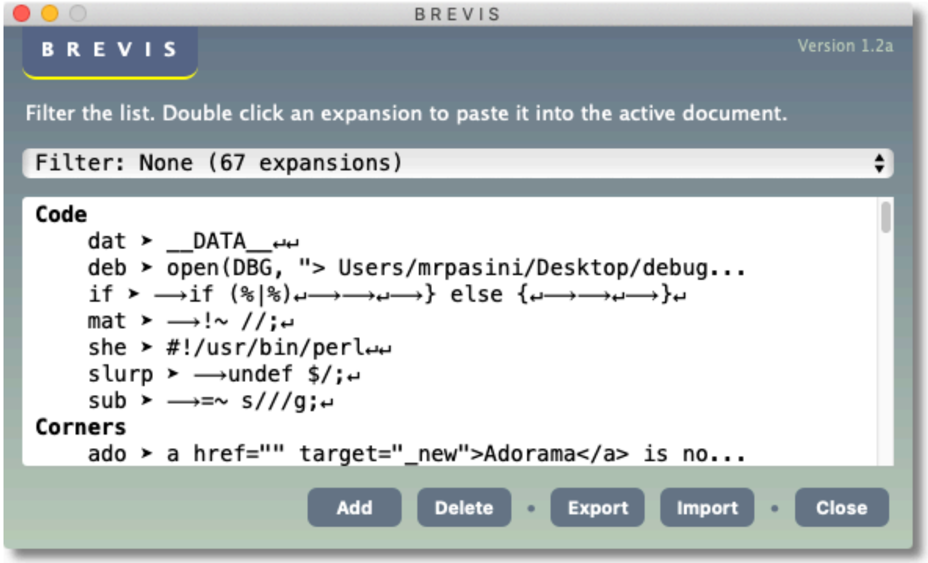
Brevis

Name: **Brevis**
Author: **Mike Pasini**
Version: **1.2a**
Last Update: **10 April 2022**



Filter the list. Double click an expansion to paste it into the active document.

Filter: None (67 expansions)

```
Code
    dat ➤ __DATA__↵↵
    deb ➤ open(DBG, "> Users/mrpasini/Desktop/debug...
    if ➤ ⟶if (%|%)↵⟶⟶↵⟶} else {↵⟶⟶↵⟶}↵
    mat ➤ ⟶!~ //;↵
    she ➤ #!/usr/bin/perl↵↵
    slurp ➤ ⟶undef $/;↵
    sub ➤ ⟶=~ s///g;↵
Corners
    ado ➤ a href="" target="_new">Adorama</a> is no...
```

- A single three-macro system for all text expansions
- Text expansions are organized in categories
- One global variable stores all your text expansions
- Initial character of typed string trigger modifies behavior
- Support for tokens, including the insertion point token
- Two separate utilities to edit your text expansions in a text editor
- Compatible with existing Keyboard Maestro text expansion macros
- Control Panel to list, filter, add, delete, export and import expansions
- Create a text expansion from selected text, including emoji, etc.

INTRO

Brevis (*ars longa, vita brevis* or *endless work, brief life*) is a compact, efficient system for managing a collection of single action, insert-text expansions triggered by typed strings.

In short, it automates text expansions.

Text expansions allow you to type a short abbreviation for a longer string of characters, making it quick to reliably enter frequently used blocks of text.

Usually with Keyboard Maestro, you would create an entirely new macro for each text expansion, organizing the macros in various macro groups. The Keyboard Maestro forum offers several macros that create one from a selection to make that part easier but it's often hard to remember less frequently used expansions and it can be unwieldy to launch and hunt around the Keyboard Maestro editor to edit or delete them.

Rather than using the Keyboard Maestro editor itself to manage a collection of separate macros for each text expansion, Brevis provides both 1) an easy-to-use user interface to create, run, list, export, import and merge them as well as 2) a faceless way to create and run them.

Whether you need a quick text expansion temporarily or can't remember the ones you've already recorded, Brevis quickly comes to the rescue.

It's easy to create one from a selection (based on an early macro by Peter N Lewis). You won't forget it because Brevis displays a list of expansions in alphabetical order organized by categories that can be displayed for execution with a keystroke. And you won't have to open the Keyboard Maestro editor to manage your text expansion macros because Brevis provides tools to do that too.

It also has few other special features.

A Versatile Trigger

Among those is a flexible typed string trigger, which is independent of the shortext itself. So `,this` and `;this` and `=this` can all call the same expansion.

Why?

Because Shortext can use the leading punctuation to trigger variations in playback. For example, by default the semi-colon will insert text *by typing* instead of *by pasting*. And the equals sign will swallow the non-word character that closes the trigger.

You can expand those sorts of options to meet your own requirements in the macro itself, of course.

Expansion Features

In addition to straight text expansion (eg. **`km`** into **`Keyboard Maestro`**), Brevis can reposition the cursor within the expansion, process Keyboard Maestro tokens for things like data handling and window names, plus record tabs and returns.

Categories

To make it easier to work larger collections of text expansions, the Brevis Control Panel lists them by categories like Typinator and TypeIt4Me.

You can also filter the list to display only one category, too. So if you have a list of expansions you use to write HTML, for example, you can filter the list for them and just click any expansion ilisted n the Control Panel to apply it as needed.

Space Saving Design

The typical text expansion macro XML export is about a 5K export. This system of three macros to create, run and display your expansions is about a 63K XML export. So if you use a dozen or more text expansions, Brevis will keep a lid on the size of your Keyboard Maestro plist even as you add more and more expansions to it.

At last count I had nearly 50 text expansions managed by Brevis. That would expand the Keyboard Maestro plist by about 250K or 190K more than using Brevis.

One Global Variable

Brevis remains compact because it stores all the triggers and expansions in one JSON-formatted global variable rather than in the Keyboard Maestro plist itself.

That’s what makes Brevis tick.

You can export that variable as a text file for backup and import a text file of expansions from another system into your existing expansions.

And you can export a filtered list of expansions to share with colleagues. So if you’re working on a document that requires a lot of acrobatic typing, you can export the expansions you use to complete your feats without failure and colleagues can merge them into their own Brevis collections.

INSTALLATION

Brevis requires three macros in its own group, which installs as **`MacroApp: Brevis`**:

- Control Panel
- Create From Selection
- Run Brevis

All of the text expansions are stored in **`Brevis`**, a single global variable with the same name as the system, which can be exported to and imported from a text file using the Control Panel. It's a stringified JSON file with three fields: category (**`c`**), trigger (**`t`**) and text (**`x`**). The single letter field names (as well as the compact, non-prettified JSON) help minimize the variable's size.

Another global variable **`Brevis Path`** retains the name of the export directory.

The macros themselves use local variables extensively and are coded in native Keyboard Maestro actions, Perl and JavaScript.

To install the system, double click on **`MacroApp/ Brevis Macros.kmmacros`** and enable the group.

The .json filea **`Brevis.Dates.json`** and **`Brevis.Emoji.json`** are included in the distribution as an example of date token and emoji expansions. You can use Brevis to **`Import`** them into your system.

Optional Utilities

The Brevis distribution also includes separately two utilities:

`Utility: Edit` pastes a prettified (so you can read it) JSON-formatted text file of all your expansions in a BBEdit window, ready for editing.

`Utility: Save` copies that JSON to the clipboard where it becomes compact JSON before being written back to the **`Brevis`** global variable.

You can install them by clicking on **`MacroApp/ Brevis Macros.utilities.kmmacros`**. They will be added to the **`MacroApp/ Brevis Macros`** group.

There's more about using these optional utilities in the *Usage* section of this document.

TRIGGER

The default typed string trigger is a regex:

```
[,;=+*^][:alpha:][[:alnum:]-]+\W
```

Translated into plain English, that's 1) one of several punctuation characters not likely to conflict with real-world text 2) followed by a letter 3) followed by at least one additional letter, hyphen or numeral but any number of them 4) up to a non-word character (like a space, period, quote, etc.).

Two of the initial punctuation characters change playback behavior:

- `;` will insert the expansion by typing, otherwise it's pasted
- `=` will swallow the non-word closing character from.

For example, when working in InDesign, I use `;` to signal my expansions by typing. And when I'm filling out a Web form or a table in HTML, I use `=` to avoid extra spaces at the end. But normally, I just continue typing with the next non-word character, like a space, a period or a quotation mark.

The hyphen is not likely to conflict with real-world text and makes it possible to organize your triggers more memorably. For example, `bk-title` might trigger CSS for a book title variation while `bk-byline` might trigger CSS for the byline.

If the default set doesn't include it, you may add your favorite initial character to the first set of brackets in the Run Brevis trigger, the only place it appears.

You can customize the playback behavior of a expansion further down in Switch action of the Run expansion macro but it isn't necessary.

Conflict Elimination

Because Brevis waits for a non-space character before evaluating the trigger, it does not conflict with any other Keyboard Maestro you may have created that uses a similar initial punctuation character.

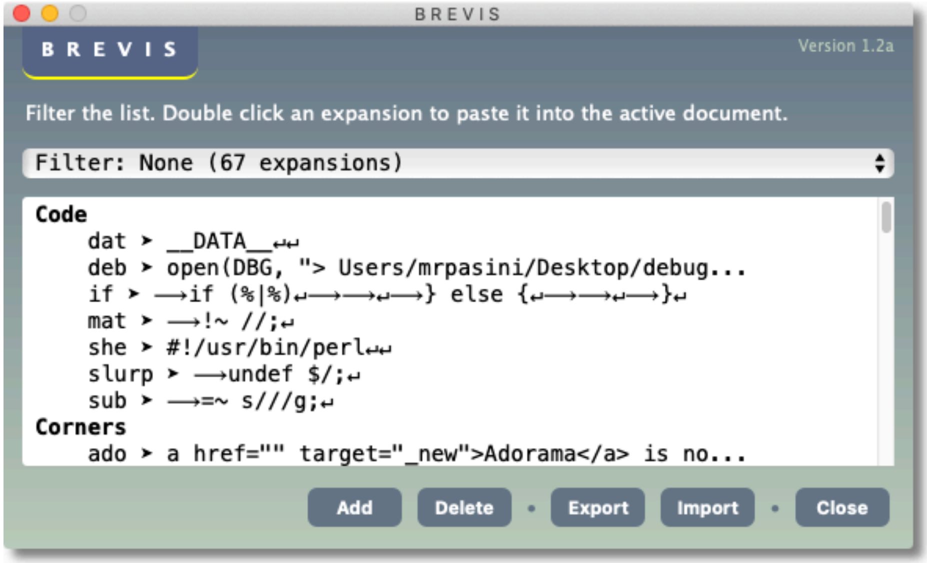
I have multiple-action text expansions for date processing and variations on a trigger that begin with a semi-colon and an equals sign and they work as expected without conflicting with Brevis.

USAGE

You can quickly create an expansion from a selection using a hot key for the **`Create From Selection`** macro or create one from scratch using the Control Panel, which also offers a few other options.

Control Panel

The Control Panel can be triggered by **`Shift-Option-Comma`** or the system menu.



Filter the list. Double click an expansion to paste it into the active document.

Filter: None (67 expansions)

```
Code
    dat ➤ __DATA__↵↵
    deb ➤ open(DBG, "> Users/mrpasini/Desktop/debug...
    if ➤ ⟶if (%|%)↵⟶⟶↵⟶} else {↵⟶⟶↵⟶}↵
    mat ➤ ⟶!~ //;↵
    she ➤ #!/usr/bin/perl↵↵
    slurp ➤ ⟶undef $/;↵
    sub ➤ ⟶=~ s///g;↵
Corners
    ado ➤ a href="" target="_new">Adorama</a> is no...
```

It displays one line of text help before two popup lists:

A popup **Filter list** that includes a status line showing which filter is in effect and the number of expansions available in the filtered list

A list of the **available expansions** in alphabetical order by trigger so you don't have to remember them and can easily find one.

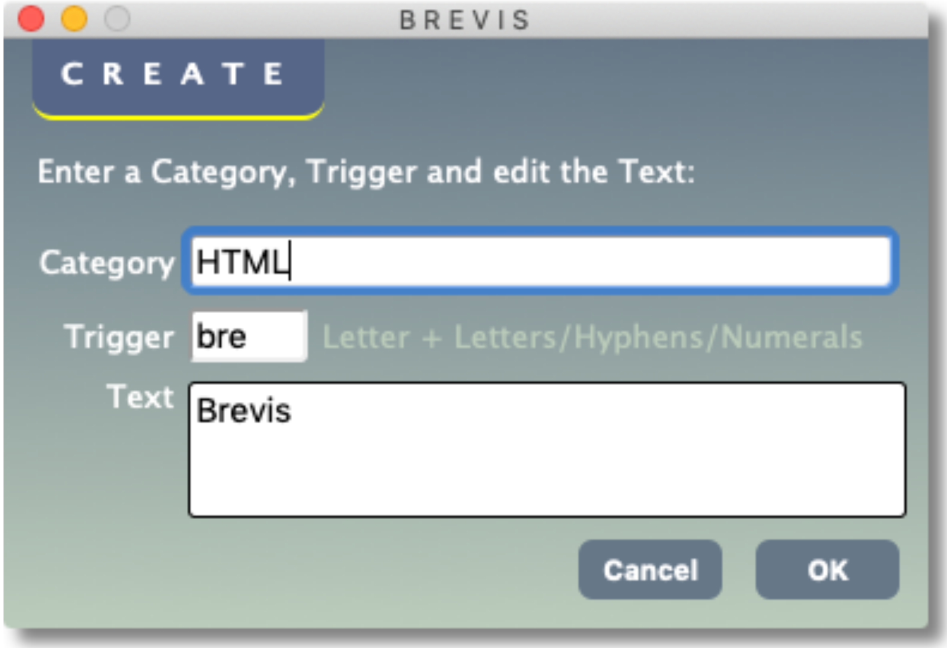
The filters are simply the categories in which you've organized your expansions. The last option is always **`None`**, so you can remove any filter to see the whole list.

If an expansion is too long to display on one line, the expansion will be truncated and followed by an ellipse. If the expansion contains tabs or newlines, they will be represented with symbols so each line in the Control Panel contains only one expansion.

You can navigate the list from the keyboard. Only one selection is possible, an inconvenience only if you are deleting expansions. Select an entry and press the Return key or double click an entry to insert it into the active document.

There are also five buttons along the bottom of the Control Panel that include Add, Delete, Export, Import and Close.

Add. Presents a dialog box for entering a new trigger and expansion.



Enter a Category, Trigger and edit the Text:

Category: HTML

Trigger: bre

Text: Brevis

Delete. Removes the selected expansion from the global variable Brevis.

Export. Writes the current set of expansions stored in the global variable Brevis to the file you indicate. The output directory is remembered in the global variable Brevis Path.

If a filter is in effect, only the filtered expansions will be written to the text file. The suggested name of the text file will include the category name (eg. `Brevis.Code.json`). That makes it easy to share limited sets of expansions with others. But it's only a suggestion.

Import. If the JSON file name you are importing ends in “None.json” then Brevis completely replaces whatever the global variable Brevis currently has with the Brevis in the text file you select for import.

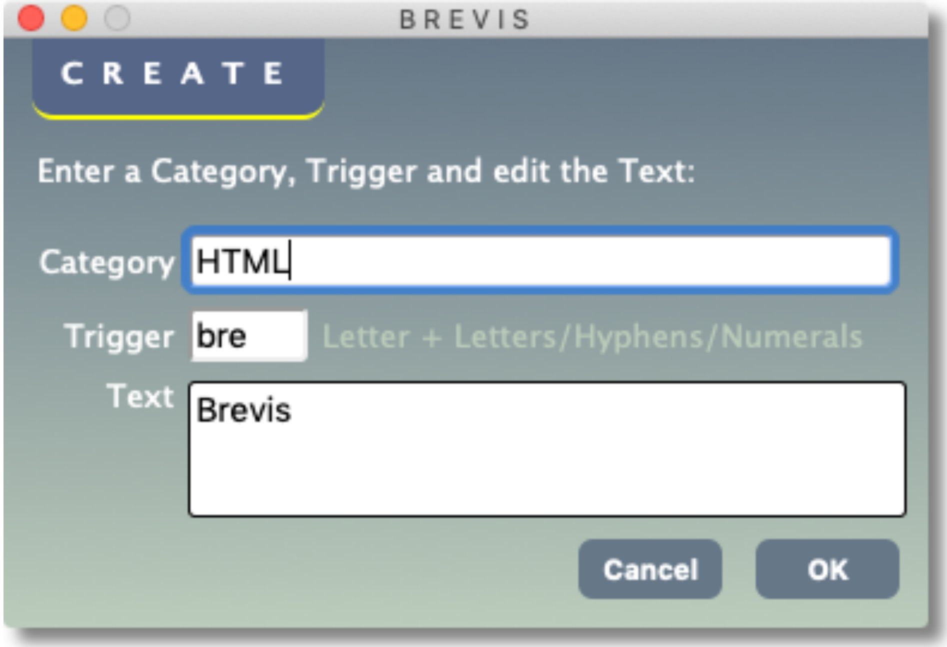
But if the JSON import file does not match “None.json,” Brevis imports any of the import file’s expansions that do not already exist in the global variable Brevis. It checks only the trigger for duplicates, though. This behavior was previously available from the Merge button, which has been removed.

Close. Closes the Control Panel.

In addition, you can insert an entry displayed in the list in two ways. Double clicking on the entry or selecting an entry and pressing the Return key. Either will insert it by pasting in the active document.

Create From Selection

The macro to create an expansion can be triggered by **`Shift-Option-C`** or the system menu. Expansions can also be created from scratch via the Control Panel's Add command, which uses the same macro.



Enter a Category, Trigger and edit the Text:

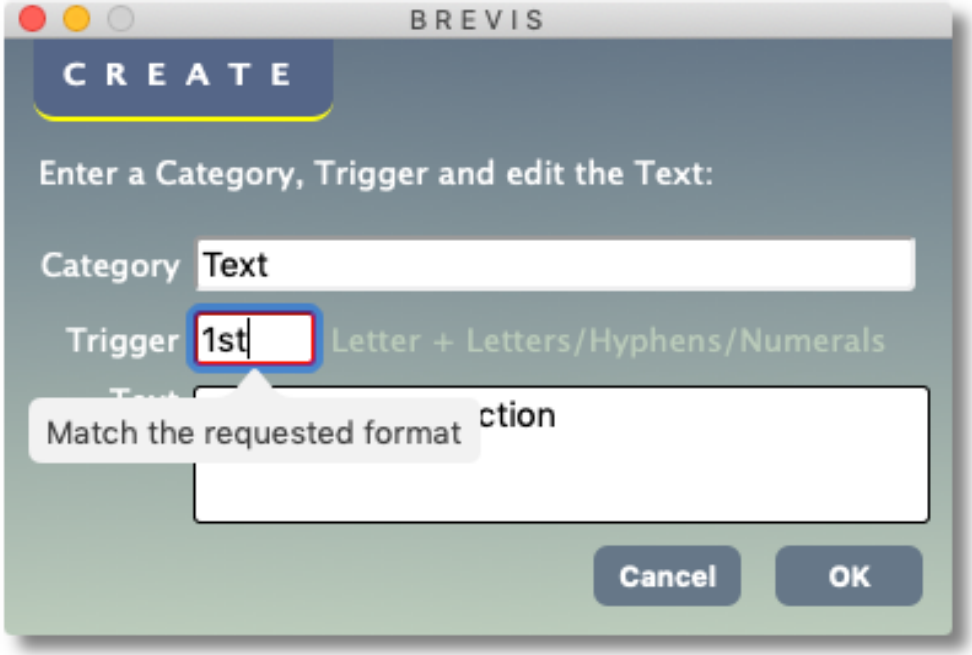
Category: HTML

Trigger: bre

Letter + Letters/Hyphens/Numerals

Text: Brevis

The three fields are checked for errors as you tab through them. If an error is detected the border of the entry field turns red. If you click on the OK button, a text bubble appears near the first problem to explain how to correct the problem, as in the illustration below.



Enter a Category, Trigger and edit the Text:

Category Text

Trigger 1st Letter + Letters/Hyphens/Numerals

Match the requested format

ction

Category

The last used category will be suggested as the default category. Simply type over it to enter anything else, including creating a new category. This will be blank if you use the Add button from the Control Panel.

Categories are stored full text in the global variable so it’s wise to choose brief names when possible. So “Cascading Style Sheet Code” is less efficient that “CSS” would be.

Trigger

The first three characters of the text selection will be lowercased as a suggested expansion in the subsequent prompt. There's nothing magical about three letters but that avoids most conflicts without requiring excessive typing.

The first character must be a letter (to ensure it’s unlikely to be real world text) but after that letters, numbers and the hyphen are all legal.

Text Expansion

Quotes, tabs and newlines are all legal characters for expansion. The global variable stores them literally but the exports escapes quotes and translates the others into symbols.

The insertion point token (`%|%`) is also supported in expansions. Inserting those three characters into an expansion will place the cursor at that point after expansion.

You can also use other text tokens including ICU dates and Keyboard Maestro tokens like `%SafariTitle%`.

You can add an expansion from the Control Panel's Add button or selected text with the hot key trigger or via the system menu. The macro will not add the expansion if the trigger already exists.

New expansions are added to the top of the list in the global variable (although they are displayed in alphabetical order by category in the Control Panel) on the theory they have a more immediate need over the older ones. But I haven't noticed any difference in execution between my first and last entries, even with 50 of them.

Run Brevis

The Run Brevis macro to insert an expansion is triggered by the pattern of a leading punctuation character followed by a letter and then either letters or numerals before a non-word character.

It therefore won't conflict with any Keyboard Maestro macro that uses the same leading punctuation because Brevis is waiting for the non-word character whereas a Keyboard Maestro macro does not require that.

In short, Keyboard Maestro macros have priority over Brevis expansions.

If the trigger begins with a semi-colon or includes the insertion point token, the expansion will be inserted by typing, which tends to work better in word processing and page layout software that use stylized text. Otherwise it will be pasted, which is faster.

To expand the expansion, simply type it in the active document followed by any non-word character. If you use an equals sign trigger, only the expansion will be typed. If you use any other punctuation trigger, the non-word character (like a quote, period, comma, etc.) will be added to the end of the expansion on the assumption it's intentional punctuation.

Editing Expansions

You might expect some sort of editing option so you can tweak the trigger and expansion but there are issues with editing expansions.

You can certainly edit the `Brevis` variable itself from Preferences in the Keyboard Maestro editor if you are careful about observing the JSON syntax rules mentioned below.

But there's an easier and safer approach. To change how an expansion works:

- Expand it in a text editor
- Edit the text of the expansion
- Use Brevis to delete the old expansion
- Select the edited text
- Create a new expansion from the selection

The dialog box will allow you to change the category and trigger if you want.

So like any self-respecting text expansion, these are self-editing marvels. And that keeps Brevis lean and mean.

An Alternative

Of course as soon as I wrote that, I wanted to edit a few of my own expansions. Using the Control Panel would add a lot of overhead to load the Add window with the variables and delete the old entry from the variable before entering the new one.

Even then you'd just be able to edit one at a time.

So I've included two companion utility macros you can get from the same place as Brevis in a separate file:

Utility: Edit pastes a prettified (so you can read it) JSON-formatted text file of all your expansions in a BBEdit window, ready for editing.

Utility: Save copies that JSON to the clipboard where it becomes compact JSON before being written back to the `Brevis` global variable.

The first one is pretty safe stuff. If you prefer a different code editor than the free version of BBEdit, just change that reference in the macro.

The second one, however, can hose your expansions. Which is why the utilities are not bundled with Brevis.

How can you hose the global? Easy. BBEdit isn’t going to check your JSON for problems, so it’s possible you might write illegal JSON back to the variable. You can write illegal JSON if you do not use \n for newlines, \t for tabs and \” for actual quotes in expansions.

Note that both rely on BBEdit’s English language menu system so if you’re using a different language or text editor, adjust as necessary.

You can copy everything and run it through an online JSON lint validator like JSONLint, which is what I used during debugging. But if you play by the rules, that isn’t really necessary.

It is wise, however, to do an Export from the Control Panel *before* using these utilities. If you do happen to hose your global (the Control Panel will be empty), you can delete the variable and Import the exported text file and be no worse off than before.

NOTES

Following are a few notes about using Brevis.

Tokens

Brevis only handles “single action, insert-text expansions triggered by typed strings.” But because it can handle tokens, it’s a bit more useful than that may sound.

For example, entering the `%|%` insertion point token will position the cursor at that spot after expansion.

Other useful tokens include ICU dates for date formatting (like `%ICUDateTime%yyyy.MM.dd%` for “2021.08.26”) and Keyboard Maestro tokens (like `%SafariTitle%`). Brevis will expand these tokens before inserting the text in your document.

The text expansion itself happens to be a JSON token that is expanded into a variable which is subsequently filtered for any included tokens.

Variable Size

Keyboard Maestro observes a limit on the total size consumed by its global variables.

In the forum post KM Engine Log Concerns, Peter Lewis explained that Keyboard Maestro global variables are stored as Unix environment variables, which have a limit of about 256K. Script actions stop working if your total variable storage approaches that with a message in the Engine log that the variable has been “excluded from environment to ensure the environment is not too large.”

Brevis itself doesn’t much impinge on that limit, typically requiring no more space than a few paragraphs of text.

My `Brevis` global variable, for example, is 3.5K for 47 expansions, some of them rather long boilerplate. My 159 Keyboard Maestro global variables total just 9K, well below the 256K threshold.

While Brevis itself may not threaten that limit, if your Keyboard Maestro installation is already near the limit, you might keep this limitation in mind. You can minimize the storage requirement by deleting variables that don't require permanent storage or using local or instance variables where applicable.

Keyboard Maestro Size

The size of the Keyboard Maestro plist file, in which all your macros are stored, is also of some concern. Peter Lewis remarked on one occasion that the application was designed for a plist file size of 1.5-MB, which accommodates hundreds of macros.

At 1.1-MB I have about 500 active macros, none of which store images. Of that 63K is taken by Brevis. The two optional utilities add 9K.

Many Keyboard Maestros exceed the 1.5-MB target by quite a lot with no issues. The most noticeable issue would be sluggish Keyboard Maestro Editor behavior.

And when there are issues, they are often traced to storing images in the plist file. These might be custom icons or, more likely, an image stored with the **`Find Image on Screen`** action.

Toggler

I use my Toggler macro to enable/disable **`Run Brevis`**, the macro that executes an expansion. There are times when it can be inconvenient to have these enabled (like editing minified CSS).

Other Expansion Tools

Keyboard Maestro itself is the ultimate shortext tool, really, watching for typed strings of any length with no specific precede and able to transform that shortcut into anything once triggered.

Brevis, in contrast, is just a very quick and efficient way of managing lots of simple text expansions that are all stored in just one Keyboard Maestro variable that needs just one macro to make numerous expansions.

Typinator and TypeIt4Me are often cited in the Keyboard Maestro forum (among others) as better ways than Keyboard Maestro to handle a large number of text expansions. And certainly Brevis is not as comprehensive a solution as any of those.

But Brevis does handle a class of single action, insert-text expansions efficiently and makes it very easy to use them. I use individual Keyboard Maestro macros outside Brevis to handle date math, calculations and links. Those kinds of expansions rely on more actions to do processing in addition to inserting text. But Brevis does my routine text expansions.

JavaScript

As I worked out various features of Brevis, I usually inserted action after action. As these became more complex, they often became unwieldy so I recoded them in JavaScript.

On two occasions this strategy saved 5K each in the size of Brevis, collapsing half a dozen Keyboard Maestro actions into one **`Execute JavaScript for Automation`** action. You can see these primarily in the Control Panel code.

I would normally have been inclined to use Perl, but the macOS distribution of Perl doesn't natively handle JSON formatted files whereas JavaScript does, making it a little easier to share this solution.

Brevis does use Perl elsewhere to handle text massaging where all it had to do was grab an environment variable.

In any case, I was delighted with the space savings.

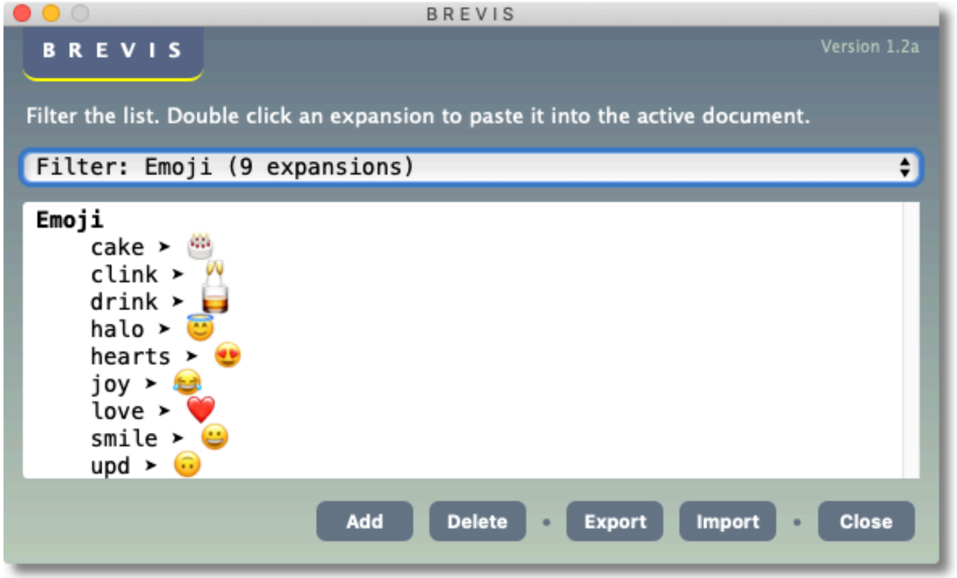
Scrivener

This documentation is the first time I've used Scrivener to produce a PDF for my macro projects. It relies on a custom Scrivener compile format.

It's been a big advance over using Pages for PDF generation, loading faster and behaving as expected.

Emoji, Special Characters

I was looking into the free macOS utility Rocket to make it easier to enter emoji into text when it occurred to me that Brevis might be able to do the same thing. Indeed it can:



Filter the list. Double click an expansion to paste it into the active document.

Filter: Emoji (9 expansions)

Emoji
cake ➤ 🎂
clink ➤ 🥂
drink ➤ 🥃
halo ➤ 😇
hearts ➤ 😍
joy ➤ 😂
love ➤ ❤️
smile ➤ 😀
upd ➤ 🙃

In my case, I used a text description for the emoji. And I'm happy to report it works exactly as you would expect. For special characters, too.

The easiest way to build a set of these expansions is with the `Create from Selection` option. Paste your favorites into a document from the menu bar `Show Emoji & Symbols` app, select each one and run the macro giving each a trigger.

Feedback

While Brevis itself has undergone many redesigns and revisions (including three major design revisions) and comprehensive if not complete testing, I haven't been using this version very long, so I'm happy to hear about any issues and promise a prompt fix.

You can reach me on the Keyboard Maestro forum as `@mrpasini`.