell*), e.g. for verification of data integrity of whole directory structures or encryption. When calculation or verification of cryptographic hashes (checksums) of a few files is all that is needed, I find

HashForm more efficient to use because of its graphical interface, drag & drop and browsing functionality.

4 SOME LINKS

- *HashForm* home.
- *HashShell* - a cryptographic command-line shell, which includes functionality provided by *HashForm* and many other functionality, in addition (including symmetric and asymmetric encryption, keys and certificate manipulation, etc.).
- *IGShell* - IGLib's shell application, set of selected or demonstrational functionality provided by *IGLib*, accessible through command-line interpreter (shell). Interactive mode is possible. *IGShell*, can also be used to launch the *IGForm* as its embedded application. *IGShell* also includes the complete *HashShell* as its embedded application.
- *Investigative Generic Library (IGLib.NET)* - a library on which the software (including *HashShell* and *IGShell*) is based. Some other software:
  - NeurApp – an educational application for visually exploring features of function approximation with artificial neural networks (ANN). It creates 1D and 2D ANN models of user defined functions and provides visualization capabilities to compare these models with originals. *NeurApp* can be downloaded from Softpedia.
  - *AnnApp* – software for exploring multidimensional ANN-based models.
  - *IGShell* - a shell application based on *IGLib*.
  - *Oscillator Modeller* – an educational application for playing with nonlinear driven oscillators and their resonance curves.
- Wikipedia: Checksum - explains checksums and how they are used to verify data integrity, especially, whether data has been changed during storage or transmission (e.g. download from internet).
- Wikipedia: Cryptographic hash function - hash functions that can also be used as checksum algorithms. These functions have additional requirements that make it difficult for malicious software or persons to falsify results of data integrity tests. Beside calculation of checksums, cryptographic hash functions are used for several other purposes, including password protection and data authenticity (in conjunction with asymmetric encryption algorithm).
- Wikipedia: Data integrity - Explains the topic in more detail.
- .NET Framework - download site for Windows. *HashForm* must have one of the.NET framework variants installed that include the *WinForms*. .NET framework is already installed on most Windows machines, so you will probably not need to download it if you run the application on Windows.
- Mono Framework - download site for Windows, Linux and Mac OS. Mono includes implementation of WinForms and can therefore be used to run *HashForm*. The application was tested on Linux with Mono and works reasonably well. On Windows machines I recommend using Microsoft's .NET framework, which is usually preinstalled on computers with Windows, so you don't need to download and install the framework yourself.