Keyboard Maestro 10 User Manual

- Troubleshooting
- Support
- Glossary
- Administrative Details

Overview

Keyboard Maestro will take your Macintosh experience to a new level. Keyboard Maestro enables you to create or record custom macro shortcuts that you can activate at any time. For example, your macros could help you navigate running applications or work with an unlimited number of clipboards. Best of all, every macro you create is available using simple keystrokes you choose or a variety of other triggers. The only limit to Keyboard Maestro is your imagination!

Using Keyboard Maestro's powerful Macros, you can make your Mac behave the way you want it to behave – open documents when and where you want them, type sentences with the press of a key, expand abbreviations into entire paragraphs, control web applications, and much more. You simply define what you want your Mac to do and when you want it done.

Keyboard Maestro comes complete with a clipboard history, saving everything you copy for later use so you'll never lose something on your clipboard again, as well as named clipboards where you can store commonly used images or text.

Keyboard Maestro also includes a powerful Application Switcher and Window Switcher so you can cycle through applications or windows, closing, hiding, launching, and more as well as an Application Launcher that lets you quickly launch applications.

Keyboard Maestro requires a Mac running macOS 10.13 High Sierra or later.

Keyboard Maestro is free to try with no limitations. Once the trial period ends, a license must be purchased to continue using it.

What's New?

Keyboard Maestro 10 expands on the powerful base of previous versions, improving the editor, adding many new actions and triggers, Paste by Name, status menu display, subroutines, and more. Keyboard Maestro 10 requires macOS 10.13 High Sierra or later.

This version is dedicated to Jim Underwood (JMichaelTX [https://forum.keyboardmaestro.com/u/jmichaeltx]), who sadly passed away this year, with thanks for his outstanding assistance and generosity on the forum, spending thousands of hours helping folks get more out of Keyboard Maestro. He will be sorely missed.

Changed in 10.2

- Added option to Press a Button action to wait for the button to exist and be enabled.
- Added option to Prompt With List action to Always Show All Entries. (forum [https://forum.keyboardmaestro.com/t/is-it-possible-to-re-populate-all-in-a-prompt-with-list/26258])
- Added option to Prompt With List action to not trim white space from entries. (forum [https://forum.keyboardmaestro.com/t/user-prompt-with-list-feature-request-turn-off-trim-spaces/24423])
- Added PromptWithListModifiers token. (forum [https://forum.keyboardmaestro.com/t/use-modifiers-with-prompt-with-list/23394])
- Added PromptWithListShowAllLimit (default 100) hidden preference.
- Added support for dragging files on to applications in the Application Switcher.
- Added support for com.vivaldi.Vivaldi.snapshot browser.
- Added Quote for Process Tokens filter.
- Added kCGMouseEventDeltaX/kCGMouseEventDeltaY to mouse moved events. (forum [https://forum.keyboardmaestro.com/t/generate-low-level-mouse-movement-events/27923/6])
- Request Screen Recording permission when PIXEL function is used.
- Fixed the Send SMS/iMessage action in recent macOS systems.
- Fixed an issue with executing shortcuts that contain non-ASCII characters in their names.
- Fixed an issue with ⌘⌥F global search erroneously adding ⌘ or ⌥ character to the search field.

- Ensure that the Clipboard History is saved every three hours (if saving is enabled).
- Ensured image specification text is searchable in the editor.
- Hopefully remove the alert when switching to an enabled keyboard layout.

Changed in 10.1.1

- Fixed a case where simulating certain keys (arrows, function keys) could trigger the matching Hot Keys.

Changed in 10.1

- Added Execute Shortcut action.
- Added Edit Shortcut action.
- Added Execute or Edit Shortcut (based on the option key) action.
- Added Get Macro, Edit Macro, and Execute Macro Shortcut Actions.
- Added Get Variable and Set Variable Shortcut Actions.
- Added Get Active Macros Shortcut Actions.
- Added Calculate Shortcuts Action.
- Added Process Tokens Shortcuts Action.
- Added Search and Search & Replace Shortcut Actions.
- Added support for Shift paste plain in Paste by Name action.
- Added support adjusting modifiers in Keystroke selections (including pseudo-modifiers).
- Fixed AppleScript support for catchactions in Try/Catch action.
- Stop implicit reading of variables by web browser actions counting as a variable use.
- Ensure recorded Insert Text by Typing action has token expansion disabled.
- Append Text to a File action should not overwrite a file if the file exists but cannot be read.
- Fixed an issue when deleting a condition if there are two other conditions that are identical.
- Fixed an issue with AppleScript search and search & replace commands not honouring the case sensitive flag.
- Resolved an issue simulating hot keys in recent systems by supporting Numeric Pad and Function pseudo-modifiers. (forum [https://forum.keyboardmaestro.com/t/tip-resolving-catalina-mojave-accessibility-security-permissions-issues/16620])
- Resolved an issue where script error messages could be truncated.
- Restored the MouseGetCountdown hidden preference. (forum [https://forum.keyboardmaestro.com/t/is-there-a-way-to-set-default-waiting-time-to-get-coordinates-from-5-seconds-to-3-seconds/14980/10])
- Removed a dependency on /bin/bash for updates.

Changed in 10.0.2

- Added window.KeyboardMaestro.Log to Custom HTML Prompt functions.
- Action Notes are searched when searching for matching actions.
- Added "note:" Search Strings.
- Make Status Menu icons a little larger by default.
- Fixed blurry icons when using Stream Deck Set Image action.
- Fixed Click Mouse action's Get facility Y coordinate being inverted.
- Ensure Finders Selection collection returns paths without trailing slashes.
- Ensure FinderInsertionLocation and FinderSelection(s) tokens return paths without trailing slashes.
- Ensure Display Progress window honours Next Engine Window position.
- Adjusted some clipboard popup menus to not extend all the way across an action.
- Adjust Select All to not initially include the Unmatched Macros entry.
- Adjusted Create Unique File action to include "copy n" before the extension, not after it. (forum [https://forum.keyboardmaestro.com/t/create-unique-file-action-in-keyboard-maestro-v10-0-1/25126])
- Fixed Fast User Switch action in Monterey.
- Fixed Custom HTML Prompt not honouring *data-kmwidth*, *data-kmheight* etc.
- Fixed adding "Before Version 10" revision.
- Fixed some cases where revision history was not saved. (forum [https://forum.keyboardmaestro.com/t/km10-macros-not-backed-up-for-several-weeks/24893])

Changed in 10.0.1

- Fixed a crash if the Cochin Italic font is missing or disabled.
- Fixed a crash when recording actions.
- Changed the editor Get mouse location to include delay to allow window selection. (forum [https://forum.keyboardmaestro.com/t/new-get-location-and-drop-down-list-km-v10/24477])
- Adjusted dragging Subroutine macros to action lists or to make macro aliases to use Execute Subroutine action. (forum [https://forum.keyboardmaestro.com/t/dragging-and-dropping-a-subroutine-macro-creates-an-execute-a-macro-action-v10/24498])
- Process Text Tokens in Display Progress action. (forum [https://forum.keyboardmaestro.com/t/km-10-new-progress-bar-action-and-variables/24542])
- Improved remembering the position of Macro Group status menu items. (forum [https://forum.keyboardmaestro.com/t/possible-menu-bug-menu-bar-status-menus-change-their-order-after-i-set-them/24547])
- Corrected typo of Miscellaneous in Icon Chooser SF Symbols.
- Fixed showing status menus when macro group activation is Always active and variants. (forum [https://forum.keyboardmaestro.com/t/feature-request-add-optional-customizable-status-menus/19156])
- Fixed an issue with Command-Option-Numbers hiding but not closing the Clipboard History Switcher. (forum [https://forum.keyboardmaestro.com/t/selection-by-number-in-clipboard-history-switcher/4470/])
- Fixed Prompt for User Input variable type popup menu being in front of variable value field.
- Fixed Custom HTML Prompt window not responding to key presses when the title bar is disabled (forum [https://forum.keyboardmaestro.com/t/html-prompt-not-responding-to-esc-or-custom-keypress-events-when-title-hidden-v10-0/24478])
- Fixed %FinderSelection% token including a trailing end of line character.
- Fixed Set Next Engine Window Position action Center and Center At options. (forum [https://forum.keyboardmaestro.com/t/km-10-move-or-set-next-engine-window-position/24593])
- Fixed Search the Web action failing to activate the web browser.

Editor

- Added configurable Favorite Actions. (forum [https://forum.keyboardmaestro.com/t/can-i-add-custom-action-to-other-actions-and-for-it-to-be-searchable-and-have-a-custom-name/4843]) (forum [https://forum.keyboardmaestro.com/t/macro-kmfam-favorite-actions-and-macros/4854])
- Added Select Macro by Name to the macro selector popup. (forum [https://forum.keyboardmaestro.com/t/request-make-it-easier-to-choose-a-macro-for-actions-like-enable-disable-macro/22807])
- Added This Macro and This Macro Group options to the macro selector popup. (forum [https://forum.keyboardmaestro.com/t/request-make-it-easier-to-choose-a-macro-for-actions-like-enable-disable-macro/22807])
- Added search field to macro selector popup.
- Added Evaluate Condition Results option.
- Disable Evaluate Condition Results after action or safe macro import.
- Support dragging a .kmactions file in to an action list.
- Add Get functionality to Area selection for various actions like Move & Resize Window and Capture Screen.
- Used new Prompt for Screen Rectangle for Get facility of Click Mouse action.
- Added search field to Insert All Actions, All Functions, All Tokens, and Variables menus.
- Added Select Last Aborted Action menu item.
- Added `Or by Execute Macro` to macros editor listing the Execute a Macro actions referring to this macro. (forum [https://forum.keyboardmaestro.com/t/suggestion-macro-inspector-executed-by/8182])
- Added Cut, Copy & Delete to contextual menu for Macro Group and Macro columns.
- Added Copy as Execute a Macro, Set Macro Enable and Mark Macro actions to contextual menu for Macro column.
- Added Copy as Set Macro Group Enable and Toggle Macro Group actions to contextual menu for Macro Group column.
- Added Paste to the `No Action` drag target contextual menu when the clipboard contains actions.
- Support double-clicking dividers in the editor window to set ideal size.
- Support control-up/down arrow in action lists to move actions up/down. Also command-control for top/bottom. (forum [https://forum.keyboardmaestro.com/t/is-there-any-shortcut-to-move-actions/2244])
- Support shift-insert action to insert the action above the current selection. (forum [https://forum.keyboardmaestro.com/t/feature-request-insert-action-above/23927])
- Added Edit ➤ Insert ICU Date Field menu to insert the various ICU Date components.
- Added Edit ➤ Insert ICU Date Field By Name (^D) to insert the various ICU Date components by name.

- Added File ➤ Export as Folder and Export All Macros as Folder to export macros as individual files.
- Support tapping modifiers in editor search field to insert modifier symbol.
- Added Sort Macros by Size.
- Added size to Macro Inspector.
- Added "size:10000" to search filter.
- Added "And 99 Filtered Macros" pseudo macro entry to Macros column when some macros are filtered by the search field.
- Added warning on Cancel All Macros, Cancel This Macro, Retry Loop, etc actions if it is not the last action of a sequence.
- Added OCR Screen and Paste by Name to the Macro Library.
- Fixed Add Variable/Function/Token/Date Field by Name in Value Inspector.

Engine

- Added option to include Macro Groups in the status menu bar. (forum [https://forum.keyboardmaestro.com/t/feature-request-add-optional-customizable-status-menus/19156])
    - Updating icons and title.
    - Updating information in the menu.
    - Executing on selection or automatically when the menu is displayed.
    - Added Group Status Menu trigger.

  * Added support for manipulating Keyboard Maestro Engine windows in the Manipulate a Window action.

- Added Set Next Engine Window Position action.
- Subroutines
    - Added Subroutine trigger which defined parameters to be passed to the macro.
    - Added Execute a Subroutine action to execute a macro, passing it parameters.
    - Added Return from Subroutine action to return a value from subroutine macros.

Triggers

- Added Unlock trigger. (forum [https://forum.keyboardmaestro.com/t/execute-macro-every-time-mac-unlocks/2347])
- Added Appearance Changed trigger. (forum [https://forum.keyboardmaestro.com/t/trigger-request-change-in-dark-mode/21119])
- Added Power Status Changed trigger.
- Added "long press" option for Hot Key and USB Device Key triggers.
- Disallow adding duplicate redundant triggers (like two Power Status Changed triggers).

Actions

- Added Paste by Name action giving Spotlight-like search of clipboard history.
- Added Prompt for Screen Rectangle or Location action.
- Added Try/Catch and Throw actions. (forum [https://forum.keyboardmaestro.com/t/email-notification-upon-macro-failure/18416]) (forum [https://forum.keyboardmaestro.com/t/feature-request-try-catch-finally/5954])
- Added Display Progress action.
- Added Display Progress option to For Each action.
- Added a Pause Until Change action to detect clipboard, modifiers, keyboard, mouse, or application changes.
- Added Create Unique File action.
- Added additional field types to the Prompt for User Input action:
    - Added Date, Time and Date & Time picker fields.
    - Added Slider fields.
    - Added Color Well fields.
- Remove "Instance " and "Local " from variable names in Prompt for User Input form.
- Added icon chooser selection to Set File Icon and Set Clipboard to Image and other actions.
- Added support for selecting multiple files or folders in the Prompt for File action. (forum [https://forum.keyboardmaestro.com/t/prompt-for-multiple-files/15297]).
- Added support for creating the folder when selecting a new folder in the Prompt for File action.

- Added *separated by* option to Substrings In collection. (forum [https://forum.keyboardmaestro.com/t/suggestion-for-new-collection-for-for-each-list-items/23014/4])
- Added option to control text processing of input text for a variety of actions. (forum [https://forum.keyboardmaestro.com/t/bug-regex-replace-that-converts-some-text-to-lower-case-in-error-macro-v9-2/21168])
- Added option to Search and Replace action to replace only the first or last match. (forum [https://forum.keyboardmaestro.com/t/request-add-switch-to-km-search-replace-to-turn-global-matches-on-off/22986])
- Added option to control volume of Speak Text action. (forum [https://forum.keyboardmaestro.com/t/speak-text-volume/18362])
- Added a variety of macro environment variables (eg KMINFO_TriggerValue) to the Execute a Shell Script action.
- Added a calculation option to Switch other actions.
- An Assert action with an empty condition now always asserts.

Filters

- Added filter Sort, Reverse and Shuffle Lines. (forum [https://forum.keyboardmaestro.com/t/help-randomizing-a-list-of-words/3409])
- Added filter Escape for Regular Expression.
- Added filter Quote for JSON.
- Added filter URL components such as scheme, host and path. (forum [https://forum.keyboardmaestro.com/t/extract-domain-from-url-with-regex/15736/8])
- Added filter Encode & Decode Base64.
- Added filter Encode HTML With Numeric Entities.
- Added filter Calculate MD5.
- Updated to the latest of John Gruber & Aristotle Pagaltzis Title Case filter, and internalized it to remove perl dependency.

Tokens

- Added %LastWindowID% token.
- Added %PromptWithListText% token.
- Added %PasteByNameText% token.
- Added %UserHome% token.
- Added %FinderSelection% and %FinderSelections% tokens.
- Added %AccessedVariables% token.

Functions

- Added DOUBLECLICKINTERVAL, KEYREPEATDELAY, KEYREPEATINTERVAL functions.
- Added SCREENINDEX function that returns the index of the specified screen.

Clipboard History

- Added Characters / Words / Lines count to Clipboard History Switcher.
- Adjusted Clipboard History Switcher numbering to match %PastClipboard%.
- Added Command-Control/Option/Shift-numbers to Clipboard History Switcher to Paste or Set optionally plain clipboards.
- Added Option-double-click (or option-Return) in Clipboard History Switcher to set the system clipboard.

Custom HTML Prompt

- Added ProcessAppleScript command to Custom HTML Prompt.
- Added option to turn off title bar on Custom HTML Prompt window. (forum [https://forum.keyboardmaestro.com/t/why-isnt-custom-html-prompt-resizable/4602/8])
- Added option to make Custom HTML Prompt window transparent. (forum [https://forum.keyboardmaestro.com/t/why-isnt-custom-html-prompt-resizable/4602/8])
- Added context sensitive WINDOW function for Custom HTML Prompt window resizing operations.

Debugger

- Added a button in the debugger to edit the macro action. (forum [https://forum.keyboardmaestro.com/t/request-provide-km-macro-debugger-button-to-take-you-to-the-editor-macro-in-use/14111])
- Added display of recently accessed variables in the debugger. (forum [https://forum.keyboardmaestro.com/t/request-provide-dynamic-display-of-macro-variables-in-debugger/13655]) (forum [https://forum.keyboardmaestro.com/t/bug-report-local-instance-variables-show-in-variable-token-list/14241/2])

AppleScript

- Added support for getting and setting the Display in Menu Bar Macro Group settings via AppleScript.
- Added "xml" AppleScript property for macros, macro groups and smart groups. (forum [https://forum.keyboardmaestro.com/t/suggestion-quick-diff-for-macros-inside-km/20307/6])
- Added "group xml" AppleScript property for macro groups.
- Added "last used" AppleScript property for macros. (forum [https://forum.keyboardmaestro.com/t/km-last-execution-date/16166])
- Added "modification date" AppleScript property for macro groups and smart groups.
- Added AppleScript support to the editor for showing preference panes.
- Added support for AppleScript "edit" command, eg "edit macro 1".
- Added support for AppleScript to get/set the editing property of an editor window.
- Added option to Search & Replace to replace only the first or last match.
- Added instance parameter to process tokens and calculate commands.
- Added selectAction command to select a specific action by id.
- Added divider1 and divider2 properties to the editor window.
- Added size property to Macro Groups and Macros.

Minor

- Added SF Symbols to Icon Chooser (11.0+)
- Added Export as Text Service.
- Added Export as Finder Quick Action.
- Show notification when the launching editor quits the engine.
- Excluded com.microsoft.ole.source..* flavors from clipboard for better Microsoft compatibility.
- Adjusted Copy action to note clipboard changes even if the clipboard is not read. (forum [https://forum.keyboardmaestro.com/t/copy-function-timeouts-without-updating-system-clipboard/14959/23])
- Adjusted VoiceOver to speak marked status in palette entries. (forum [https://forum.keyboardmaestro.com/t/midi-control-change-for-blind-users/20031/29])
- Use ENV_PWD to set the current working directory for shell scripts. (forum [https://forum.keyboardmaestro.com/t/node-js-dirname-pointing-to-a-temporary-folder/23570/9])
- Improved the Share to Forum sheet. (forum [https://forum.keyboardmaestro.com/t/suggested-ui-change-for-sharing-uploading-macros-actions-from-km-editor/14909])
- Support longer variable names in Prompt For User Input when they will fit under the icon.
- Support formatting AppelScript and JavaScript conditions.
- Support shift arrow selection in Prompt With List (Multiple Selections). (forum [https://forum.keyboardmaestro.com/t/help-to-create-a-macro-list-delimiters-item-enumeration/20223/8])
- Converted use of WebView to WKWebView.
- Improved appearance of Preferences toolbar in Big Sur.
- Improved display of CALCULATE function when the contents are potentially valid.
- Support Return/Enter as a keystroke for the ↩ character in the Conflict Palette. (forum [https://forum.keyboardmaestro.com/t/space-or-return-to-trigger-the-first-macro-in-a-conflict-panel/23327/8])
- Added Audio Input and Camera entitlements to the Engine (to allow their use via scripts etc).
- Added Restore Excluded Application entry to Applications Palette contextual menu.
- Added Tooltips for Palettes. (forum [https://forum.keyboardmaestro.com/t/feature-request-palette-with-icons-only-show-hints/3739]) (forum [https://forum.keyboardmaestro.com/t/request-tooltips-for-palette-nuttons/16273])
- Added AdditionalWebBrowserBundleIDs preference to add alternative bundle IDs for front browser list.

- Allow Microsoft Edge and Brave Beta as a possible Front Browser Chrome-equivalent.
- Allow Spotlight-like windows to be movable.
- Adjusted the hit boxes for various application and macro popup menus. (forum [https://forum.keyboardmaestro.com/t/moving-actions-that-have-selection-options-within-a-macro/22335])
- Ignore the option key when switching applications in the Applications Switcher if the option key is used in the trigger.
- Remove "Instance " and "Local " from variable names in Prompt for User Input form.
- Report lack of Screen Recording permission when that stops an action from working.
- Better reporting of lack of Screen Reporting errors in Find Image and Click Mouse actions.
- Added control over OCR Image behaviour when the action fails.
- The Macro Group macros in the Global Macro Palette use the Macro Group's custom icon if any.
- Prompt For User Input no longder displays hidden fields in non-edit mode. (forum [https://forum.keyboardmaestro.com/t/plug-ins-hidden-parameters-showing-in-editor/22521])
- Switched to CWWiFiClient for Wireless Network trigger, token and conditions.
- Display some editor alerts within the editor window.
- Display some logged error messages more prominently.
- Display time trigger time in localized format in non-edit mode.
- Avoid logging repeated errors.
- Use `pmset displaysleepnow` to sleep the display on 11.0+, which works on M1 Macs.
- Adjusted Write File to fail rather than trash a folder that will be replaced with a file. (forum [https://forum.keyboardmaestro.com/t/setting-path-as-folder-in-write-to-file-empties-the-folder/23666/19])
- Only report large variables excluded from environment every ten minutes.
- Added appropriate window titles (visible via AppleScript/accessibility) to the various "By Name" windows. (forum [https://forum.keyboardmaestro.com/t/how-to-get-the-position-of-a-keyboard-maestro-user-prompt-window/24011])
- Avoid selecting the macro/group name when deleting macro/groups.
- Added action id to failed action log message. (forum [https://forum.keyboardmaestro.com/t/error-handling-debugging-line-numbers/14133])
- Added a few more default status menu icon options.
- Added additional protections for the Write File action.
- Fixed activation of windows so you can use the Clipboard Switcher to paste in to other Keyboard Maestro Engine windows.
- Fixed a bug enabling the Send button in the Report Bugs form (irony).
- Fixed width of Trigger Macro by Name and other Spotlight-like windows in Big Sur. (forum [https://forum.keyboardmaestro.com/t/display-bug-in-trigger-macro-by-name-layout-in-big-sur/20711/4])
- Fixed Paste XML actions. (forum [https://forum.keyboardmaestro.com/t/what-is-the-status-of-the-paste-xml-action-feature/21815]).
- Fixed an issue with Prompt For User Input and long variable names. (forum [https://forum.keyboardmaestro.com/t/request-stop-shortening-long-variable-names-in-prompt-for-user-input/16271]).
- Fixed an issue with a couple actions not saving changes when a token was selected.
- Fixed an issue with file references to images that included non-ASCII characters.
- Improve accessibility of editor toolbar buttons.
- Fixed Login Window action in Big Sur (CGSession no longer exists).
- Fixed a display issue for popup menus in Plug In Actions. (forum [https://forum.keyboardmaestro.com/t/bug-report-parameter-display-truncated-in-third-party-plug-ins/22694])
- Fixed Trim Image to keep the full resolution and DPI.
- Fixed a potential crash with the Reveal a File action.
- Fixed an issue related to ^ in the Substrings In collection. (forum [https://forum.keyboardmaestro.com/t/issue-with-regex-in-for-each-to-extract-first-string-in-each-line/22724])
- Fixed an issue where renaming a macro via the contextual menu would not scroll the name field in to view.
- Fixed an issue with Search and Replace matching multiple times at the end of the string. (forum [https://forum.keyboardmaestro.com/t/keyboard-maestro-regular-expression-bug-trim-whitespace-from-start-and-end-of-a-string/22963]).
- Fixed an issue where the engine might not be notified of macro changes if the editor is quit immediately after making the change.
- Fixed an issue with the Sleep Screen action in Apple Silicon Macs.

Download

Download Keyboard Maestro [https://www.stairways.com/action/linkthru?download] now to try all these great capabilities. Or keep reading for even more details about the Features of Keyboard Maestro.

Alternatively, you can contact us [mailto:mailto:support@stairways.com] if you have a question about whether Keyboard Maestro can solve your automation needs. We want all our customers to be satisfied, so we are happy to help you understand how Keyboard Maestro can achieve your automation goals.

Features

Keyboard Maestro is a productivity enhancer with several main functions. It allows you to:

- record and design your own macro shortcuts and activate them at any time.
- work with clipboard history using Clipboard History Switcher.
- work with an unlimited number of saved clipboards using Named Clipboard Switcher.
- navigate through running applications with Application Switcher and open windows with Window Switcher.

Macro Groups allow you to organize your macros. Think of them as folders of macros. Each Macro Group controls when the macros it contains are active. A Macro is made of two parts: a set of Triggers you choose to determine when the macro is executed and a list of Actions that define what the macro does when it is executed.

By creating macros, you can customize your Mac to suit your use, streamline tedious tasks, and remove opportunities for mistakes by automating repetitive jobs. Make stubbornly difficult applications behave the way you want them to; press a key and have the computer do the next minute worth of tedious tasks for you; type a few characters and have a page full of boilerplate text appear; and so much more. Soon you'll wonder how you could have used your Mac without Keyboard Maestro.

Here are some of the primary features of Keyboard Maestro.

General

- Use Dark Mode or Light Mode (10.14+).
- Sync your macros across multiple Macs using DropBox or other file sharing systems.
- Trigger macros by key, typed string, menu, by name, remotely, via MIDI, and more.
- Use the included Assistance system to help if you get stuck.
- Use the included Macro Debugger for detailed control of your macros in action.

Macro Groups

- Create Macro Groups, which contain Macros and control when they are active.
- Macro Groups can be displayed in the menu bar with additional information and macros.
- Macro Groups can be restricted to or excluded from specific applications.
- Macro Groups can be restricted to or excluded from specific windows.
- Macro Groups can be activated or deactivated with hot keys, or via the status menu or global macro palette.
- Macro Groups can display a themed palette of contained macros.
- Macro Groups can be enabled or disabled.
- Customize the Macro Group icon by pasting an icon, or using the Icon Chooser and creator.
- If you are syncing your macros, Macro Groups can be disabled on specific Macs.

Triggers

- Macros can be triggered by one or more Macro Triggers using any number of the following:
  - Hot Key trigger – when you press, hold, release or multi-tap a key.
  - Global Macro Palette trigger – with a click on a context sensitive Macro Palette.
  - Status Menu trigger – by selecting from a global system status menu.
  - Typed String trigger – type a string of keys.
  - Application trigger – on launch, quit, activate, deactivate, or periodically while an application is running or active.
  - Audio Output Changed trigger – whenever your audio output device changes.

- Clipboard Changed trigger – when the system clipboard changes.
- Clipboard Filter trigger – select macros from the clipboard switchers.
- Cron trigger – periodically based on time or day or date.
- Dragged File trigger – trigger when a file is dragged onto the macro in a palette icon.
- Display Layout Changed trigger – trigger when the display layout changes.
- Engine Launch trigger – when the Keyboard Maestro engine launches.
- Focused Window trigger – when the front/focused window changes.
- Folder trigger – when a file is added to or removed from a folder.
- Gesture trigger – when you draw a pre-set shape with the mouse or trackpad.
- Idle trigger – when your Mac has been idle for a period of time (but not yet asleep).
- Login trigger – when you log in to your Mac.
- MIDI trigger – when a MIDI note is pressed or released, on controller change, or any packet.
- Mounted Volume trigger – when a volume is mounted or unmounted.
- Periodic trigger – periodically while logged in.
- Public Web trigger – over the Internet, explicitly to the public, or via authenticated log in.
- Remote trigger – when you access a link on our trigger server.
- Sleep trigger – when the system goes to sleep.
- Time of Day trigger – at a particular time of day.
- USB Device Key trigger – when you press, hold, release or multi-tap any HID (Human Interface Device) device key.
- USB Device trigger – when a USB device is attached or detached.
- Wake trigger – when the system wakes from sleep.
- Wireless Network trigger – when your Mac connects or disconnects to/from a wireless network.
- iPhone trigger – from your iPhone, iPod touch or iPad.
- Script trigger – from an AppleScript or other script.
- URL trigger – from an AppleScript or other script.

Actions

- You can create Macro Actions manually or by recording them.
- You can save preconfigured actions as favorites for reuse.
- You can download [https://www.stairways.com/action/linkthru?thirdpartyactions] or write your own Plug In Actions.
- There are many, many actions covering a broad range of facilities, including:
  - Plug In Third Party Actions.
  - Application Control actions to launch, activate, quit, show, hide, etcetera, applications.
  - Clipboard actions to set, read, delete, style, filter, search, replace, etcetera clipboards.
  - Control Flow actions to pause, loop, test conditions, process sets of items, and more.
  - Debugger actions to breakpoint, pause, step over, step into, etcetera, while debugging macros.
  - Execute AppleScripts, shell scripts, JavaScript (for Automation or in Safari/Chrome) and more.
  - Read, write, trash, duplicate, append files, or get or set information about them.
  - Safari/Chrome actions to work with the front web page, filling forms.
  - Image actions to find, capture, read, write, crop, annotate, or display images.
  - Interface actions to move or resize windows, click the mouse, type keystrokes, press buttons, etcetera.
  - iTunes actions to play tracks, fast forward, rewind, change the volume and more.
  - Actions to control Keyboard Maestro itself, enabling or disabling macros, showing and hiding palettes, etcetera.
  - MIDI actions to send notes, control changes, or arbitrary packets.
  - Send notifications via email, SMS, iMessage, notification center, sounds, alerts, etcetera.
  - Open files, folders, URLs, system preferences, even 1Password bookmarks.
  - QuickTime player actions to play movies, step forward or backward and more.
  - Actions to control a Stream Deck device.
  - Actions to launch applications, select windows, or switch clipboards.
  - Sleep, restart or shut down your Mac, Fast User Switch, log out, change brightness or volume, etcetera.
  - Type or paste strings, speak text, work with variables or dictionaries.
  - Perform calculations, prompt for information, read or write passwords from the Keychain and more.
  - Web actions to search the web, download files, remotely trigger actions, etcetera.

Basically, if you can do it yourself, Keyboard Maestro can probably do it for you.

Editor

- Dark Mode support.
- Multiple editor windows.
- Interactive Help system.
- Large Text support.
- Basic Touch Bar support.
- Smart Groups.
- Select from recently triggered or edited macros.
- Autocompletion.
- Integrated access to wiki help.
- Rename, color, group, and add notes to actions.
- Palette Theme Editor.

Named Clipboard and Clipboard History Switcher

- Never lose your clipboard again.
- Browse your past clipboards and paste any previous clipboard item.
- An unlimited number of Named Clipboards.
- Send clipboard entries to other Macs.
- Clipboards display rich text and images.
- Use Quick Look to view clipboard entries.
- Apple Clipboard Filter triggered macros directly to selected clipboard entries.
- Set clipboard entries as favorites so they are always available in your clipboard history.
- Copy, Cut or Paste to/from Named Clipboards using a single keystroke.
- Clipboard History is optionally saved across logins and restarts.
- Named Clipboards are saved permanently.

Application and Window Launching and Switching

- Display a Cover Flow view of available applications for quick launching.
- Optionally replace the system Command-Tab application switcher.
- Customize the switcher to match the look you want.
- Switch to any application or window with a keystroke.
- Switch to an application and hide all others.
- Easily select the exact application or window you want.
- Launch, hide, quit or force quit any application.
- Close or minimize any window.
- Quit (or force quit) and relaunch applications.
- Get Info or reveal applications.
- Choose the application ordering you want: alphabetically, by last use, or by launch order.
- Sort windows alphabetically or by window depth order.
- Optionally hide other applications.
- Optionally always hide other applications.

Purchase

Keyboard Maestro is engineered by Stairways Software Pty Ltd and distributed by FastSpring [https://www.stairways.com/action/linkthru?fastspring]. Keyboard Maestro is licensed on a per-user basis and individual users may use it on up to five Macs.

New customers can purchase Keyboard Maestro for US$36 by choosing the Keyboard Maestro ➤ Purchase Keyboard Maestro menu. Customers with five or more users should contact us [mailto:mailto:support@stairways.com] for a volume discount quote.

Keyboard Maestro 10 is a paid upgrade for most users of previous versions. Existing users are eligible for a discount and can purchase an upgrade by choosing the Keyboard Maestro ➤ Purchase Keyboard Maestro Upgrade menu.

Customers who purchased Keyboard Maestro after 1 March 2021 have been issued a free upgrade to Keyboard Maestro 10. If you have not received your free license, you can find your free license upgrade at https://enquiry.stairways.com/ [https://enquiry.stairways.com/].

Customers who purchased Keyboard Maestro 9 prior to March 2021 can upgrade to Keyboard Maestro 10 for US$18 until 15 December 2021. After that date, or customers who purchased older versions of Keyboard Maestro can upgrade to Keyboard Maestro 10 for US$25. If you have not received your instructions on how to upgrade, you can find details by looking up your Keyboard Maestro license at https://enquiry.stairways.com/ [https://enquiry.stairways.com/].

Customers who have not disabled upgrade emails have been emailed with new license or upgrade instructions as appropriate. If you have not received this email, please contact us [mailto:mailto:support@stairways.com] so we can resolve this promptly.

It is our informal policy to have a paid major upgrade roughly once every 12 to 24 months. This allows us to have a reasonably consistent revenue stream with which to fund development of Keyboard Maestro, and ensures we are working as much for existing customers as to expand the customer base. It means you can keep what you have purchased and not have to pay a subscription, but also that we will still be around working on improving Keyboard Maestro and adding features for when you choose to upgrade.

A fully-functional trial version of Keyboard Maestro is available for download from https://download.stairways.com/ [https://download.stairways.com/].

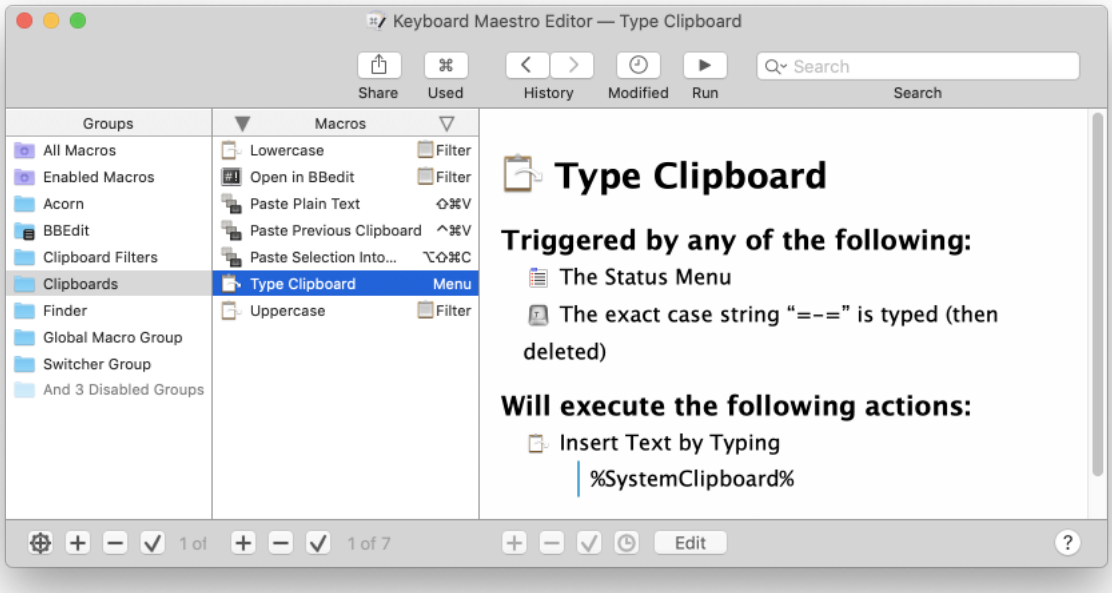
For sales enquires, customer service, technical support, or to contact project management, our current contact information is listed at https://contact.stairways.com/ [https://contact.stairways.com/].

For more information about anything to do with Keyboard Maestro, visit https://www.keyboardmaestro.com/ [https://www.keyboardmaestro.com/].

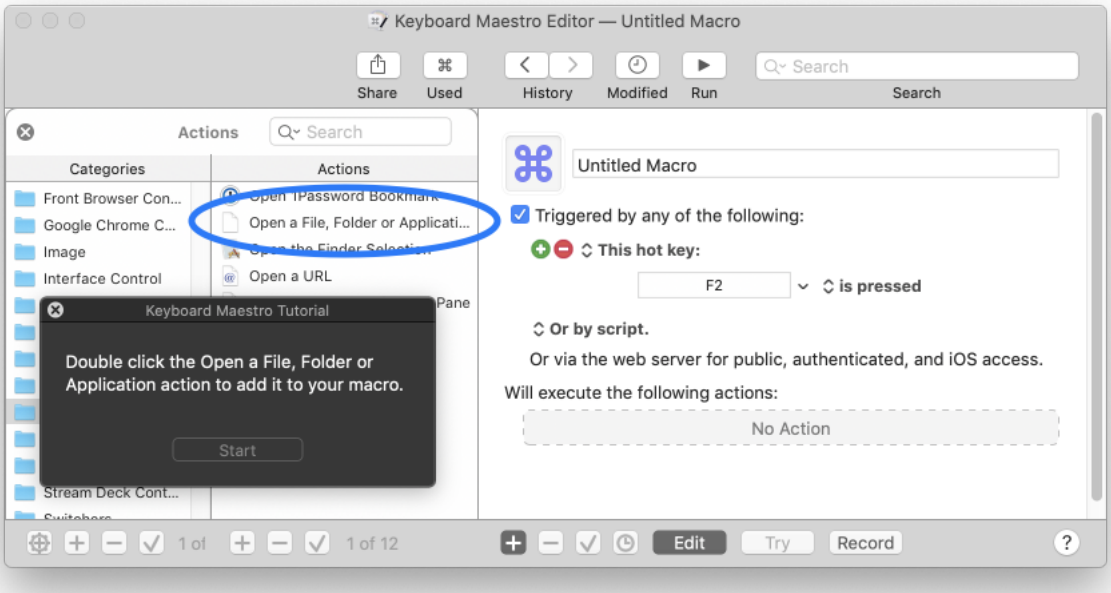
Screenshots

Here is a quick taste of what Keyboard Maestro offers.

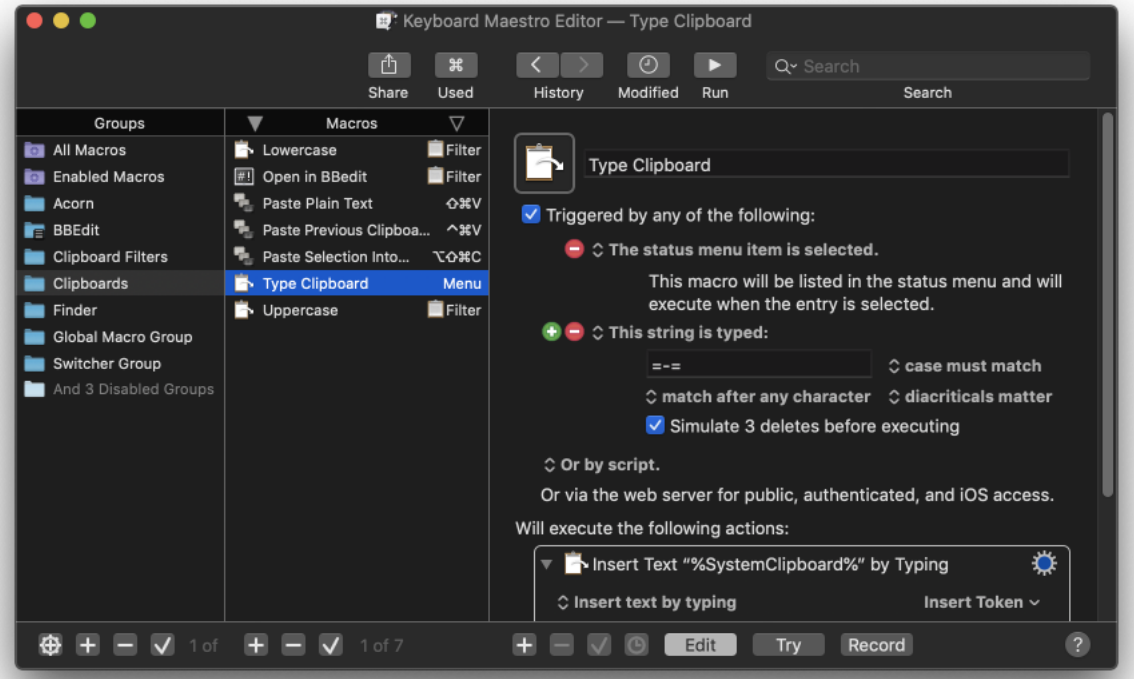
Macros Window



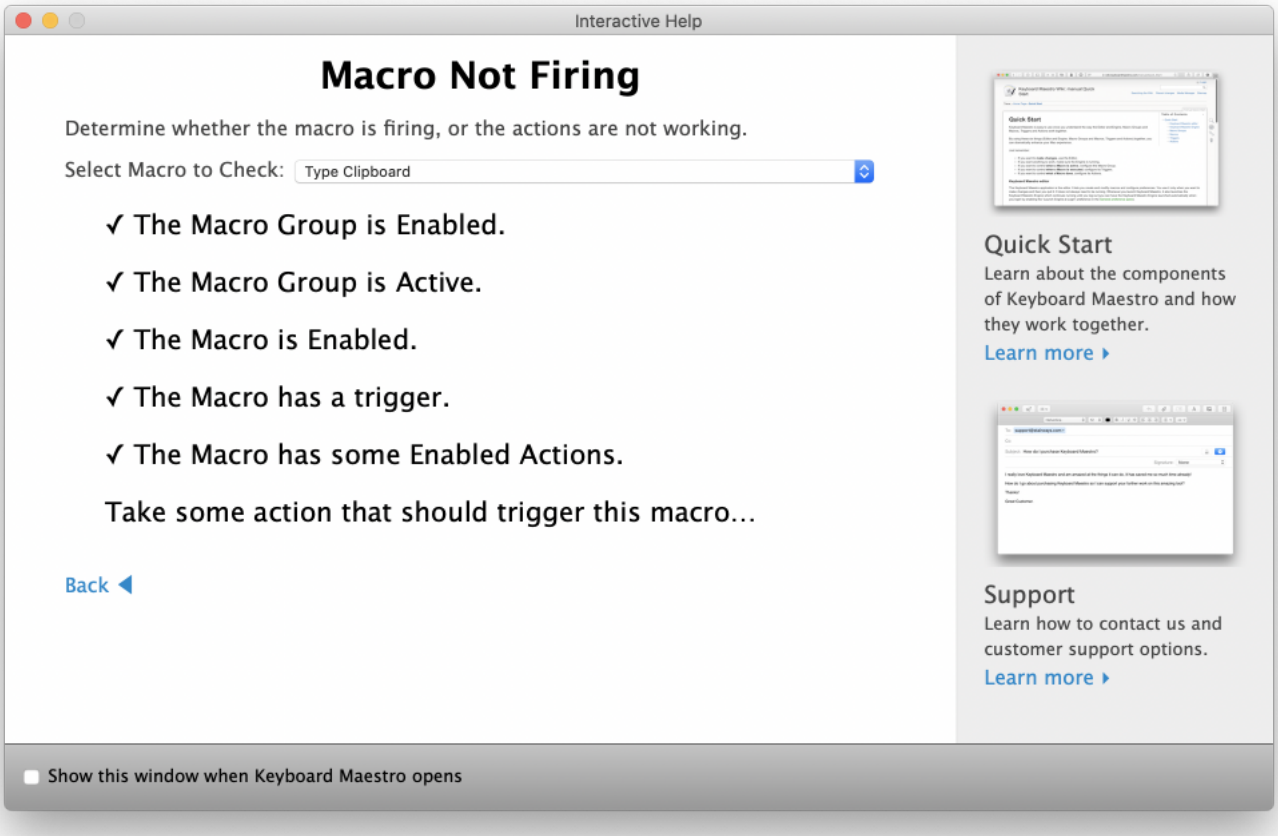
Macro Editor



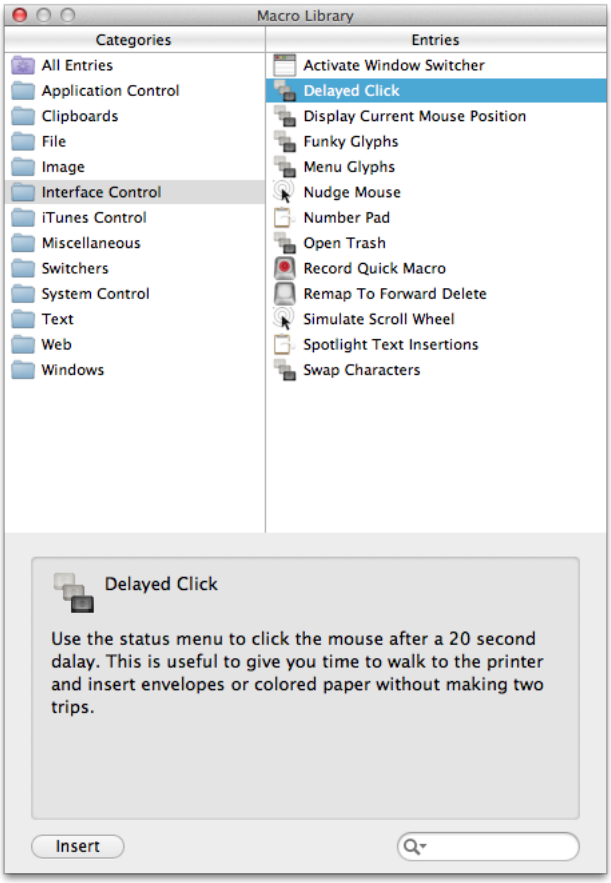
Dark Mode



Interactive Help

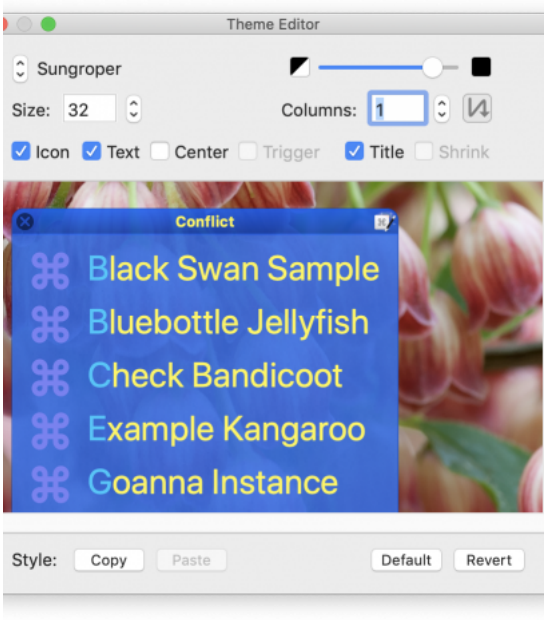


Macro Library

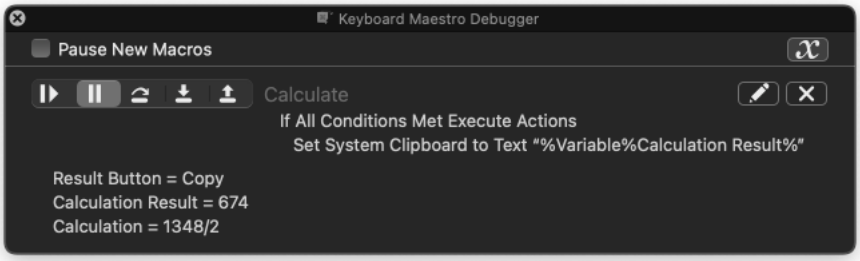


Icon Chooser

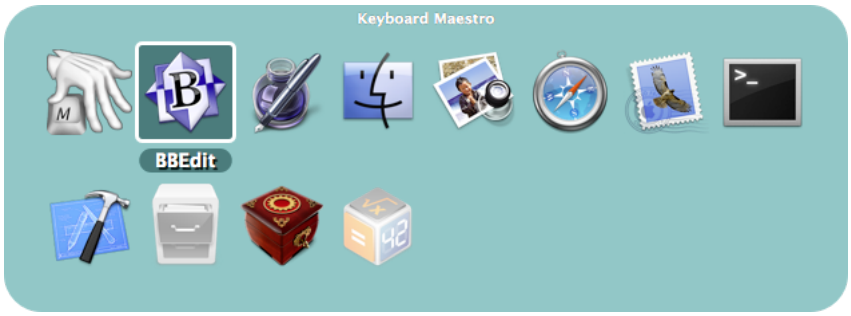
Palette Theme Editor



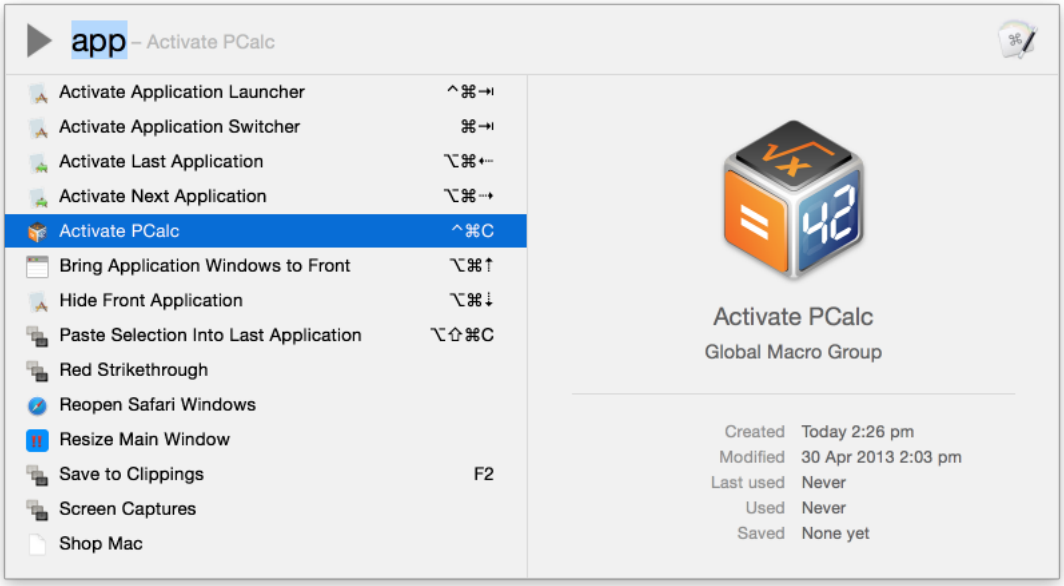
Macro Debugger



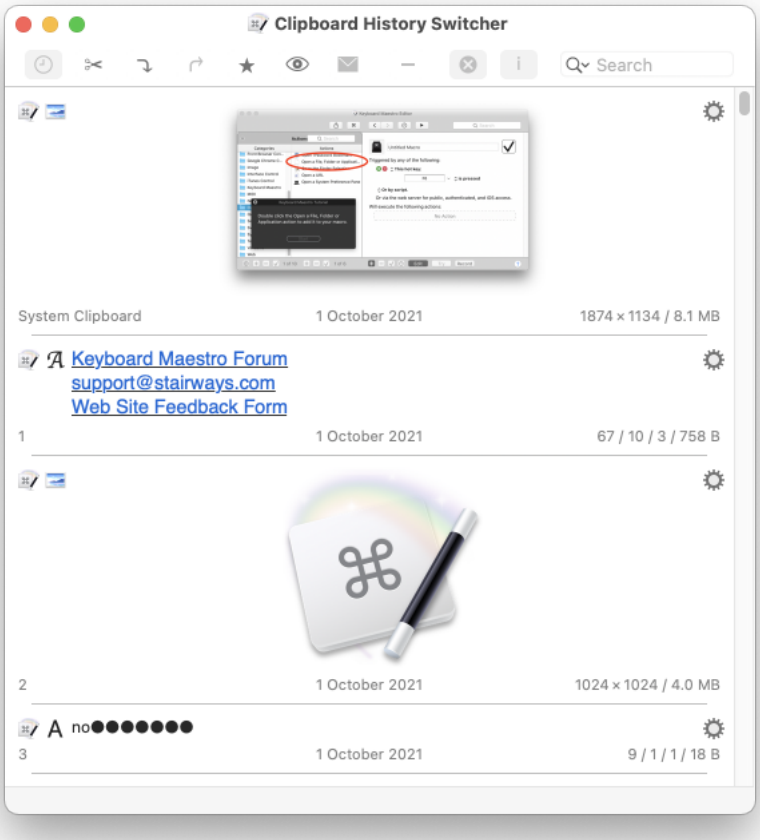
Application Switcher Window



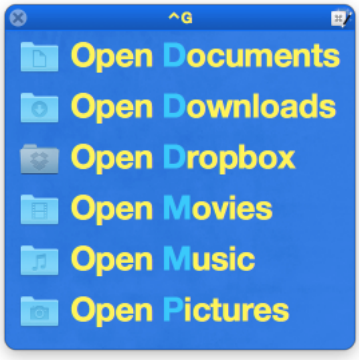
Trigger Macro by Name



Clipboard History Switcher Window



Conflict Palette



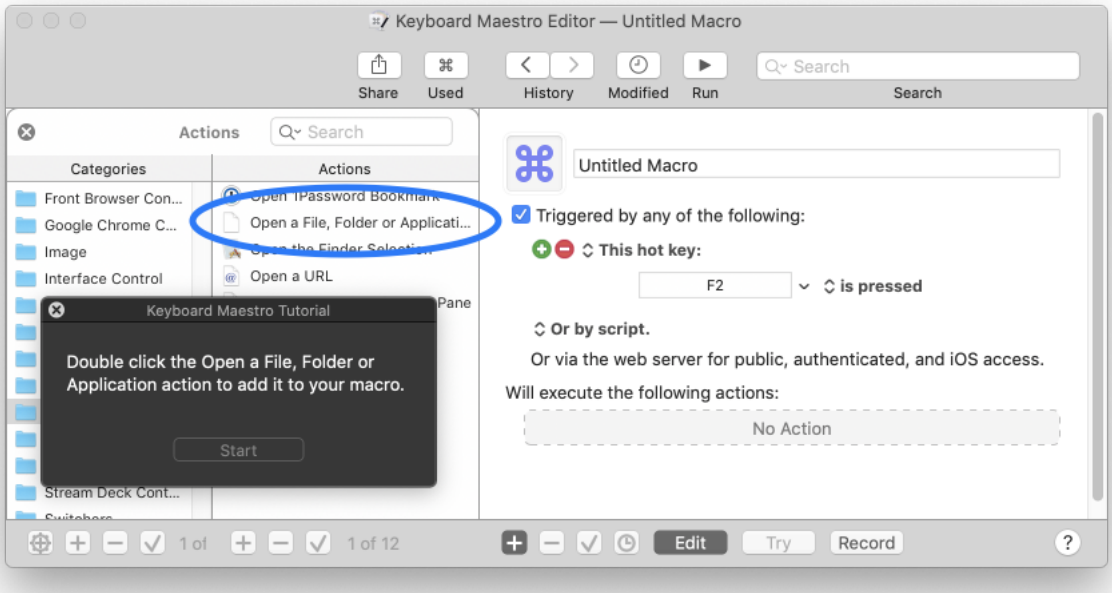
Tour

This demonstration will begin to show you the power and versatility of Keyboard Maestro.

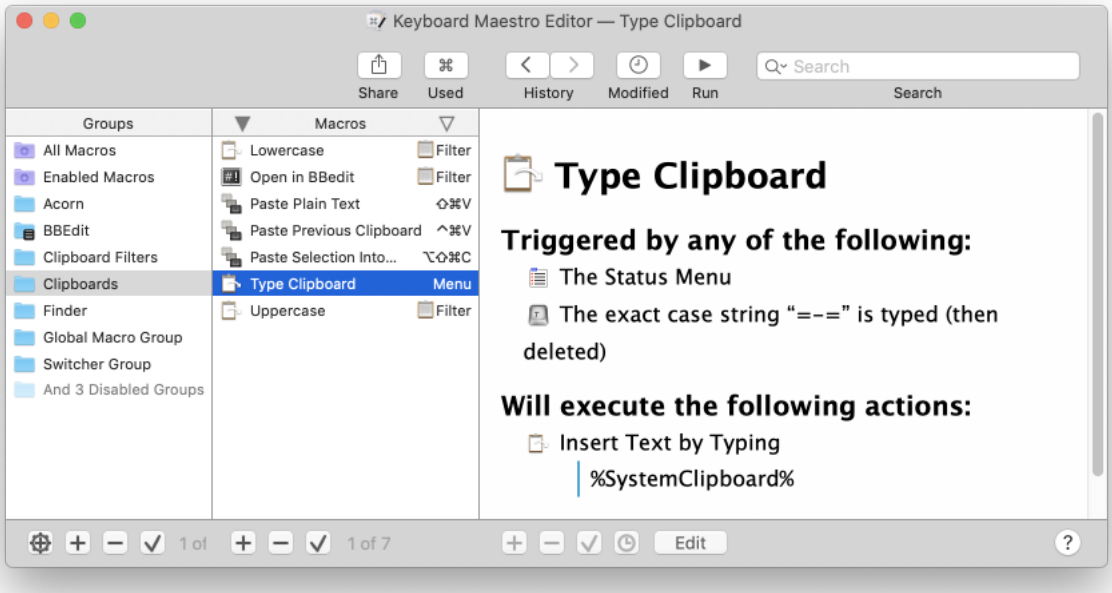
Getting Started

To start, launch Keyboard Maestro. It will initially display the Welcome window.

If you are new to Keyboard Maestro, start the tutorial and Keyboard Maestro will show you how easy it is to create a macro.



Close the Welcome window to display the Macros window.



You can see some example Macros we have included for you.

Make New Macro

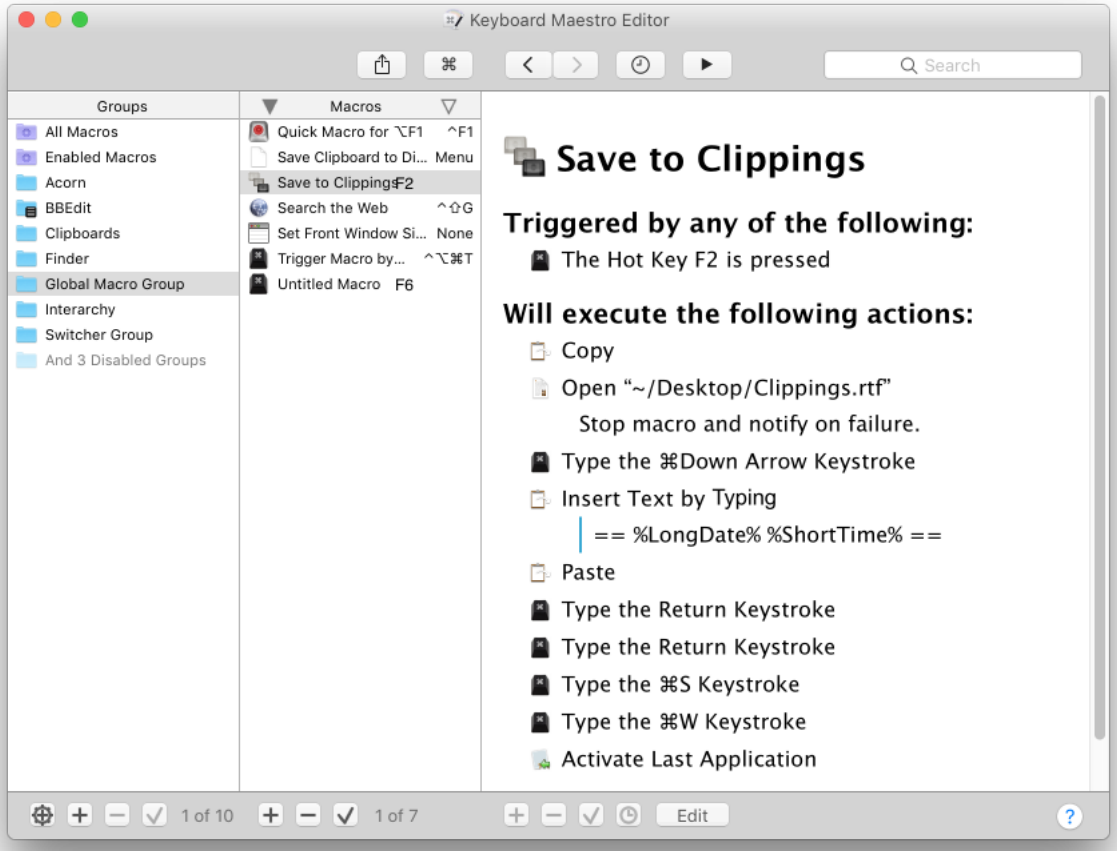
Click the + button under the Macro column to add a macro and display the Macro Editor window. We will now design a complex Macro enabling you to save clippings to a text file. First, launch TextEdit and create a new empty document. Save this blank document as Clippings.rtf in your Desktop folder.

Normally, to add a selection to the Clippings.rtf file, you would have to do all this:

- Press Command-C to copy the selection in an application.
- Go to the Finder, open your Desktop folder, then open the Clippings.rtf file.
- Go to the end of the file in TextEdit.
- Press the Return key and type a line of dashes and return to separate the clippings.
- Press Command-V to paste the selection you made before.
- Save the file and close the document.
- Switch back to the application where you originally selected some text or a picture.

That is all very tedious, and probably explains why most people never even bother with such an operation.

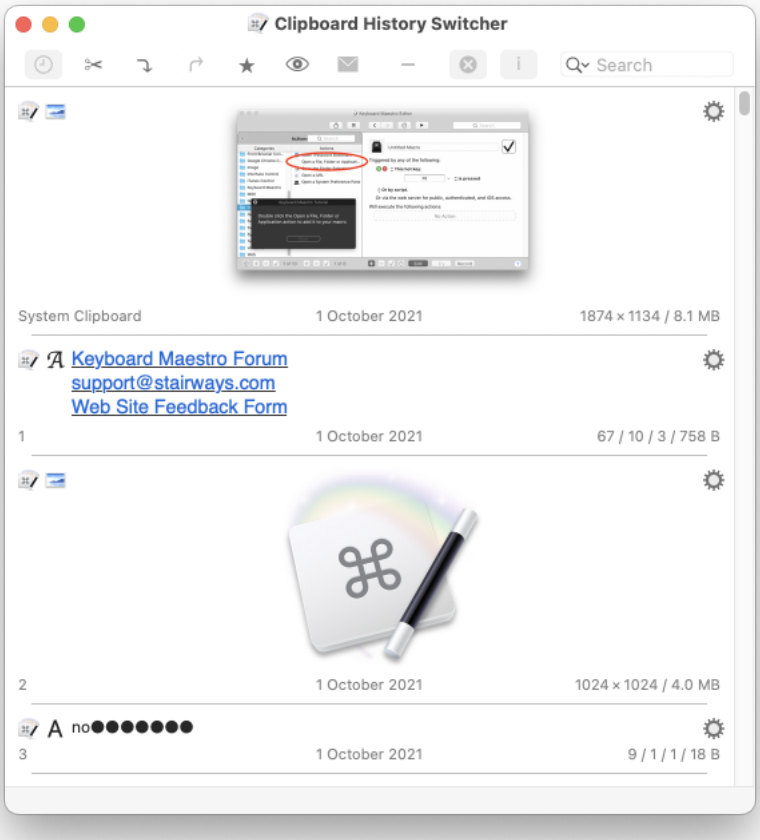
So let's define the whole sequence as a Macro.



Now any time you have some text you want to save, just select it and press F2 (or Fn-F2 depending on the state of the Use all F1, F2, etc. keys preference in the System Preferences). What used to be too much hassle to bother with is now done in seconds!

Clipboard History

Keyboard Maestro automatically remembers your clipboard history, saving a copy of each new clipboard item as you copy it. You can then paste any previous clipboard using the defined Hot Key (by default, Command-Control-Shift-V).

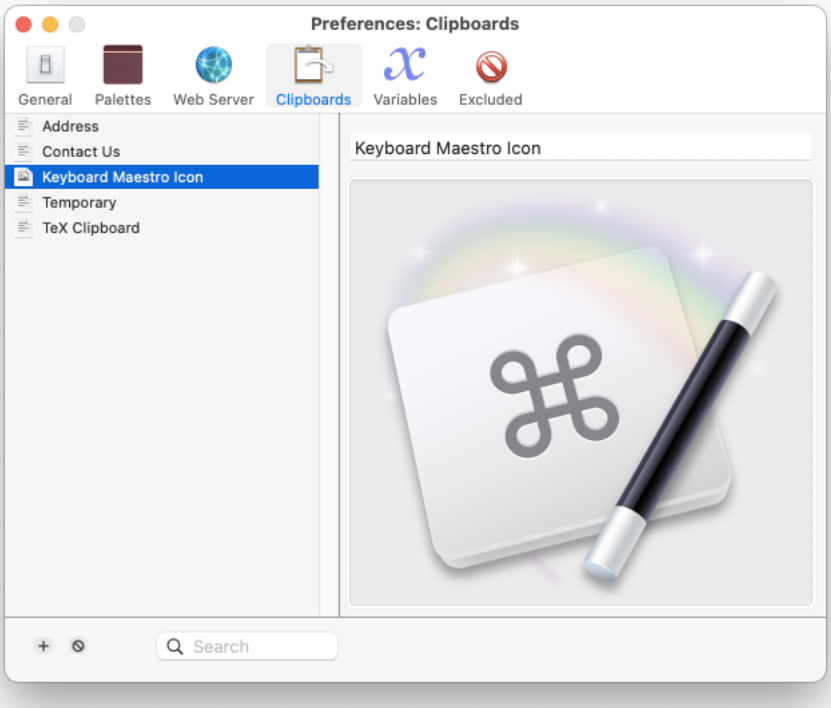


Another advantage of the Clipboard History is you can leave the window open (by toggling the Close After Action (x) button) and paste previous clipboards by simply double clicking the desired item. This is particularly helpful when you need to copy many items in many places to be pasted into one application. Hold down the shift key to paste without styles.

Keyboard Maestro can even save your clipboard history across restarts and log outs. Just enable the "Save Clipboard History to Disk" preference in the General preference pane.

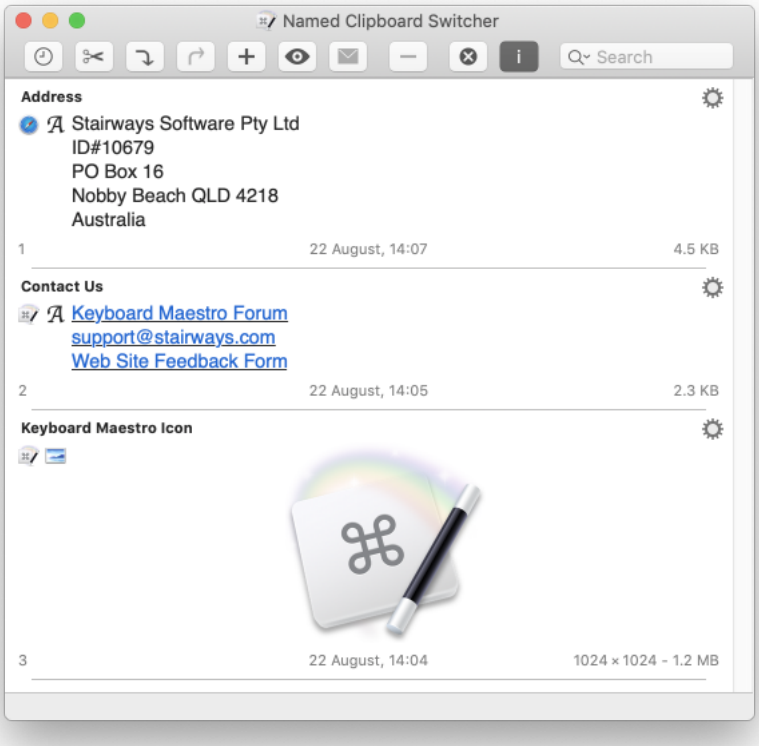
Named Clipboards

For more permanent information, Keyboard Maestro lets you create as many named clipboards as you want using the Clipboards preference pane.



Named clipboards let you save frequently used information, like your company logo, timely information like a customer's address, or your address so you never have to type it again. This allows you to paste the saved information whenever you want, wherever you want.

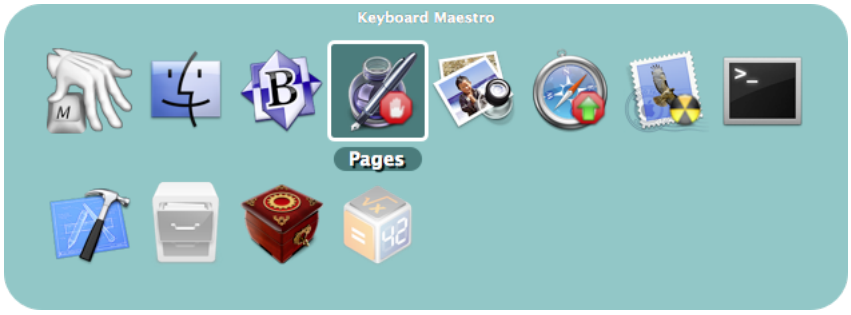
Then you can cut, copy or paste to/from the named clipboards using the defined Hot Keys (by default, Command-Shift-X, C and V respectively, but you can change them, too).



If you have a large screen, or a specific job that needs it, you can have the window stay open (by toggling the Close After Action (x) button) and copy or paste named clipboards easily.

Application Switcher

At the end of the Macro we defined, we used the Switch To Last Application action to switch back to the application you were using before. But most of us use a lot more than one application and we need to be able to switch between them, whether or not the application is already open. You can do more than just switch between applications with Keyboard Maestro: it lets you define applications that always appear when you are switching so you can launch them quickly, and also Excluded Applications that you never want to see. As well, you can quit, hide or show applications. The Application Switcher is activated like any other macro, so you can find it in the Switcher Group in the Macros window. By triggering the Application Switcher macro (for example by entering the default Command-Tab key trigger, though this can be changed) you can display the Application Switcher window.

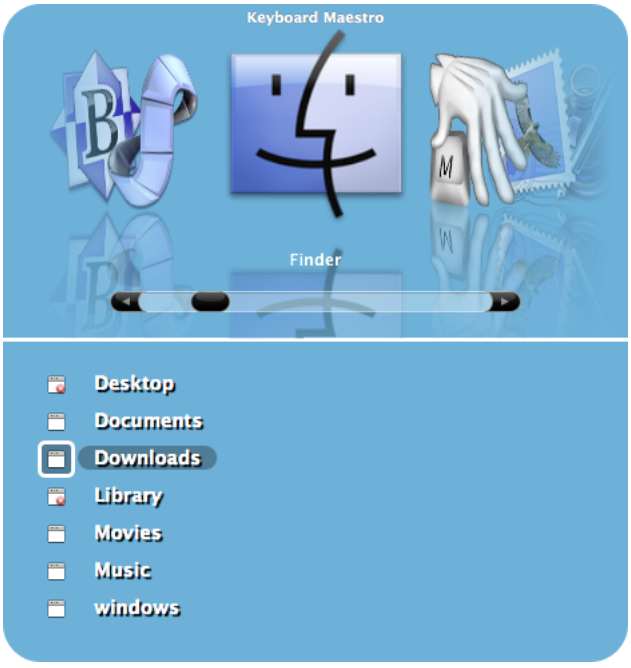


Note the applications set to be launched or quit.

If you want a specific keystroke to always launch a certain application, you can define a Macro to do that. For example, you could create a Macro with a Hot Key Trigger of F3 so it always activates Safari.

Window Switcher

Keyboard Maestro also includes a Window Switcher. By triggering the Window Switcher macro (for example, by pressing Control-Tab), the Window Switcher window appears showing all the windows of current application (in the current space).



Note the windows set to be closed. This is a very useful way of cleaning up an application with a lot of open windows.

Conduct Your Mac Like a Pro!

This is just a taste of all that Keyboard Maestro can do for you. It's time you started getting the most from your Mac! Download Keyboard Maestro [https://www.stairways.com/action/linkthru?download] today and you can be working faster and smarter in no time.

Links

Download Keyboard Maestro from https://download.stairways.com/ [https://download.stairways.com/].

Purchase Keyboard Maestro at https://purchase.stairways.com/ [https://purchase.stairways.com/].

Look up your current or previous license status and serial numbers, and get information about discounted upgrades from https://enquiry.stairways.com/ [https://enquiry.stairways.com/].

Join the Keyboard Maestro Forum [https://forum.keyboardmaestro.com/] online community consisting of the developers and Keyboard Maestro users at https://forum.keyboardmaestro.com/ [https://forum.keyboardmaestro.com/].

Documentation describing Keyboard Maestro is available at https://documentation.keyboardmaestro.com/ [https://documentation.keyboardmaestro.com/].

A wiki containing additional information, macros and other resources for Keyboard Maestro is available at https://wiki.keyboardmaestro.com/.

For sales enquires, customer service, technical support, or to contact project management, our current contact information is listed at https://contact.stairways.com/ [https://contact.stairways.com/].

For more information about anything to do with Keyboard Maestro visit https://www.keyboardmaestro.com/ [https://www.keyboardmaestro.com/].

Quick Start

Keyboard Maestro is easy to use once you understand the way the Editor and Engine, Macro Groups and Macros, Triggers and Actions work together.

By using these six components together, you can dramatically enhance your Mac experience.

Just remember:

- If you want to **make changes**, use the Editor.
- If you want anything to work, make sure the Engine is running.
- If you want to control **when a Macro is active**, configure the Macro Group.
- If you want to control **when a Macro is executed**, configure its Triggers.
- If you want to control **what a Macro does**, configure its Actions.

Keyboard Maestro editor

The Keyboard Maestro application is the editor. It lets you create and modify macros and configure preferences. You use it only when you want to make changes and then you quit it. It does not always need to be running. Whenever you launch Keyboard Maestro, it also launches the Keyboard Maestro Engine which continues running until you log out (you can, and should, have the Keyboard Maestro Engine launched automatically when you login by enabling the "Launch Engine at Login" preference in the General preference pane).

Keyboard Maestro Engine

The Keyboard Maestro Engine is a background only application that enables all of Keyboard Maestro's features. It responds to your Hot Key presses, watches the time, tracks applications and maintains your clipboard history, handles receiving clipboards, displays palettes, and, of course, executes your Macros. It should be running at all times, so we recommend you enable the "Launch Engine at Login" preference in the General preference pane.

Macro Groups

Keyboard Maestro allow you to organize your Macros into Macro Groups which are like folders of macros. Each Macro Group controls when the macros it contains are active. A Macro Group can target or exclude specific applications, which means the macros it contains will only be active in those desired applications. For example, you can have macros which are active only in Mail.app, or only in Mail.app when a compose window is open.

A Macro Group can also act as a container for specific-use macros which are enabled only when you specifically activate them. For example, you could create a Macro Group containing macros that resized or repositioned windows using the arrow keys, but those macros would only be active after a specific Hot Key was pressed so that the arrow keys could be used normally at other times.

Macro Groups can be displayed as palettes, allowing you to create your own custom toolbars which can be configured with a variety of themes.

Macro Groups can be displayed in the menu bar, allowing you to show updating information in the menu bar, and quickly access the macros it contains.

You create a Macro Group by clicking the + button at the bottom of the Macro Groups list. You can disable or enable Macro Groups by clicking the ✓ button. You can configure a Macro Group by selecting it and clicking the Edit button, or by double-clicking on it.

You can also create Smart Groups, which are essentially saved searches and will show you any macros that match any of the set of search criteria.

Macros

Keyboard Maestro's main purpose is to execute Macros. A Macro lives in a Macro Group and consists of a set of Triggers that determine when the macro is executed, together with a list of Actions that define what the macro does when it is executed. You create a Macro by clicking the + button at the bottom of the Macros list. You can disable or enable Macros by clicking the ✓ button. Keep in mind that a Macro can only be active when the Macro Group that contains it is active. You can edit a Macro by selecting it and clicking the Edit button, or by double-clicking on it.

Triggers

A Trigger defines when a macro will be executed. There are a variety of Triggers available, the most common is the Hot Key trigger which executes the macro when a specified Hot Key is pressed. Similarly, you can use a Typed String trigger to execute a macro when you type some text (for example =addr=). Other common triggers are the Macro Palette which lets you trigger a macro by clicking on a context (front application or window) sensitive floating palette of macros and the Status Menu trigger which displays the macro in the Status Menu. You can trigger macros from the clipboard history (to apply to the clipboard entry), or by drawing a shape with the mouse or trackpad.

You can also trigger a macro when you login or when your Mac sleeps or wakes or goes idle, at a specific time or on a specific day, when an application launches, activates or quits, by executing a script, or remotely using a web browser or iPhone or our trigger server. And you can trigger a macro when something changes, like a volume being mounted or unmounted, plugging in headphones, connecting a monitor, a USB device, or to a wireless network.

A Trigger will only execute the macro if the Macro Group and Macro are enabled and currently active. You create Triggers by creating or editing a Macro and clicking the green ⊕ button near the top of the macro detail view.

Actions

When a Macro is Triggered it executes a list of Actions. Keyboard Maestro performs each of the Actions in order. There are a wide variety of Actions allowing you to control applications, simulate user interface events like key presses, mouse clicks and menu selections, work with files or images, control your Mac or the clipboard, or display a variety of powerful switchers (Application, Window, Clipboard and Clipboard History Switchers). You can also execute a script (AppleScript, Java, Swift, shell Script or Automator Workflow) or even download or create your own custom plug in actions. There are many, many more actions, so if you can do it yourself, Keyboard Maestro can probably do it for you. You create Actions by creating or editing a

Macro and clicking the + button to display available actions, or by clicking on the Record button and performing the action while Keyboard Maestro records your actions to your Macro.

Assistance

If you have any problems writing macros, you can get help from a number of places.

- This User Manual describes the basics of using the Keyboard Maestro editor.
- The Tutorial will show you how to create a simple macros. You can start the Tutorial by choosing the Help ➤ Tutorial menu.
- The Quick Start describes the components of Keyboard Maestro and how they work together, and it is important to understand these basics to get the most out of Keyboard Maestro.
- The Wiki includes lots of information on all aspects of Keyboard Maestro.
- The Forum [https://forum.keyboardmaestro.com/] is a great place to ask for help with macros.
- Each action in Keyboard Maestro includes a Help option in the ⚙ menu at the top right of the action.
- Holding down the option key while choosing from any of the Insert menus in the Edit menu will get you help on actions, functions, or tokens.
- Holding down the option key while choosing from the new trigger menu will get you help on the triggers.
- Holding down the option key while choosing from the new condition menu will get you help on the conditions.
- Holding down the option key while choosing from the new collection menu will get you help on the collections.

And last but not least, you can ask Keyboard Maestro itself for assistance by choosing Help ➤ Interactive Help menu.

In particular, the Interactive Help window can help you:

- if something is happening that you do not expect — Keyboard Maestro can show you what it is doing.
- if a macro is not doing anything — Keyboard Maestro can help you to learn why.

How do I ...?

How do I install Keyboard Maestro?

To install Keyboard Maestro, download it [https://download.keyboardmaestro.com/] and then use the Finder to move it to your Mac's Applications folder (or anywhere you like).

Note: You must use the Finder to do this since the Finder will turn off Apple's App Translocation security feature when you move the application.

When you launch Keyboard Maestro it launches an invisible "Keyboard Maestro Engine" that continues to run even after you quit Keyboard Maestro. The engine is the process that enables your Macros, Application Switcher, and Clipboard History Switcher to work. This means that they will continue to work after you quit Keyboard Maestro, as long as the engine is still running.

You can quit or launch the engine manually using the File menu.

You should consider turning on the "Launch Engine at Login" preference in the General preference pane to ensure all of Keyboard Maestro's facilities are available to you as soon as you login or startup your Mac.

How do I upgrade Keyboard Maestro?

Keyboard Maestro includes an automatic upgrade mechanism, so to upgrade Keyboard Maestro simply click the Install Update button when prompted.

To upgrade Keyboard Maestro manually, quit the engine by choosing the File ➤ Quit Engine menu, and quit the editor, and replace the Keyboard Maestro application in your Applications folder with the new one. Finally, launch Keyboard Maestro to restart the engine.

Keyboard Maestro will automatically import your previous version macros, clipboards and preferences. Your old macros will be saved in the ~/Library/Preferences/Keyboard Maestro/Keyboard Maestro Macros Saved Version 9.plist in case you wish to revert to version 9 for any reason.

If you are upgrading directly from a much older version, you will get better results by upgrading to the last of each major version in turn, ie, run 2.1.3, then 3.5, then 4.3.2, then 5.3.2, then 6.4.8, then 7.3.1, then 8.4.2, 9.2, and then the current version. You can download old versions from our file archive [https://files.stairways.com/].

How do I purchase Keyboard Maestro?

New customers can purchase Keyboard Maestro for US$36 by choosing the Keyboard Maestro ➤ Purchase Keyboard Maestro menu. Customers with five or more users should contact us for a volume discount quote.

You can look up your current or previous license status and serial numbers, and get information about discounted upgrades from https://enquiry.stairways.com/ [https://enquiry.stairways.com/].

Thanks for supporting us and enabling us to continue work on Keyboard Maestro.

See also the Purchase section.

Can I purchase Keyboard Maestro from the Mac App Store?

Keyboard Maestro will not be available on the Mac App Store. Apple requires applications on the Mac App Store to be Sandboxed, and workflow applications like Keyboard Maestro cannot be sandboxed so it is excluded from the Mac App Store.

See also the Purchase section.

How do I register Keyboard Maestro?

When you purchase Keyboard Maestro you will be given a serial number, and will also promptly be emailed your username (email address) and serial number in the "Thanks For Your Purchase" email. Although you can retrieve this information from us at any time in the future, it is a good idea to keep this safe.

If you do not receive your serial number promptly after purchasing, it may be that the email has not reached you, possibly due to spam filtering on your email service. In this case, try looking up your purchase at https://enquiry.stairways.com/ [https://enquiry.stairways.com/] (although that will email you your serial number which might again be lost to over-zealous spam filters).

Once you have your username (email address) and serial number, launch Keyboard Maestro and either immediately click the Use Existing License button or choose Keyboard Maestro ➤ Register Keyboard Maestro menu and then enter the username (email address) exactly as shown and the serial number exactly as shown and click the OK button. If you have any problems, recheck that the email address and serial number you are entering are exactly as shown (the serial number's email address does not change even if you have changed your email address with us) and also that your license matches the major version number (eg, a version 10 license will work with version 10.x of Keyboard Maestro). If you are already registered, the Register Keyboard Maestro menu will show you the registration details in the About Keyboard Maestro window — if it says it is registered to you, you are all set.

How do I get started?

The first thing to do is to read the Quick Start and do the tutorial by choosing the Help ➤ Tutorial menu. You may also want to subscribe to our Getting Started emails (Keyboard Maestro will ask you to subscribe or you can do that at your customer database page [https://www.stairways.com/form/manageemail] on our web site).

The best way to start using Keyboard Maestro is to start simple and then grow your macros organically.

You may also like to joint the forum (https://forum.keyboardmaestro.com/ [https://forum.keyboardmaestro.com/]) and ask for help with any macros you get stuck on. The forum is a fantastic resource, and if you are having troubles building a specific

macro, the forum is the first place you should go to ask for help since we frequently cannot help with macros for specific applications that we probably don't have access to.

If you would really like to extend your use of Keyboard Maestro, consider enrolling in David Spark's Keyboard Maestro Field Guide [https://learn.macsparky.com/p/km].

Then use your Mac normally and keep an eye out for things you do repetitively. Things like:

- launch or switch to a particular application.
- open a particular document.
- type a specific string of text (eg your name, address, etc).

When you notice something, think about making a Macro to do it and assigning it to a Hot Key or a Macro Palette or Status Menu trigger.

Try to be consistent with your Hot Keys, for example you might have a set of applications you open, using a function key for each, and a set of documents you open, using a Control-Function Key combination for each of them, and a bunch of snippets of text your type, using Control-Letter for each of them (the letter could be a mnemonic to help you remember which one is which). Consider putting a sticker on your keyboard across the top of your function keys to help you remember which ones do which function. Remember that you can use the number pad keys as well.

Also, keep in mind common command keys and system defined hot keys and try to avoid conflicting with them.

See also the Tips section.

How do I create a new Macro?

If you have not done so, use the tutorial by choosing the Help ➤ Tutorial menu to lead you through the process of creating a simple macro.

To create a Macro, launch Keyboard Maestro and select the Global Macro Group, then click the + button under the Macros list. Give the Macro a name, add one or more triggers, and one or more actions. The Macro is immediately active — you do not need to stop editing, quit the editor or anything else.

An easy way to generate macro actions is to turn on recording and proceed to show Keyboard Maestro what you want to do. Then turn recording back off and look through the actions — chances are you will want to delete or adjust some of the recorded actions to make a robust macro, but this will give you a quick start on creating the macro.

How do I find a Macro I've used or modified recently?

You can sort the macros by name, by trigger, by date created, modified or used. So if you launch Keyboard Maestro, select the All Macros smart group, and then choose the View ➤ Sort Macros by Date Modified menu (or choose the View ➤ Sort Macros by Date Used menu) to sort the recently modified (or used) macros to the top.

You can also click the ◷ button above the macro editor pane to select from recently modified macros, or use the ⌘ button to select from recently used macros.

How do I cancel a running Macro?

You can cancel all running macros by holding all the modifiers (Control, Option, Shift and optionally Command) and clicking on the Keyboard Maestro Status Menu Icon.

You can cancel a specific macro by choosing from the Status Menu ➤ Cancel menu and selecting the macro. This is also a useful way of seeing what macros are currently running, if any.

You can also see and cancel macros by choosing the Status Menu ➤ Start Debugging menu.

And finally, you can use the Cancel All Macros, Cancel Other Macros, Cancel This Macro, or Cancel Just This Macro actions.

In desperation, you can quit the Keyboard Maestro Engine from the status menu, and you can launch the editor while holding all the modifiers down to quit all running engines and not launch the engine.

How can I get the mouse coordinates on the screen or in a window?

If possible, you should avoid using mouse click actions. They tend to be very fragile, easily broken by subtle changes to the system or applications. And they also require the screen to be in the expected state, so you usually need to add a Pause action before them to ensure the item they are clicking on is where it is expected to be.

That said, you can use the Mouse Display window by choosing the Window ➤ Mouse Display menu which will let you see the mouse coordinates relative to the window or main screen. Click the lock button to lock the display after a few seconds, and then put your mouse over the desired location. You can then change the relative corner, and click the Clipboard buttons to copy the coordinates.

Alternatively, you can simply record the click. Turn on recording in Keyboard Maestro, go to where you want to be, wait a couple seconds for the screen to be stabilised so the click will be relative to the front window, and click. Turn off recording, delete any extraneously recorded actions and you have your coordinates. Immediately after recording you can adjust the relative corner of the window or screen, and the recorded coordinates will adjust to match.

Remember that offsets are always to positive to the right and positive down, so if you are making a mouse click up from bottom edge of a window or screen, or left from the right edge of a window or screen, you will need to use negative coordinates.

How do I insert styled/colored text or images?

You can use the Insert Text action to insert styled text by pasting.

You can insert an image by copying it to the clipboard and then pasting it in using the Paste action (which just types the Command-V keystroke).

You can get your image from a Named Clipboard, or by reading an image file.

Here is how to create a Macro to insert an image when you press a Hot Key.

- Create a macro (see the How do I create a new Macro? section) with the Set Clipboard to Image action.
- Paste your image in to the action, or have the action read the image from a file or Named Clipboard.
- Then use the Paste action to paste your image.

Name the macro, and assign it whatever trigger you desire.

Now whenever you trigger the macro you just created, your image will be pasted in.

How do I Insert the current date?

You can insert text using the Insert Text action, and it processes Tokens. There are some basic date format tokens (such as the %LongDate% token), or you can use the %ICUDateTime% token with any ICU date format [https://www.stairways.com/action/linkthru?icudatetime].

How do I get more than one Macro Palette (Toolbar)?

There are three kinds of palettes in Keyboard Maestro:

- There is one "Global Macro Palette" which includes any active macro that has the Macro Palette trigger. It appears whenever there is any active macro with the Macro Palette trigger. It shrinks to the size of an icon until you hover over it and then it expands to display the currently active macros with Macro Palette triggers. You can show and hide it using specific Macro Palette actions.
- Each Macro Group can be displayed as a palette. The macro group can be global to all applications, or specific to any subset of applications. It can be toggled on and off with a hot key (or a status menu selection or from the Global Macro

Palette) or it can be displayed for a single action. Actions can hide or show the macro group palettes.
- When a hot key (or typed string or device key) conflicts (ie, triggers more than one macro), the Conflict Palette appears which lets you select from the conflicting macros. This can be an easy way to allow a single hot key to offer a multitude of similar actions. You can then use number keys or the mouse to select the desired hot key, or can use other keys to filter the palette until only one macro remains.

So to have more than one macro palette, create a macro group for each desired palette and configure it to show a palette as desired. Put your macros in there. Create as many of these as you like. The macros in such a macro group are only active while the palette is displayed, so if you only display it occasionally, especially only for one action, then they can have very simple hot keys (like plain letters for example).

You can control the order of macros in a macro palette (or the status menu) by prefixing their name with two characters and a closing parenthesis (eg "01)" - two characters and a closing bracket). The macros will be sorted based on the code, but the code will be stripped off before display in the palette (or status menu).

How do I use a multiple keystroke trigger?

You can assign the same hot key to multiple macros, and Keyboard Maestro will display a palette of the conflicting macros when you press that hot key allowing you to select the desired macro. You can select a macro from the palette using either number keys, or by typing the first distinct character to filter the macros down until only one is left. This is especially useful when you have a variety of similar or related tasks. You can also assign a hot key to a macro group which can activate it for one action (with or without a palette), and the contained macros can have whatever "second" hot key you desire.

But Keyboard Maestro does not directly support assigning a two-keystroke hot key to a trigger. The problem with multiple keystroke triggers like Option-F R is what to do if you type Option-F A? Logic dictates that the Option-F A should go through to the system unimpeded, but Option-F R should be swallowed entirely. But this is impossible. The only way to do it would be to swallow the Option-F key, and then swallow the second key and then resubmit the Option F and the second key unless it matches Option-F R.

However, that is fraught with peril and cannot work robustly in the presence of other applications placing things on the keyboard event queue (or even a sufficiently fast typist).

For example, suppose you quickly typed Option-F A B. Keyboard Maestro would have swallowed the Option F and then the A, and then resubmitted it to the event queue, resulting in the stream of characters B, Option-F, A. There is no way to avoid this race condition, and as such Keyboard Maestro does not support any such mechanism.

As described above, Keyboard Maestro has a variety of ways you can use Option-F as a hot key that allows a second key to be used to select a macro. However in all cases it is clear that the Option-F has been used and there is no concept that the Option-F might come back later to do something else.

How do I configure the Application Switcher?

The Application Switcher (and all the switchers) are activated by macro actions. Keyboard Maestro creates a default "Switcher Group" Macro Group containing several macros, each macro has a hot key trigger and a matching action which activates the appropriate switcher.

So to configure the Application Switcher, launch Keyboard Maestro, select the Switcher Group, and double click the Activate Application Switcher macro. You can then configure the various Application Switcher parameters, such as style and icon size by configuring the Application Switcher action. You can also configure the hot key used to activate the switcher, or disable the switcher.

How do I backup / migrate / transfer my installation to another Mac?

If you are going to continue using both Macs, you should use Macro Syncing rather than transferring your information. And indeed you can use Macro Syncing to transfer to a new Mac (even if both Macs wont be on at the same time as long as you preserve the sync file).

To transfer all your information to a new Mac you can copy the ~/Library/Application Support/Keyboard Maestro folder and optionally the ~/Library/Preferences/com.stairways.keyboardmaestro.* files to your new Mac. You can get to ~/Library in the Finder by holding down the shift (or option) key and selecting Library from the Go menu.

Make sure that Keyboard Maestro and Keyboard Maestro Engine are not running on the source or target Mac when you transfer the folder. And make sure the resulting files and folders have the **correct ownership and permissions**.

If you copy the com.stairways.keyboardmaestro.* files, the new Mac will have the same MacUUID token and be considered the same Mac for syncing purposes, so you should not do this if you intend to continue using the old Mac.

Alternatively, to just transfer the macros, you can turn on Macro Syncing and save the macro sync file anywhere. Transfer it to the new Mac, install Keyboard Maestro as normal, and then turn on macro syncing and select the transferred file to replace your macros. Then turn off macro syncing. This will transfer just the macros, not any settings, variables, or clipboards or the like.

How do I uninstall Keyboard Maestro?

Launch Keyboard Maestro and ensure the "Launch Engine at Login" preference in the General preference pane is turned off. Then choose the File ➤ Quit Engine menu to quit the engine, and then choose the Keyboard Maestro ➤ Quit Keyboard Maestro menu to quit the application.

You can then trash the Keyboard Maestro application from your Applications folder, as well as the preferences in the ~/Library/Application Support/Keyboard Maestro folder and ~/Library/Preferences/com.stairways.keyboardmaestro.* files and logs in the ~/Library/Logs/Keyboard Maestro folder.

How do I revert to a previous version of Keyboard Maestro?

Launch Keyboard Maestro and ensure the "Launch Engine at Login" preference in the General preference pane is turned off. Then choose the File ➤ Quit Engine menu to quit the engine, and then choose the Keyboard Maestro ➤ Quit Keyboard Maestro menu to quit the application. You can then trash the Keyboard Maestro application from your Applications folder.

For version 2, open the ~/Library/Preferences folder and the folder ~/Library/Preferences/Keyboard Maestro/Saved Version 2 folder. Move the files from the latter folder into the former folder. Trash the ~/Library/Preferences/Keyboard Maestro folder. Download Keyboard Maestro 2.1.3, if necessary, from https://files.stairways.com/keyboardmaestro/keyboardmaestro-213.dmg [https://files.stairways.com/keyboardmaestro/keyboardmaestro-213.dmg]. Move Keyboard Maestro 2 to the Applications folder and launch it. Turn on the "Launch Engine at Login" preference if desired.

For version 3, open the ~/Library/Preferences/Keyboard Maestro folder. Trash the Keyboard Maestro Macros.plist and replace it with the Keyboard Maestro Macros Saved Version 6.plist. Download Keyboard Maestro 3.5, if necessary, from https://files.stairways.com/keyboardmaestro/keyboardmaestro-35.zip [https://files.stairways.com/keyboardmaestro/keyboardmaestro-35.zip]. Move Keyboard Maestro 3 to the Applications folder and launch it. Turn on the "Launch Engine at Login" preference if desired.

For version 4, open the ~/Library/Preferences/Keyboard Maestro folder. Trash the Keyboard Maestro Macros.plist and replace it with the Keyboard Maestro Macros Saved Version 6.plist. Download Keyboard Maestro 4.3.2, if necessary, from https://files.stairways.com/keyboardmaestro/keyboardmaestro-432.zip [https://files.stairways.com/keyboardmaestro/keyboardmaestro-432.zip]. Move Keyboard Maestro 4 to the Applications folder and launch it. Turn on the "Launch Engine at Login" preference if desired.

For version 5, open the ~/Library/Application Support/Keyboard Maestro folder. Trash the Keyboard Maestro Macros.plist and replace it with the Keyboard Maestro Macros Saved Version 6.plist. Download Keyboard Maestro 5.3.2, if necessary, from https://files.stairways.com/keyboardmaestro/keyboardmaestro-532.zip [https://files.stairways.com/keyboardmaestro/keyboardmaestro-532.zip]. Move Keyboard Maestro 5 to the Applications folder and launch it. Turn on the "Launch Engine at Login" preference if desired.

For version 6, open the ~/Library/Application Support/Keyboard Maestro folder. Trash the Keyboard Maestro Macros.plist and replace it with the Keyboard Maestro Macros Saved Version 6.plist. Download Keyboard Maestro 6.4.8, if necessary, from

https://files.stairways.com/keyboardmaestro/keyboardmaestro-648.zip [https://files.stairways.com/keyboardmaestro/keyboardmaestro-648.zip]. Move Keyboard Maestro 6 to the Applications folder and launch it. Turn on the "Launch Engine at Login" preference if desired.

For version 7, open the ~/Library/Application Support/Keyboard Maestro folder. Trash the Keyboard Maestro Macros.plist and replace it with the Keyboard Maestro Macros Saved Version 7.plist. Download Keyboard Maestro 7.3.1, if necessary, from https://files.stairways.com/keyboardmaestro/keyboardmaestro-731.zip [https://files.stairways.com/keyboardmaestro/keyboardmaestro-731.zip]. Move Keyboard Maestro 7 to the Applications folder and launch it. Turn on the "Launch Engine at Login" preference if desired.

For version 8, open the ~/Library/Application Support/Keyboard Maestro folder. Trash the Keyboard Maestro Macros.plist and replace it with the Keyboard Maestro Macros Saved Version 8.plist. Download Keyboard Maestro 8.2.4, if necessary, from https://files.stairways.com/keyboardmaestro/keyboardmaestro-824.zip [https://files.stairways.com/keyboardmaestro/keyboardmaestro-824.zip]. Move Keyboard Maestro 8 to the Applications folder and launch it. Turn on the "Launch Engine at Login" preference if desired.

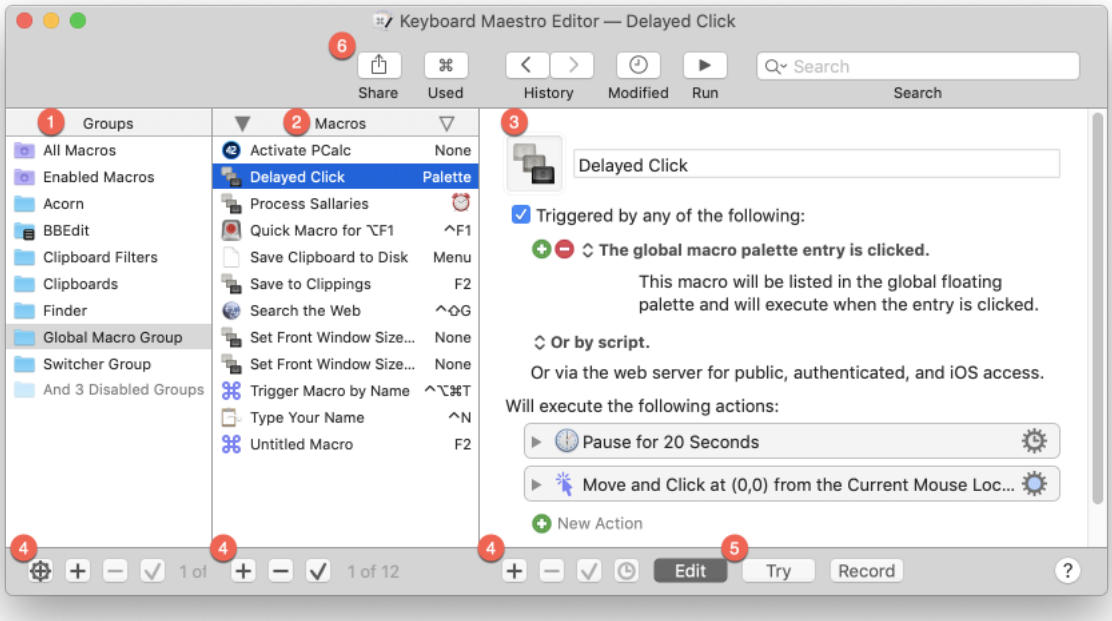
For version 9, open the ~/Library/Application Support/Keyboard Maestro folder. Trash the Keyboard Maestro Macros.plist and replace it with the Keyboard Maestro Macros Saved Version 9.plist. Download Keyboard Maestro 9.2, if necessary, from https://files.stairways.com/keyboardmaestro/keyboardmaestro-92.zip [https://files.stairways.com/keyboardmaestro/keyboardmaestro-92.zip]. Move Keyboard Maestro 9 to the Applications folder and launch it. Turn on the "Launch Engine at Login" preference if desired.

Editor Window

To edit your macros, launch the Keyboard Maestro application. It will display one or more Editor Windows. If you close the last editor window, the Keyboard Maestro application will quit, but the Keyboard Maestro Engine will continue running to perform your macros. Normally you only launch the Keyboard Maestro editor application when you want to change your macros, and leave only the Keyboard Maestro Engine running normally.

You can create a new editor window by choosing the File ➤ New Editor Window menu.

Window Elements



Main Elements

1. ***Groups Column*** 1️⃣ (on the left)
    - List of all of the Macro Groups and Smart Groups (shown at top) in your account.
    - Smart Groups are saved searches of macros.
2. ***Macros Column*** 2️⃣ (in the middle)
    - List All of the Macros in the Selected Macro Group.
    - Macros are the primary way you describe what you want Keyboard Maestro to do.
    - You can filter the list to find specific macros by using the Search Field (upper right corner).
3. ***Details Column*** 3️⃣ (on the right)
    a. Display or Edit the details of the selected Macro, Macro Group, or Smart Group.
        - This is where you enter the actual details of the selected Macro.
        - Layout changes depending on which type is selected.
    b. **If a *Macro* is selected**, shows the following elements:
        I. *Macro Name*
        II. *Enabled Checkbox* ✅ (shown just before the text "Triggered by any of the following").
            - Must be checked to enable your Macro to be triggered.
        III. *Trigger List* (if any) that have been assigned to your Macro.
        IV. *Action List* – the meat of your Macro that provides the processing steps to be performed.
    c. **If a *Macro Group* is selected**, shows the following elements:
        I. *Macro Group Name*.
        II. *Enabled Checkbox* ✅ (shown just before the text "Enable Macro Group").
            - Must be checked to enable your Macro Group.
            - If unchecked, none of the Macros in that Macro Group can be triggered.
4. You can resize the columns with the dividers 4️⃣, and double click on them to auto-resize the columns.

Toolbars

1. **Toolbar at Bottom**
    - Below each of the above columns are tools (buttons) to be used only for that column.
    - See below for details
2. **Toolbar at Top**
    - At the top of the window are tools generally apply to all Macros.
    - See below for details.

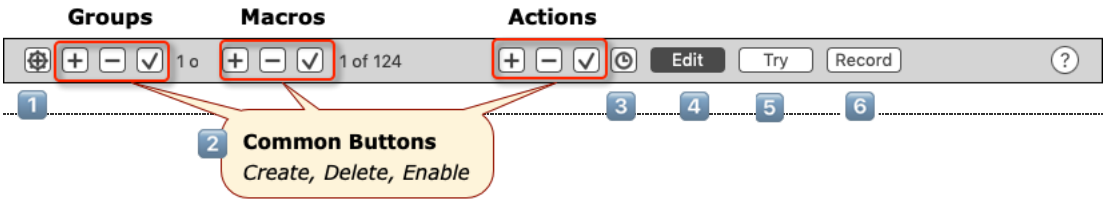
Toolbar at Top

Along the top of the window is the toolbar. From left to right the buttons let you:

1. ***Share*** 1️⃣ Share your Macros or Actions, including directly to the Keyboard Maestro Forum [https://forum.keyboardmaestro.com/].
2. ***Used*** 2️⃣ Select the most recently used (triggered) macros.
3. ***History*** 3️⃣ Select from the recently edited macro history.
4. ***Modified*** 4️⃣ Select the most recently edited macro.
5. ***Run*** 5️⃣ Run the selected macro.
6. ***Search*** 6️⃣ Search for Macros and Actions.
    - Press ⌘F to search Macros only in the selected Macro Group.
    - Press ⌘⌥F (or ⌘F a second time) to search all Macros.
    - If you enter simple text, it will search for any text in any Action, or Macro Name, that contains that text.
    - To better restrict the Search, use Search Qualifiers.
    - Tap a modifiers key to enter the corresponding modifier symbol.
    - The *Macros Column* will be filtered to only matching macros.
    - Any parts of the macro that match will be highlighted in striped blue in the *Details Column*.

- If any macros are filtered out, an entry will show at the bottom of the macro column list representing all the hidden macros.

Toolbar at Bottom



Buttons:

1. **Common Buttons** 2 at the bottom of each column
   (Applies to Item in that column)
   a. Create (+)
   b. Delete (-)
   c. Enable/disable (✓)
2. **Groups Column Buttons**
   a. Create Smart Group ⊕ 1
3. **Details Column Buttons**
   a. Set Timeout for selected Action 3
   b. Toggle Edit Mode Edit 4
   c. Try the selected actions Try 5
      I. This is for testing *Actions*, NOT the entire Macro.
      II. To test the entire Macro, click the Run Button ▶ in the Top Toolbar
   d. Start Macro Recording Record 6

Editing

Edit mode may be toggled on/off by clicking the Edit button on the Bottom Toolbar, or by selecting the menu *View* > *Stop/Start Editing Macros*. Note that macros are saved immediately after every change (there is a small "dirty" marker in the bottom right corner of the editor window that shows when the save occurs). There is no need to ever turn off Edit mode, or do anything else to have your macro changes be recognised by the Keyboard Maestro Engine.

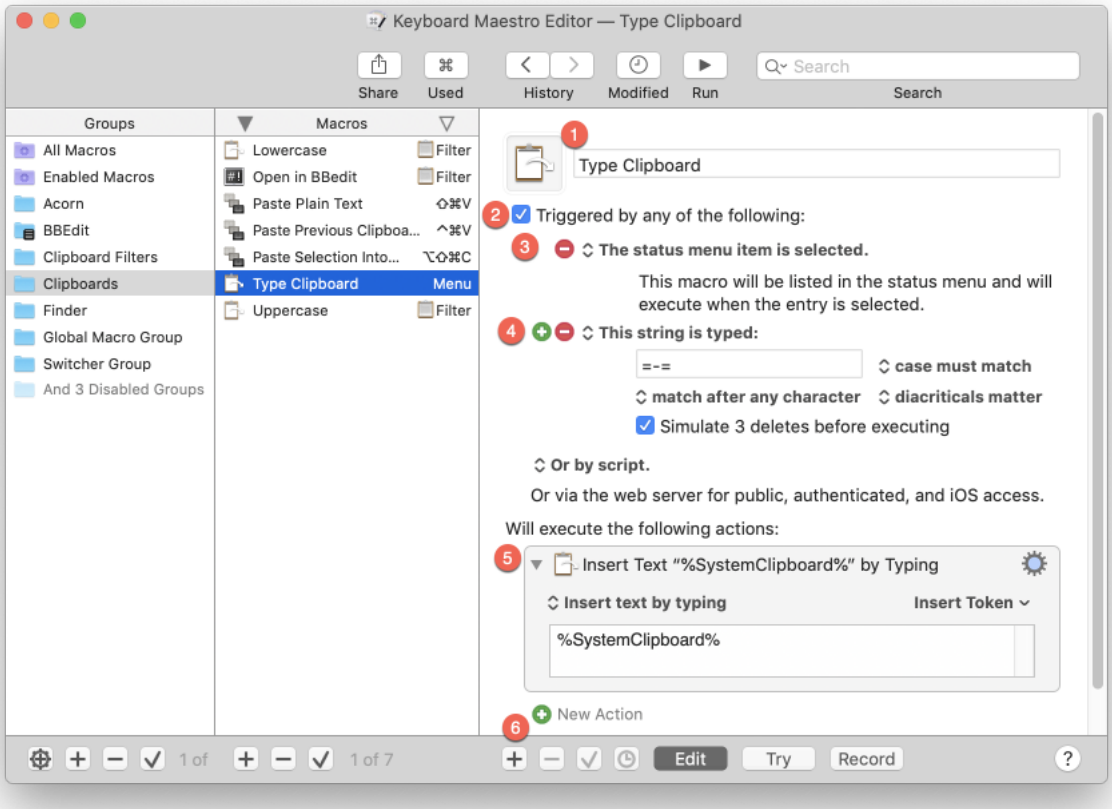
Actions

You can add a new action by any of the following:

1. Click the New Action button.
2. Click the + button at the bottom of the *Details Column* to display the list of actions.
3. Choose the Edit ➤ Insert Action By Name menu.
4. Press ⌘^A to show a Insert Action By Name Popup.
5. Choose from the Edit ➤ Insert Action menu.

Macros

When you are in Edit Mode and you select a Macro, you can edit its behaviour.



1. ***Icon and Name*** 1 You can adjust the icon by double clicking the icon and selecting an image in the Icon Chooser and you can change the name of the Macro.
2. ***Enabled*** 2 You can enable or disable the macro with this checkbox (a macro must be enabled for it to be available to be triggered (executed) - its parent macro group must also be enabled and active).
3. ***Triggers*** 3 You can add Macro Triggers to the macro which control when the macro will run.
4. ***Triggers*** 4 You can add more than one trigger to define different ways that you can start the macro.
5. ***Actions*** 5 You then add one more Macro Actions which definite when the macro does when it runs.
6. ***New Action*** 6 Click the New Action button to add additional actions.

Triggers

A macro consists of a set of triggers 3. When it is active (ie, when it is enabled, and its parent Macro Group is enabled and active), if any of these triggers happen, then the macro executes the actions.

You can add a trigger by clicking and holding on the green + button and selecting the kind of trigger you would like to use.

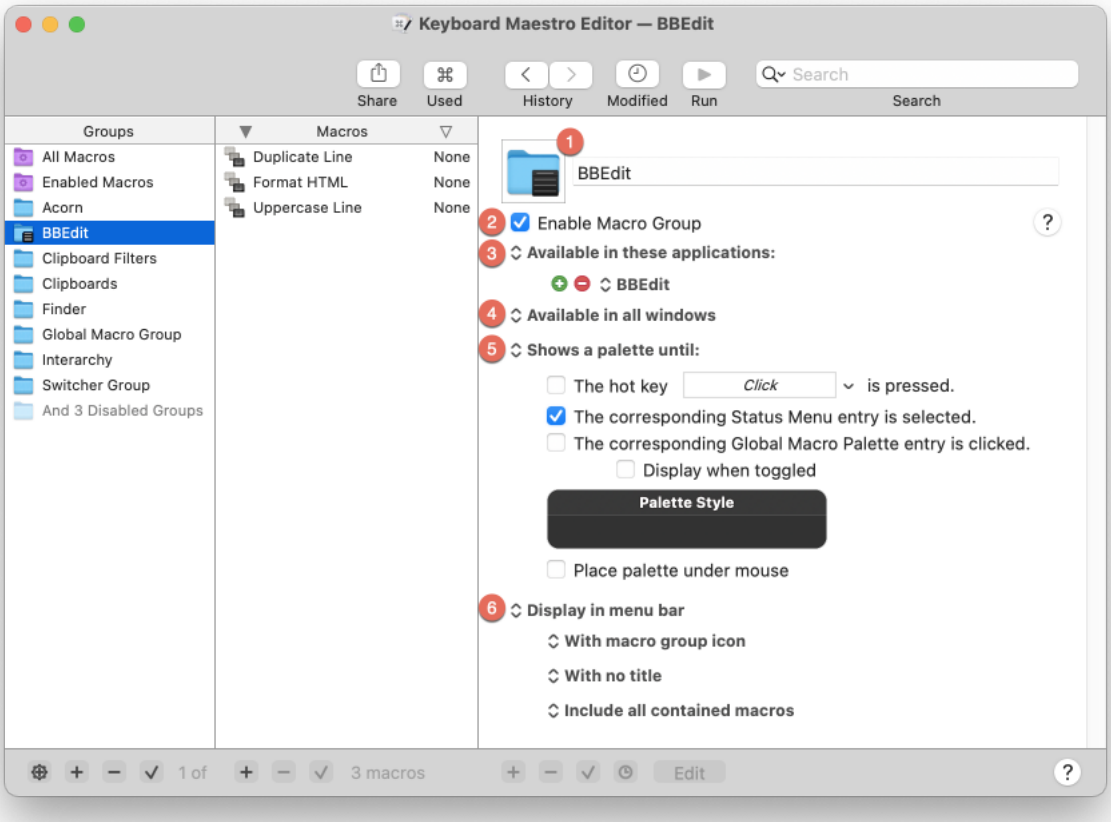
Some triggers, like the Status Menu trigger 3, do not have any additional configuration. Simply adding them to the macro will have macro triggered when the event happens (in this case, the macro will be listed in the Keyboard Maestro status menu, and will be triggered if you choose it from that menu).

Other triggers, like the Typed String trigger 4, require you to configure more details about when the trigger should happen, such as the specific string you have to type to trigger the macro. For the macro shown, the macro will be triggered when you type the string "=-=".

When a macro is triggered, it will execute the actions 5.

Macro Groups

When you are in Edit Mode, and you select a Macro Group, you can edit its configuration.



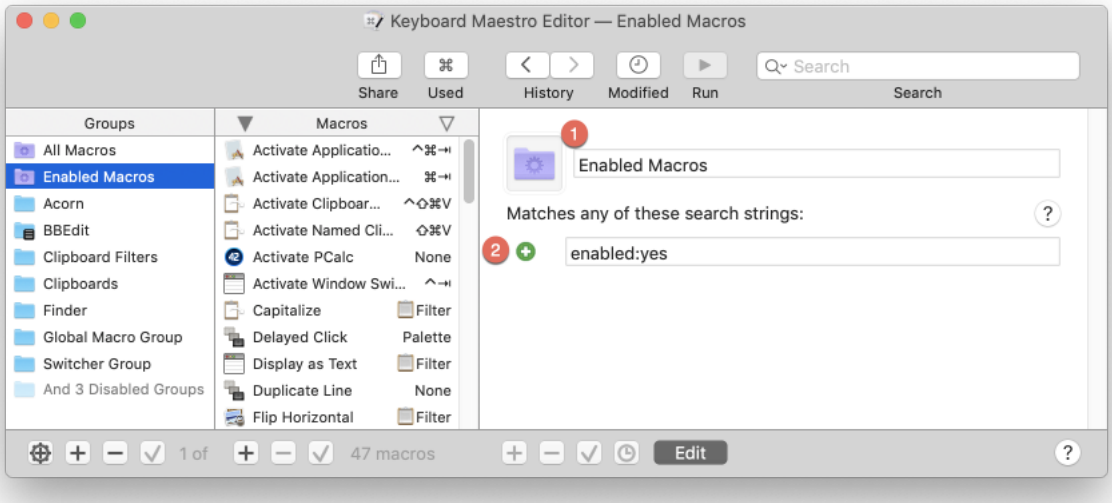
You can adjust the icon by double clicking the icon 1 and selecting an image in the Icon Chooser and you can change the name of the Macro.

For a Macro to be available to trigger, it must be enabled and its parent Macro Group must be enabled 2, and active. The configurations for the Macro Group control when it will be active, and thus when the macros it contains will be active (available to be triggered and executed).

You can adjust whether the macro group is enabled 2, whether it is only active in some specific applications 3 (ie, when they are at the front, and their menu bar is showing), whether it is only active in specific windows 4, whether it is only active when you explicitly toggle it on or off, and whether and how it shows a palette containing its macros 5 and whether and how the macro group displays in the menu bar 6.

Smart Groups

When you are in Edit Mode, and you select a Smart Group, you can edit its search settings.



You can adjust the name of the Smart Group 1, and adjust its Search Strings 2.

You can add another Search String by clicking the green + button. Any macro that matches any of the search strings will be listed in the Macros column.

Macro Groups

Keyboard Maestro organizes your macros into Macro Groups which are like folders of macros. Each Macro Group contains a number of macros and controls when those macros are active.

A Macro Group can target or exclude specific applications, which means the macros it contains will only be active in those desired applications or when those applications are running. For example, you can have macros which are active only in Mail.app. It can also be active only in specific windows. So for example, you can have macros which are active only in Mail.app, or only in Mail.app when a compose window is open.

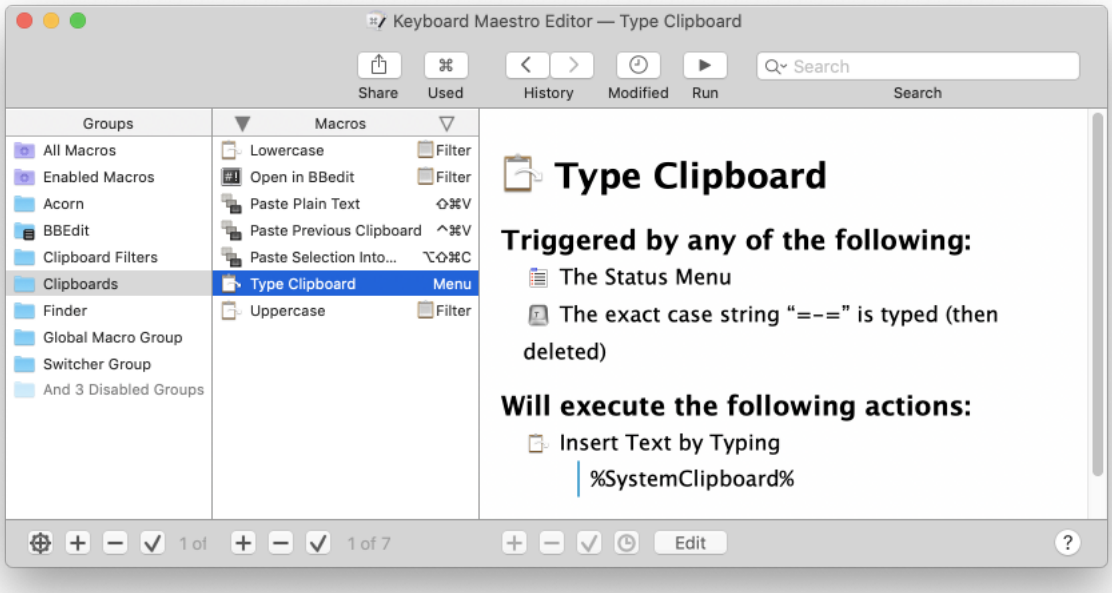
A Macro Group can also act as a container for specific-use macros which are enabled only when you specifically activate them. For example, you could create a Macro Group containing macros that resized or repositioned windows using the arrow keys, but those macros would only be active after the Hot Key was pressed so that the arrow keys could be used normally at other times.

Macro Groups can be displayed as palettes, allowing you to create your own custom toolbars which can be configured with a variety of themes and to appear under the mouse.

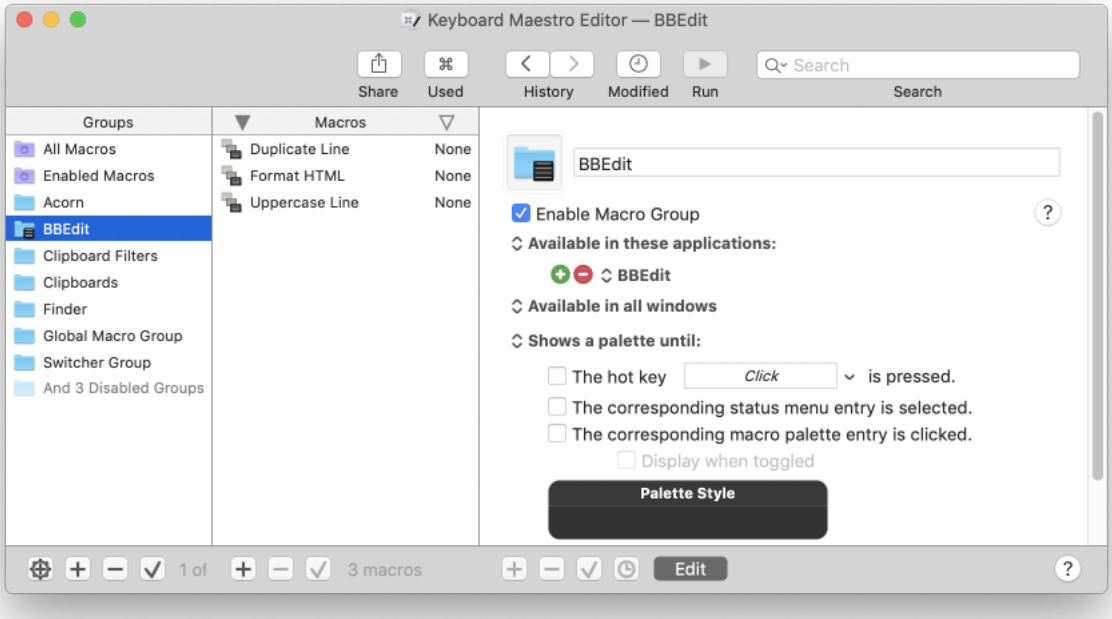
Macro Groups can also be displayed in the menu bar, including information that is displayed and dynamically updated, as well as offering another way to trigger macros.

If you are syncing your macros with another Mac, Macro Groups can be disabled specifically on this Mac.

To create a new Macro Group, first launch Keyboard Maestro.



Now click the + button at the bottom of the Macro Groups list.



Enter the name of your new Macro Group.

You can choose to target the macros in your Macro Group at specific applications.

By default, Macro Groups and their Macros are available in all applications, that is they are always ready to be triggered. These are especially useful for Macros that give you instant access to applications or documents, or type in globally applicable text. For example you might have a Macro to launch your email client or word processor, a Macro to open your financial accounts, and a macro to type your name or email address.

To have macros only active in specific applications, set the Macro Group to be "Available in the following applications" and add the desired applications to the list.

For example, you could have macros targeted at:

- Mail.app that insert common text messages.

- BBEdit and Xcode that insert code chunks or duplicate lines or add #include headers.
- Safari that configure windows or enter information.
- Photoshop or Acorn that arrange items or script guides.

To have macros active everywhere **except** specific applications, set the Macro Group to be "Available except in the following applications" and add the desired applications to the list. For example, you could exclude macros from triggering in games. Also, if you have an application that uses lots of function keys for crucial tasks, you could exclude that application to allow you to use the function keys for macros elsewhere without conflicting with that application.

You can restrict a macro to be active only in certain windows based on the window title.

You can also choose to activate the macros manually or display the macros in a floating macro palette. The options are:

- Always activated.
- Activated for one action when:
- Activated/deactivated when:
- Shows a palette for one action when:
- Shows/hides a palette when:
- Shows a palette until:
- Always activated and shows a palette for one action when:
- Always activated and shows/hides a palette when:
- Always activated and shows a palette until:

To have the macros in a Macro Group always ready to be triggered, set the Macro Group to be "Always activated".

To have macros that are active only immediately after you trigger the Macro Group with no visible palette, set the Macro Group to be activated "Activated for one action when". The macros in the Macro Group will be enabled when you trigger the Macro Group and will remain enabled until either any macro is triggered or you press any other key. You could use this to create a set of related actions with easily remembered hot keys that will not conflict with normal use because they are not activated until you trigger the group. For example, you could have a group of macros to launch various applications so that Command-Control-L activates the group and then a single letter press launched the application (eg M for Mail, S for Safari, F for Finder).

As with each of the following options, you can trigger the macro group by pressing a Hot Key, by selecting from the Status Menu, or by clicking on the global floating palette.

To have Macros that are remain active after you trigger the Macro Group, set the Macro Group to be "Activated/deactivated when". The Macros in the Macro Group will be enabled when you trigger the Macro Group and will remain enabled until you dismiss the Macro Group by repeating the trigger. You could use this to create a set of related actions with easily remembered hot keys that will not conflict with normal use because they are not activated until you press the group Hot Key. For example, you could have a group of macros to move and resize windows and have Command-Control-W activate the group. Then a single arrow key press moves the front window. When the window is positioned, press Command-Control-W a second time to disable the macros.

To have Macros that are active and displayed in a macro palette only immediately after you trigger the Macro Group, set the Macro Group to "Shows a palette for one action when". The Macros in the Macro Group will be displayed in a floating Macro Palette and enabled when you trigger the Macro Group and will remain displayed until either any macro is triggered or you press any other key. You could use this to create a set of related actions that do not even need a Hot Key. For example, you could have a group of macros to launch various applications like Mail, Safari and the Finder so when you press Command-Control-L, a palette of these macros is displayed and a single click on the desired application will launch the application.

To have Macros that are active and displayed in a macro palette after you trigger the Macro Group, set the Macro Group to "Shows/hides a palette when". The Macros in the Macro Group will be displayed in a floating Macro Palette and enabled when you trigger the Macro Group and will remain displayed until you trigger the Macro Group again. You could use this to create a set of related actions that do not even need a Hot Key. For example, you could have a group of macros to align objects in a CAD application, so when you press Command-Control-A, a palette of these macros is displayed and you can click various alignment options (distribute left-right, align top edges) and then close the palette by pressing Command-Control-A a second time.

To have Macros that are initially active and displayed in a palette, set the Macro Group to "Shows a palette until". The Macros in the Macro Group will be displayed in a floating Macro Palette and enabled. The palette will close when (if) you trigger the Macro Group and will remain closed (and the macros disabled) until you trigger the Macro Group again. You could use this to create a set of related actions that do not even need a Hot Key and that are available in a Macro Palette. You could make the Macro Group available only in a specific application so that it appears only in that application. For example, you could have a group of macros to align objects in a CAD application, and have the Macro Group available (and hence the palette displayed) only in the CAD application.

Whether the macros are displayed in a Palette or not, the macros can still have Hot Key (or any other kind of) triggers which will be available whenever the macro group is active.

With the previous three settings, the macros are active if and only if the palette is displayed. The final three options are the same except that the macros are always active, so they are always available to be triggered by another means, and sometimes by the palette.

You can also display the Macro Group in the menu bar. You can select which icon to display and what title text to show in the macro bar, both of which can be dynamically updated allowing you to show information like stock prices, word counts, reminders, time, etc. For example, you could have an icon and time that showed how long you had been working without a break. The menu can include all the macros in the group, or just those macros with the Group Status Menu trigger.

You can change the targeting of an existing group by selecting it in the Groups list and ensuring you are in Edit mode by clicking the Edit button, or by double clicking the macro group.

You can disable or enable a macro group by selecting it and clicking the ✓ button at the bottom of the Groups list. If a Macro Group is disabled or inactive for any reason, all its contained Macros will be inactive. You can disable or enable a Macro Group using the Set Macro Enable action, you can show in a palette or activate a Macro Group using the Show Macro Group or Activate Macro Group actions.

Disabled Macro Groups can optionally be hidden using the choosing the View ➤ Hide Disabled Macro Groups menu.

You can control or right click on a macro group to:

- Cut, Copy, Duplicate or Delete the Macro Group.
- Copy the Macro Group as XML, or its UUID.
- Copy the Macro Group as a Set Macro or Group Enable or Toggle Macro Group action.
- Disable or Rename the Macro Group.
- Export the entire Macro Group.

The Keyboard Maestro editor has full AppleScript support so you can manipulate macro groups in many ways via AppleScript. See Scripting the Keyboard Maestro editor.

To delete a Macro Group, select the macro group and then press Command-Delete or click the - button at the bottom of the Groups list. Like all actions, you can Undo this if you make a mistake.

You cannot delete or rename or disable the Global Macro Group. If you want to control when some of the macros within the Global Macro Group are active, make a new macro group and drag those macros to it.

Smart Groups

You can create Smart Groups which let you see macros matching certain criteria by clicking the ⊕ button at the bottom of the Groups column, and you can edit a smart group selecting it in the Groups list and clicking the Edit button, or by double clicking the Smart Group.

A Smart Group consists of a set of Search Strings, and any macro that matches any of the search strings will be listed in the Smart Group.

You can control or right click on a smart group to Rename or Duplicate it.

Macros

Macros are used to automate your workflow, procedure, or process on your Mac. Macros, also known as Shortcuts, are a way of improving your productivity by allowing you to perform repetitive or frequently required actions more quickly and accurately, tailoring your Mac to your usage patterns.

A *Macro* consists of:

- One or more Macro Triggers which define when the macro will be executed, if it is Active.
- A series of steps to execute called Macro Actions.

Macros are contained in Macro Groups, much like files are contained in folders. The Macro Group determines when the Macro is Active (available to be triggered). Macro Groups control such things as which applications the Macro will be active in.

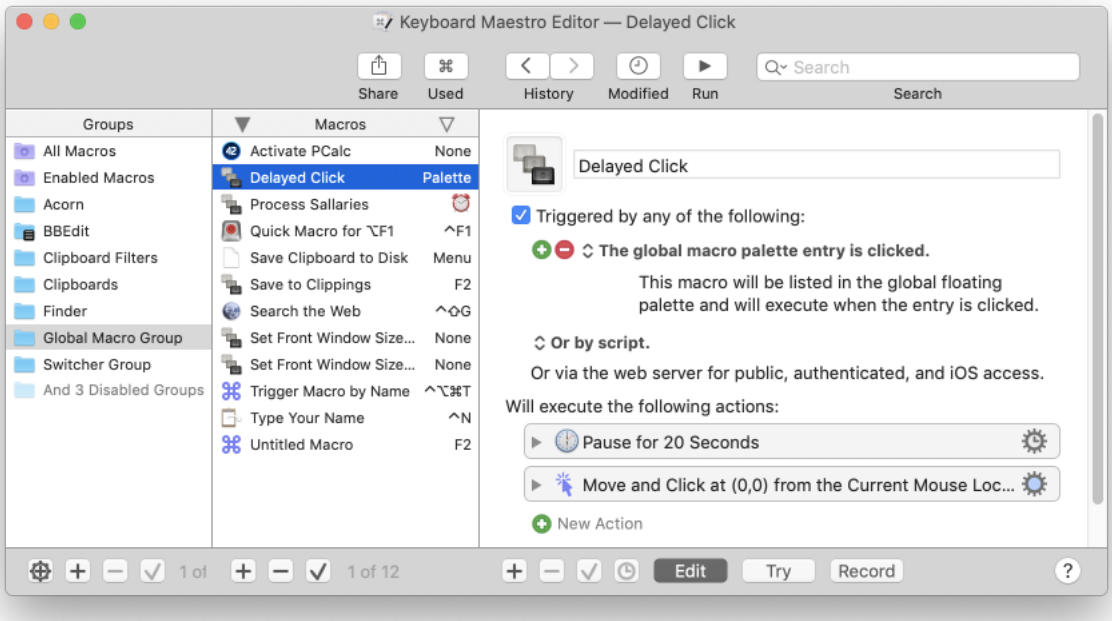
A simple example of a Macro is one which has:

- A Single Hot Key trigger, such as ^ E
- A single action like Insert Text by Pasting the text "SomeName@SomeDomain.com"
- Is in the "Global Macro Group", making it available to all applications.

Macros can do almost anything you can do manually on your Mac, and much that you can't, like running scripts.

How To Create a Macro

To create a Macro, first launch Keyboard Maestro, select the desired Macro Group to contain it, and then click the + button below the Macros list. To edit a Macro, double click it, or select it and ensure the Edit button is pressed. The Macro Editor window will be displayed.



Enter the name of your new Macro (you can skip this and it will be named for you based on the action you select).

You can create a new trigger by clicking on the green ⊕ button. There are several Macro Triggers to choose from, the most common being the Hot Key trigger which allows you to execute the Macro at the press of a key. You can define several different triggers, and any of the triggers will execute the Macro Actions. You can delete a trigger by clicking the red ⊖ button.

You can add a new action by clicking the New Action button or the + button at the bottom of the macro detail view to display the list of actions or by choosing the Edit ➤ Insert Action By Name menu or choosing from the Edit ➤ Insert Action menu. There are many Macro Actions to choose from. The actions you include will be executed in order. You can reorder the

actions by dragging them around. You can copy actions by Option-Dragging or by using Copy and Paste. You can delete an action by selecting it and pressing the Delete key. You can enable or disable actions by selecting them and clicking the ✓ button at the bottom of the macro detail view.

You can control or right click on a macro group to Disable, Enable, Rename, Duplicate, or Export it.

An easy way to generate macro actions is to turn on recording by clicking the Record button at the bottom of the macro detail view and proceeding to show Keyboard Maestro what you want to do. Then turn recording back off and look through the actions. Chances are you will need to delete or adjust some of the recorded actions to make a robust macro, but this will be quicker than creating each action manually.

Macros are continuously saved, so the macro is live as soon as it is created. It will be available immediately (subject to the restrictions of the Macro Group it is contained in). There is no need to turn off editing, switch to a different macro, or quit the editor (although quitting the editor after you have finished editing all your macros is a good idea).

You can select a recently triggered macro by clicking on the ⌘ toolbar button to select it or select from recently modified macros by clicking the ◷ toolbar button.

You can share a macro to the Keyboard Maestro Forum, or to a friend via Mail or Messages using the sharing button at the top of the Macros window.

You can disable or enable a macro by selecting it and clicking the ✓ button at the bottom of the Macros list. You can also disable or enable a Macro using the Set Macro Enable action, or from AppleScript with:

```
tell application "Keyboard Maestro"
        set enabled of macro group "Macro Name" to true
end tell
```

You can start editing a Macro from AppleScript with:

```
tell application "Keyboard Maestro"
  editMacro "Macro Name or UID"
end tell
```

The Keyboard Maestro editor has full AppleScript support so you can manipulate macros in many ways via AppleScript.

How to Edit a Macro

You edit a macro by selecting it and ensure you are in Edit mode. Double clicking on the macro will turn on Edit mode.

You can select a macro by name by choosing the View ➤ Select Macro By Name menu.

How to Run a Macro

In order to run (execute or trigger) a macro it must be active, that is:

- It is enabled
- It's containing Macro Group is enabled
- It's containing Macro Group meets all its activation criteria.

Activation/Deactivation or a Macro Group is a dynamic process, automatically set by the Keyboard Maestro Engine as your Mac's environment changes in real time. For details, see Macro Activation

You can try out a macro from the Keyboard Maestro by clicking the ▶ Run button at the top of the editor window. Of course, macros are often sensitive to the context, so it may not be appropriate to run them directly from the editor.

Generally, you run a macro by invoking any one of its triggers.

You execute a macro's action sequence by invoking one of its Macro Triggers that you have defined (such as pressing a hot key if you have configured a hot key trigger). For some triggers, no action is required on your part. For example, the macro could

be triggered at a specific time, or when a specific USB device is attached.

You can also trigger macros by name using the Trigger Macro by Name action, which in turn can be in a macro and triggered any way you desire.

You can trigger a macro using the kmtrigger: scheme with a URL like `kmtrigger://macro=MacroName&value=Value` (the value is accessible via the %TriggerValue% Text Token).

If the web server is enabled for remote access, you can trigger a macro remotely after logging in using a web browser or the Keyboard Maestro Control iPhone application. Alternatively, you can use the Remote trigger and trigger your macro using a URL from remotely via our trigger server.

You can also trigger a macro using AppleScript or another scripting language (select the "Or by script" entry to display script code in various languages such as AppleScript or Perl).

Note that the web server needs to be separately enabled in the Web Server preference pane, and all macros are subject to the restrictions of the Macro Group they are contained in. If the Macro Group is not enabled and active, the macro will not be available.

How to Import Macros

You may want to import Macro(s) you have downloaded from other sources, like the Keyboard Maestro Forum [https://forum.keyboardmaestro.com/].

You may use any of the following to import the macro file (`.kmmacros`):

- Double-click on the macro file in the Mac Finder.
- In the Keyboard Maestro app, goto *File* > *Import Macros Safely*… and select the macro file.
- In some browsers, click on the downloaded file icon at the bottom of the browser window.

Note that the macro file (`.kmmacros`) may contain one or more macros, and will also create the Macro Group if it does *not* exist in your configuration. Otherwise, it will put the Macro in your existing Macro Group.

Import Macros Safely

⚠ You should be aware that the macros will be imported in the same state, triggers, and macro group that they were saved in. By default, macros are imported disabled unless you hold the Option key down. Importing them disabled is important, because otherwise **this could result in the imported Macro being triggered (executed) as soon as it is imported.** If you are at all uncertain about the source of the macros, ensure you import the macros disabled or choose the File ➤ Import Macros Safely menu.

See also the Macro Groups, Macro Actions, Macro Triggers, Macro_Activation, Macro Examples, Macro Library and Recording sections.

Macro Triggers

A Macro is executed when any of its Macro Triggers is activated. There are several triggers to choose from, the most common being a Hot Key, that is a Macro is executed in response to a keystroke, usually in conjunction with one or more modifier keys. You can also trigger a macro by typing a string. Or you can display a Macro Group as a floating palette, or execute macros remotely.

Hold the option key down while adding a new trigger to get help on that trigger.

You can also trigger macros by name using the Trigger Macro by Name action, which in turn can be in a macro and triggered any way you desire.

The Hot Key trigger is perhaps the most common and most basic of all triggers. When you press the configured keyboard key, the system swallows the keystroke, and Keyboard Maestro executes the macro.

The Typed String trigger is also very common, it lets you execute a macro in response to a sequence of keys like an abbreviation.

The Global Macro Palette trigger lets you add your macro to a floating palette, so you can trigger it by clicking on the macro name in the palette.

The Status Menu trigger lets you add your macro to the Keyboard Maestro status menu so you can trigger it by selecting the macro name from the status menu.

There are many more triggers, and you can learn more about them on the wiki Triggers page.

Macro Actions

A Macro executes a sequence of Macro Actions in order. There are many actions to choose from. Some simple actions, such as Sleep Computer, require no other information and simply do their job, while other more complex actions, such as Select Menu Item, require you to specify more information, such as a target application or menu name. You do this by editing the values for the action.

Keyboard Maestro actions have two very distinct types of fields: text token fields and numeric calculation fields. You can learn more on the wiki Text Fields page.

There are many powerful Macro Actions available for your use, and you can sequence them together to perform complex tasks.

To add an action, edit your macro, click the New Action button, or equivalently the + button below the detail view. This will show the lists of possible actions.

To see all actions, select the All Actions category. To see plug in actions that you have added to Keyboard Maestro, select the Third Party Plug Ins category.

To select just your favorite actions, select the Favorites category. You can drag actions from your macro into your Favorites category, or control or right click on an action to add it to your favorites. Favorite actions will retain all the settings of the actions. You can delete favorite action in the action selector, or from the Add Action by Name window.

Double click or drag one or more of the actions to add them to the action list for the currently edited macro.

Alternatively, choose the Edit ➤ Insert Action by Name menu and quickly search for and add an action by name, or choose from the Edit ➤ Insert Action by Name menu. These will also include any favorite actions you have configured. If you hold the shift key down the action will be inserted above the currently selected action.

You can drag Macro Groups or Macros into the action list. With no modifiers, this will create an Execute Macro Action or Show Macro Group action. With the Option key, this will copy in the macro's actions. With Command-Shift, this will create an Enable Macro Group or Macro action. With Option-Shift, this will create a Mark the Macro action.

You can drag a file into the action list. This will create an Activate Application, Execute Script, or Open File action.

You can use control-up/down arrow to move the selected action up or down in the list, as well as command-control arrows to move the selected action to start or end of the list.

You can use the Engroup or Degroup from the Actions menu to enclose actions in other containers (or remove them from containers).

You can enable or disable actions by selecting them and clicking the ✓ button at the bottom of the macro detail view. You can set the timeout or other options on an action clicking the ⚙ menu at the top right of the action and selecting an option.

You can display more or less detail about some actions by clicking the disclosure triangle. You can disclose all the actions (in a sublist) simultaneously by option clicking on the disclosure triangle.

You can try the action immediately by clicking the ⚙ button at the top right of the action and selecting Try. Alternatively, you can try the selected actions immediately by clicking the Try button at the bottom of the macro detail view.

You can control or right-click on an action to act on it, including pasting other actions above or below it.

You can configure the action using the various menus and text fields, as well as by clicking the ⚙ button and using the action menu. This allows you to rename actions, color them, add notes, configure their time out and failure behaviour and more.

The action menu also allows you to get help on the specific action.

You can share an action sequence to the Keyboard Maestro Forum, or to a friend via Mail or Messages using the sharing button at the top of the Macros window.

You can learn more about the available actions on the wiki Actions page.

Macro Syncing

Keyboard Maestro is licensed on a per user basis on up to five Macs, so if you use it on two or more Macs you may want to transfer some of your macros from one to the other. You can do this by selecting the desired macros and choosing the File ➤ Export Macros menu to export your macros, and then importing them on the target Mac.

Alternatively, you may want (almost) all your macros on both Macs, in which case you can set up your macros to sync between them. This means any change you make on one Mac will be mirrored on the other and vice versa, although you generally should not edit your macros simultaneously on both Macs.

NOTE: You can only sync between identical versions of Keyboard Maestro. If the sync file is newer than the current version, you will be warned to upgrade, turn off syncing, or live dangerously (which temporarily disables syncing and allows you to make changes that will likely cause trouble later).

To do this, you need a file location that is mirrored on both Macs. Dropbox or a similar service is good for this, or you can use a shared file server, although you must ensure it is available at all times to both Macs.

You are then ready to start syncing your macros.

⚠ **WARNING: Syncing macros is an all or nothing affair, so any macros on the target Mac before you start syncing will be lost.**

⚠ **WARNING: If you make changes on one Mac, and then make changes on the other Mac before syncing has taken place, you may lose one or other changes.** Be especially careful if you are working with your Mac when offline.

NOTE: Keyboard Maestro references to files or applications that exist on one Mac and not the other will likely cause problems, so you should ensure any application you reference on one Mac is available at the same path on both Macs.

On the Mac that currently has your macros, choose the File ➤ Start Syncing Macros menu. **Read the text carefully**, and then click the Create New button. Save your existing macros in the macro sync file in your shared location. From now on, Keyboard Maestro will sync any changes to/from that file.

Wait for the file to be mirrored to the second/target Mac. On that second Mac, choose the File ➤ Start Syncing Macros menu. Again, **read the text carefully**, and then click the Open Existing button.

⚠ **WARNING: All of the existing macros on this second Mac will be destroyed if you continue.** If you have any macros on the second Mac that you wish to preserve, export them first, and then after syncing is enabled, import them (and they will then be synced to your other Macs).

Select the mirrored sync file. Your existing macros will be replaced with the macros from your first Mac.

Repeat the process for any other Macs.

If there are some macros you do not want active on a Mac, you can configure any given Macro Group to be disabled on that particular Mac by turning on the Disabled on this Mac setting in that Macro Group.

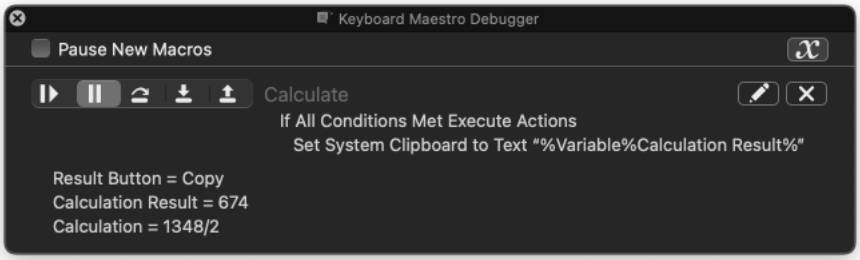
Dropbox may keep backup versions, and Keyboard Maestro keeps backup revisions (in the File ➤ Revert Macros menu), so you should be able to recover from any conflicts that happen. Dropbox may notice a conflict if you edit your macros on both Macs simultaneously. As a general rule this should not be an issue, though you may lose a change if you make changes on both Macs quickly (and note that quitting the Keyboard Maestro editor is considered a change).

Since your two Macs will likely not be identical, you may have to adjust your macros to work properly on both Macs. Using the various Tokens and Functions can help. For example the %MacUUID% token is a unique ID for each Mac, and can be used to test which Mac the macro is running on. Other functions, like the SCREEN function can be used to ensure your macro behaves appropriately regardless of the details of the Mac.

Note: Only your macros are synced. None of your preferences, clipboards or variables are synced.

Macro Debugger

Keyboard Maestro includes a built-in macro debugger which you can turn on by choosing the Status Menu ➤ Start Debugging menu or by using one of the Debugger actions. This will display the Macro Debugger window.



Once debugging, you can control whether new macros start paused or start running.

The macro debugger shows all running macros, and what action they are currently executing (including showing nested actions).

Using the buttons on the left, you can:

- **Continue** the macro – allowing it to run until completion, or until it hits a Debugger Breakpoint action.
- **Pause** the macro.
- **Step Over** the current action, including any subactions.
- **Step In To** the current action, stepping in to any included subaction.
- **Step Out Of** the current action and any other actions at the same level.

The buttons on the right let you:

- **Edit** the current action in the editor.
- **Cancel** the macro.

The button at the top right toggles displaying variables. The debugger will show the ten most recently modified (by this macro instance) variables and a summary of their values. If you want to show more information, you can use the Log action to log details to the Engine.log file, or use the Display Text action to display information in a window.

To debug a macro:

- Make sure the "Pause New Macros" is checked.
- Trigger your macro.
- Your macro will appear in the Debug window.
- Click on the icon/button with the down arrow to step into the macro to the first step.
- Click again to advance to further steps/Action.

All the debugger actions can also be done via the Debugger actions.

Keep in mind that once a macro has started executing, the engine has taken a copy of the macro to execute, so any changes you make in the editor will not affect the execution of the macro (although any changes you make to a sub macro that has not yet started executing would apply).

Also keep in mind that macros can often be time sensitive, so if you find your macro runs fine when stepped through in the debugger, but not when run normally, the issue is quite possibly that the macro is executing actions like click actions before the system has caught up and the screen is stable. Add an appropriate Pause action if that is the case.

If you close the debugger window, the paused macros will resume their normal operation.

Variables

Like most programming languages, Keyboard Maestro allows you to create Variables to store data for use later on in the same Macro, or in other Macros.

Variables can be set from many actions. You can set variables to specific tokenised text, to the result of a calculations, from user input, from the Keychain, by searching other variables, from the clipboard or Named Clipboards, as the result of scripts, and from many other sources.

Variable Naming Rules

Anywhere an Action requires a Variable, you can enter any name you like (even though often a default name is provided), as long as it conforms to these rules:

1. Variable names must start with a letter, and then can contain letters, numbers, spaces, or underscores.
2. Some Variable kinds require a specific prefix (see *Scope* below).
3. Variable names are case insensitive, but their case is remembered.
4. Variable names should not include a function or operator name with spaces around it.
    - (eg " MOD " , so "A MOD B" would not be a valid variable, although "MODULE" would be fine).

Scope

Keyboard Maestro includes variables with different breadth or scopes, in order of scope, with broadest scope first:

Scope	Availability
Global, permanent variables	• Accessible basically everywhere, including Scripts. • You can also see and edit variables in the Preferences. • You can see global variables in the Value Inspector.
Password, semi-permanent variables	• Name must begin or end with "PW" or "password" (case insensitive). • Available to macros but not displayed. • Not directly accessible via AppleScript. • Concealed in password fields in Prompt For User Input actions. • Not saved to disk.
Instance Variables, local to a specific macro execution instance	• Name must begin with "instance" (case insensitive). • Restricted to a specific execution sequence. • Available to the Macro where it was created and Sub-Macros of that Macro, for a given execution instance of the main macro. • These variables are private to each execution of the same Macro, even when running simultaneously. • Accessible by Prompt For User Input • Accessible by scripts (when used in an Execute Script Action). • Accessible by Custom HTML Prompt. • Accessible by AppleScript with the `instance` specifier.
Local Variables , local to a specific macro	• Name must begin with "local" (case insensitive) • Restricted to the specific Macro it is in.

Scope	Availability
	• *Not* available to any of its Sub-Macros • Same accessibility as Instance Variables.

Macros can create or read existing variables, which persist and are permanently stored (except Local and Instance variables which are transient, and Password variables which are never saved to disk).

Variables contain only plain, un-styled, text and may be used in any token text field in an action, as well as in Calculations in a numerical field. Variable may be used in Calculations if they contain a valid number or expression.

Kinds of Variable

Global Variables

Global Variables are permanently stored on your Mac drive (like a file), and are available for read/write in any Macro or Action, not just the Macro where the Variable was created. These variables persist between logins and restarts of your Mac.

Password Variables

Variables with names that start or end with "Password" or "PW" are considered passwords – their values will not be stored (except in memory) and they cannot be read directly by shell scripts, JavaScripts or AppleScripts, though their consequences can easily be extracted, eg with the Set Clipboard to Text action, so you should clear them as soon as they have served their purpose. The Prompt For User Input dialog will display such variables concealed in a password field.

Instance Variables

Variables with names that start with "Instance" (case insensitive, trailing space not necessary) are considered private to a specific execution sequence. Each time the macro is run, they will start empty but their value can be seen and changed by other macros within the same execution sequence (for example, if you execute a macro using the Execute Macro action, that macro can see and change the instance variable (unless you execute it asynchronously which would create a new instance)). Since two instances of a macro can be running at the same time, this is useful to ensure they each have their own version of the variable.

Since Local and Instance variables are transient, they are not shown in the Variables preferences pane.

Local Variables

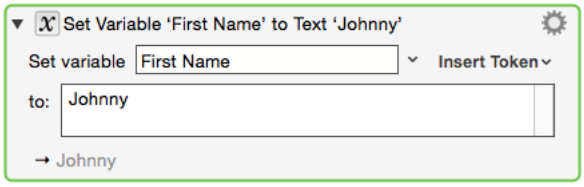
Variables with names that start with "Local" (case insensitive, trailing space not necessary) are considered local to a specific execution of a specific macro. Each time the macro is run, they will start empty and no other macro will see or be able to change values.

Note that *Local* and *Instance* Variables are available in scripts *only* when the script is called in an *Execute Script* action, unless you specify the execution instance explicitly.

Setting Variables

Variables are usually created and set by Macro Actions, but can also be set by scripts. The Action that sets a variable will create it if need be. Variables do not need to be declared in any way.

The most direction Action to set a variable is the Set Variable to Text action.



This action processes the tokens and backslashed characters in the text and sets the variable. As will all text fields, if you wish to use another variable in the text you must use the %Variable% token, eg `The value is %Variable%VarName%`.

You can also use the Set Variable to Calculation action. This action evaluates the numeric expression and then sets the variable to the result. As with all numeric fields, if you wish to use another variable in the calculation, you use the variable unadorned, eg `VarName * 3`.

There are many other actions that set variables.

Using Variables

There are many Macro Actions that can use variables. Some of these explicitly provide for entry of the variable name, but most provide for a more general text entry that accepts either token text or a numeric expression.

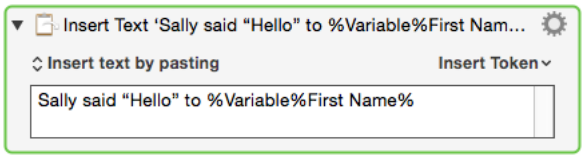
When editing a field:

- Fields that expect text tokens show a small T.
- Fields that expect calculations show a small C.
- Fields that expect just a variable name show a small V.

Variables in Text Fields

Variables can be used in text fields using the %Variable% token. The *%Variable%*`<VariableName>`% token allows you to include a variable in the text, where `<VariableName>` is the name of the variable, and this is replaced by the value of the variable.

For Example, using the Insert Text Action:



You can also use a short form of just *%Variable Name%* to include variables as long as the variable exists and has a value and there is no corresponding text token, although generally it is better and clearer to use the longer form *%Variable%Variable Name%*.

Variable Arrays

While technically all Variables are just strings, you can access a variable as if it were an array by setting the Variable to a delimited list of text, and then using the following notation:

```
%Variable%<VariableName>[<IndexNumber>]%
```

where

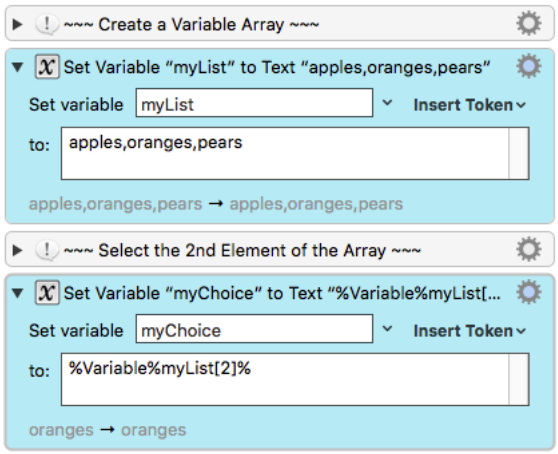
- "`<VariableName>`" is any valid Keyboard Maestro Variable with a delimited set of values.
  - By default the comma `,` is the delimiter (see below for setting a custom delimiter).
- "`<IndexNumber>`" is an integer indicating the index (starting with 1) of the array value within the Variable Array.
  - This index can be determined using any Calculation, including simply a Variable Name, like `i`.
  - Examples:
    - `%Variable%myList[i]%`
    - `%Variable%myList[3 * i + 2]%`

How to Use Custom Array Delimiter

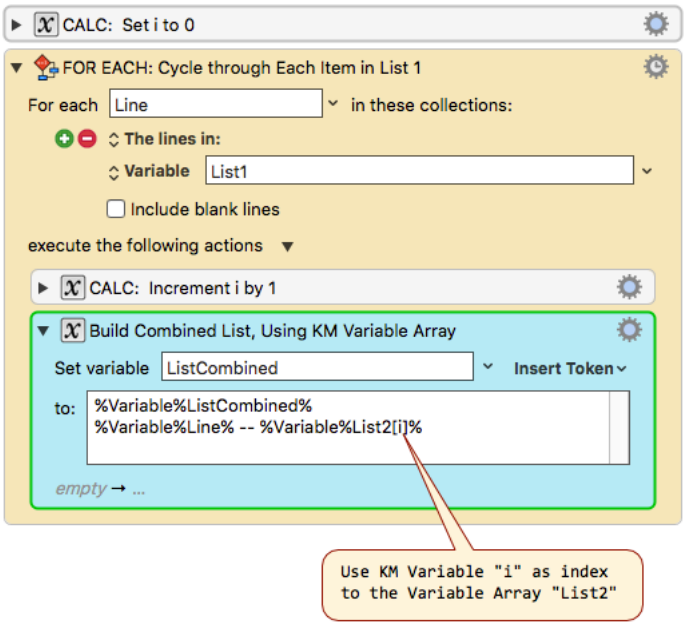
Place the *custom* delimiter *after* the *closing* bracket `]`.

- For Example: `%Variable%myList1[i]:%`, where the colon `:` is the delimiter used in the Variable Array.
- Of course, the Variable Array, *myList1* in this case, *must* use the same custom delimiter.

Simple Example of Variable Array



Example of Variable Array Using a Variable for the Index



Variables in Calculation Fields

Variables can be used in calculations if their value holds a number or a numeric expression that can be evaluated. Variables are used *unadorned* in calculations, for example `My Variable * 7`.

You can also access variables in a token field numerically using the %Calculate% token. **It is important to note that calculations can contain only numeric values**. When you use the %Calculate% token to reference a variable or variable array element, it will convert the element to a number.

Variable Dot Notation

In a Calculation field you can reference the numbers in a variable which contains a coordinate or rectangle (which is always a string, eg "100,100,200,200") using dot notation:

Variable.x	x coordinate
Variable.y	y coordinate
Variable.left	the left coordinate of a rectangle

Variable.top	the top coordinate of a rectangle
Variable.right	the right coordinate of a rectangle
Variable.bottom	the right coordinate of a rectangle
Variable.width	the width of a rectangle or size
Variable.height	the height of a rectangle or size
Variable.fuzz	the fuzz of an image match (rectangle,fuzz)
Variable.MidX	the horizontal middle of a rectangle
Variable.MidY	the vertical middle of a rectangle

The Variable Name and Dot reference are case insensitive.

Using Variables in Scripts

Variables values can be accessed from scripts you execute with Keyboard Maestro via environment variables, and from AppleScript using AppleScript commands to the Keyboard Maestro Engine, and from web browser JavaScript you execute with Keyboard Maestro via the document.kmvar dictionary, see the Scripting section for more details.

You can get and set Keyboard Maestro Variables (Global, Local, and Instance) in these types of scripts:

- AppleScripts
- JavaScript for Automation (JXA)
- JavaScript in Safari or Google Chrome (**Get only**)
- JavaScript in Custom HTML Prompts
- Shell Scripts (**Get only**)
- Swift Scripts

And you can generally access Keyboard Maestro variables anywhere else via AppleScript from whatever language you want to use.

Deleting Variables

- *Global* Variables continue to exist (remained stored) until their value is set to `""`, the empty string, even when the Keyboard Maestro Engine is quit and relaunched.
- *Password* Variables, continue to exist (remained stored) until their value is set to `""`, the empty string, or until you quit the Keyboard Maestro Engine.
- Setting a Variable to `%Delete%` does *not* technically delete it. It continues to exist. It is just *hidden* from view in all of the Variable lists and in the *Preferences*.
- However, a Variable set to `%Delete%` will behave as if it does *not* exist in most Actions, like a If Then Else Action using a *Variable Condition* comparing to *exists*.

It may be a good idea to set a Variable to `""` if you no longer need it, or if it contains sensitive information (like a password) or a large value (like a file or web page contents).

Perhaps the best way to avoid variable clutter and eliminate sensitive and/or large values is to use either *Local* or *Instance* Variables when you do not need the Variable after completion of the Macro in which the Variable is used.

Inserting In Actions

You can add a variable to an action using the Edit ➤ Insert Variable menu or Edit ➤ Insert Function ➤ Variable menu or Edit ➤ Insert Token ➤ Variable menu, or by selecting from the popup menu next to some variable fields.

Fields that expect just a variable show a small V in the field while editing it, and you can use Type Completion to complete variable names in such fields.

View Your Entire List of Variables

You can add, delete, see or change global variables in the Variables preference pane that have been created by your macros and scripts.

Local and Instance Variables are *not* accessible from either the Preferences Window.

Watching a Variable's Value

You can use the Value Inspector to watch the value of a global variable.

Tokens

Keyboard Maestro provides two means to get information about objects (like web pages and windows) in the user's environment:

1. Tokens (returns text, used in text fields)
2. Functions (returns numbers, used in numeric fields)

You can enter a Token in any text field in an Action, and when the Macro is triggered and the Action is executed, the token will be replaced by the value that is returned at run time. This allows you to easily combine static text with dynamic data that is supplied at run time, in most any Action that has a text field, like Display Text and Set Variable to Text.

Token Format

Tokens are identified by a pair of percent `%` symbols at the start and end of the Token name.

Usage	**Format**	**Example**	**Token Results**
No Parameter	`%TokenName%`	Safari open to this web page: https://wiki.keyboardmaestro.com/Home_Page [https://wiki.keyboardmaestro.com/Home_Page]	%FrontBrowserTitle% returns: "Home Page [Keyboard Maestro Wiki]"
With Parameter	`%TokenName%Parameter%`	Frontmost Window Size	%WindowSize%1% returns: "1991,1417"

Entering Tokens into Actions

The available tokens, and their definitions, are listed on the Token List page.

When you are editing a Macro, put the text cursor in a text field, and do one of the following to select and insert a Token into the text field.

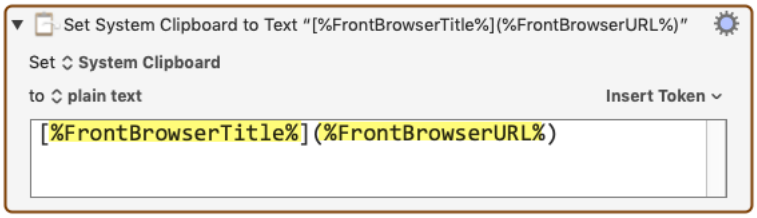
1. Press ⌘^T (or use the Menu) to display a prompt to select the Token by Name
2. GoTo menu Edit ➤ Insert Token menu
3. Click on the "Insert Token" popup at the top right of the text fields to see the same menu as in #2.
4. Of course, you can always just type the Token.

You can tell that a text field accepts tokens because a small T shows in the field while editing it.

Examples

Let's look at a real-world example. Let's say you would like to set the Clipboard to a Markdown Link [https://www.markdownguide.org/basic-syntax/#links] of the current web page you are reading. All you need to do is insert these two tokens (using the Markdown format) in a Set Clipboard Action:

1. `%FrontBrowserTitle%`
2. `%FrontBrowserURL%`



The "`[]()`" are just characters you type.

After the macro and Action are executed, you will see this on the Clipboard:

That's all you need for a great macro, which you can download from here [https://forum.keyboardmaestro.com/t/set-clipboard-to-markdown-link-of-frontmost-browser-wiki-example-macro-v9-0-1d2/15025].

Processing

The Tokens are processed, that is replaced by their value at run time, when the Macro is run, and the Action containing the Token is executed.

In most text fields, you can control how Tokens are processed using the Gear ⚙ popup menu:

- Process tokens normally, or
- Just tokens (not backslashed characters) or
- Do not process any text.

Other Uses for Tokens

Non-Printing Control Characters

An exception to the rule of enclosing Tokens in percent `%` characters, is the use of these Non-Printing Control Characters `\a,\b,\e,\f,\t,\r,\n` which correspond to (bell,backspace,escape,form feed,tab,return,line feed). These will be replaced with their value in text fields, except for Regular Expression (RegEx) fields.

Convert Hex to Unicode Characters

Text token fields also process %NN% or %NNNN% or %NNNNNN% as arbitrary hex unicode characters (eg %41% is an A, %01F300% is 🌀).

Using Special Characters as Plain Text

To include a percent in your text, simply double the percent (%%). To include a backslash \ in your text, double the backslash (\\).

Text Case Conversions

Text fields also support text case conversion using the below meta characters. You may use these in both standard Text Fields and in the Replace field of a Search and Replace by regular expression action (even though these meta characters are *not* supported by ICS Regular Expressions).

The Available Case Conversion Meta Characters are:

- `\U` converts everything up to the next `\L` or `\E` to uppercase.
- `\L` converts everything up to the next `\U` or `\E` to lowercase.
- `\u` converts the next character to uppercase.
- `\l` converts the next character to lowercase.
- `\U\l` lowercase first, then uppercase.
- `\L\u` uppercase first, then lowercase.
- `\E` stop changing case.

You should not use `\u` after `\U` or `\l` after `\L` unless you terminate the sequence with `\E` first.

Access to an Item in Token Results

With tokens that return a list of information (comma separated by default), you can access the individual items using the Variable array notation, append the item number in square brackets `[ ]` to the Token name, just before the closing `%`.

For example:

Token for Window Frame, %WindowFrame[3]%1%, will be the third comma-separated value, which is the Window width in this case.

- %WindowFrame%1% returns "3606,23,1514,1417"
- %WindowFrame[3]%1% returns "1514"

These tokens allow this indexing:

- TriggerValue
- FrontWindow*
- Window*
- Screen*
- SystemClipboard
- PastClipboard
- NamedClipboard
- TriggerClipboard
- CurrentMouse
- FoundImage

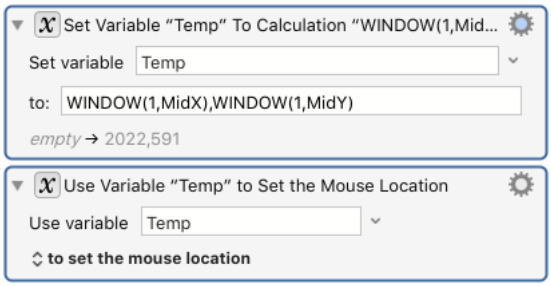
Calculations

Overview

Keyboard Maestro supports calculations in almost any numeric text fields. For example you can Pause for "60*Time in Minutes". Calculations can also use comma separated lists of numbers as arrays, and can return such arrays, so you can operate on rectangle frames and points. Numeric fields often start small with up/down step arrows, but if you type anything other than a number they will expand to allow a more complex expression to be entered. You can tell that a field accepts calculations because a small C shows in the field while editing it.

Examples

For example:



will result in the mouse being placed at the center of the front window.

Note: You must use commas for this purpose, and full stops (.) for decimal numbers, and never use any thousands separators, regardless of your desired language.

Expressions

Keyboard Maestro's expressions include precedence, nested bracketed expressions, many built-in functions, various numeric bases, so you should be able to write almost any expressions you might like to use, as well as use it as a general purpose calculator if desired.

Operators

Operators based on precedence from lowest to highest are:

Array Separator (`,`)	Allows a text Variable, which has comma separated values, to work somewhat like an Array. Assume `MyKMVariable` contains "value1,value2,value3", then `MyKMVariable[2]` returns "value2".
Ternary Operator (`?`)	a=b ? 3 : 4
Bitwise OR (`|`), Bitwise AND (`&`) and Bitwise XOR	bitwise operators, which also act as logical boolean operators for 0 (false) and 1 (true).
Comparison Operators (`<, ≤, =, >, ≥, ≠`)	compare for (in)equality and return 0 or 1.
Shift Operators (`«, »`)	shift a number left or right.
Arithmetic Addition Operators • Add (+) • Subtract (–)	Basic mathematical operations.
Arithmetic Multiplication Operators • Multiply (`*`) • Divide (`/`)	Basic mathematical operations.
Modulo [https://en.wikipedia.org/wiki/Modulo_operation] (`i MOD n`)	The remainder of the division of `i` by `n`. Both values (`i`, `n`) will be treated as integer. `n` must not be 0. the sign of the result is the same as the sign of `i`.
Integer division (`i DIV n`)	The integer quotient of the division of `i` by `n`. Both values (`i`, `n`) will be treated as integer. `n` must not be 0. the sign of the result matches the sign of regular division.
Power Operator (`^`)	exponentiation.
Unary Prefix Operators (√, –, `( )`)	square root, negation, sub-expressions.
Functions	a large variety of functions.
Numbers and Variables or Array Accesses (5,$5A,0x50,8#007,Variable,Variable[5])	identifiers and values.

Unary Postfix Operators (`!,%,°`))	factorial, percent, degrees.

Numbers are in decimal by default, but may use base 16 if they start with `$` or `0x` (eg $5A or 0x5A), or may start with a specific base followed by a `#` (eg 8#007).

Variables can be used if they contain numeric expressions, including an *array* of numbers separated by commas, in which case you can use an array index to select the desired number.

You can use either = or == for testing for equality.

Operators and functions must be in uppercase to minimize conflict with variables.

Functions

The available functions are listed on the wiki Functions page.

You can insert a function by name by choosing from the Edit ➤ Insert Function menu or choosing the Edit ➤ Insert Function by Name menu.

Variables

In numeric calculation fields, Variable Names are used without the `%` that are used in text token fields. Do not try to use tokens (like `%Variable%MyVar%`) in numeric calculation fields, just use MyVar by itself. The variable must contain a valid numeric value, or an expression that evaluates to a valid numeric value. So for example, if MyVar contains a text value of `2*3`, then the calculation `4*MyVar` will return 24.

Screen Coordinates

Keyboard Maestro refers to screen coordinates as two or four comma separated numbers in the text of a Keyboard Maestro Variable (which is always a string).

- Screen object points, like the left,top position of a window, have two values, like 12,34
- Screen object rectangles, like the frame of a window, have four values 12,34,56,78 (with a fifth value for fuzz in some instances).
  * You can reference these values by position in the string, as it the Variable were an array:
- For example:
  if the Variable `myWindow` is "12,34,56,78"
  then all of these forms of reference will provide a value of 34 in a Calculation field:
  - myWindow[2]
  - myWindow.y
  - myWindow.Top

Variable Array Access

If a variable contains a sequence of numbers separated by comma (,) then you can access that variable using array notation (eg `MyVar[5]`). So if variable MyVar has a text value of `10,20,30,40,50,60`, MyVar[5] will return 50.

The index is itself an expression, so it can be arbitrarily complex.

Indices are 1-based, so MyVar[1] is the first element. If the index is 0, the size of the array is returned (so MyVar[0] would be `6`). If the index is less than zero, the array is indexed from the end (so MyVar[-5] would be `20`).

Variable Dot Notation

In a Calculation field you can reference the numbers in a Keyboard Maestro Variable (which is always a string) using dot notation:

Variable.x	x coordinate
Variable.y	y coordinate
Variable.left	the left coordinate of a rectangle
Variable.top	the top coordinate of a rectangle
Variable.right	the right coordinate of a rectangle
Variable.bottom	the right coordinate of a rectangle
Variable.width	the width of a rectangle or size
Variable.height	the height of a rectangle or size
Variable.fuzz	the fuzz of an image match (rectangle,fuzz)
Variable.MidX	the horizontal middle of a rectangle
Variable.MidY	the vertical middle of a rectangle

The Variable Name and Dot reference are case insensitive.

Text Fields

In calculation fields, you can express a calculation as you would normally write an expression, for example:

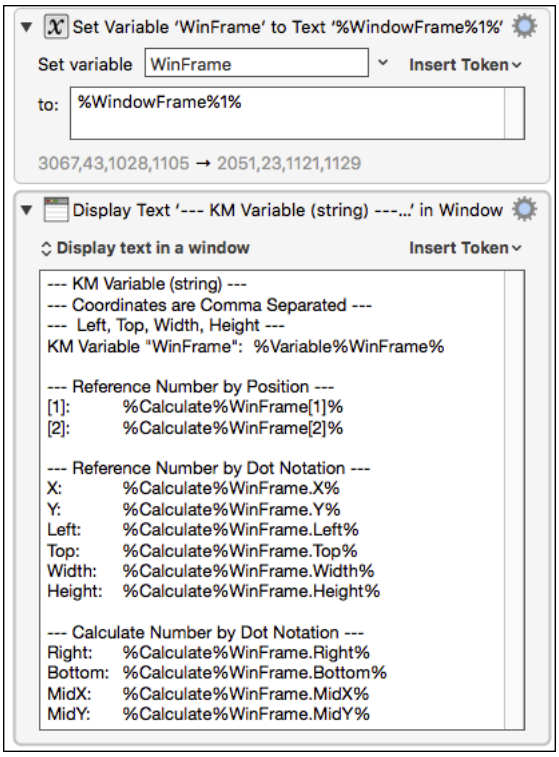
```
3 * Count + 7
```

However in a text field, since any text is allowed, you must use percent encoded tokens to indicate where more processing is required. You can include a variable in the text by using the %Variable% token, or you can use a calculation by using the %Calculate% token, or any number of other Tokens.

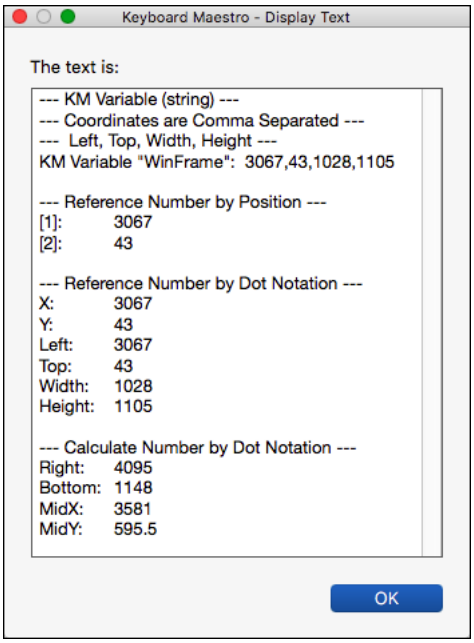
```
The result is %Calculate%3 * Count + 7%.
```

Examples of Variable Data Reference

Macro Actions



Example Results



Some example expressions are:

```
Amount in Dollars * 100
MJD() > 55928
NOW() > TIME(2012,3,23,12,2,1)
DOW(TIME(2012,4,4)) = 4
Radius*SIN(20°),Radius*COS(20°)
Window Frame[1]+Window Frame[3]/2,Window Frame.MidY
MOUSEBUTTON() + 2 * MOUSEBUTTON(4)
SCREEN(Internal,Left,10%)
```

Conditions

Keyboard Maestro includes a variety of control flow actions which perform actions depending on a set of conditions.

The condition clause of the flow control actions can be any of:

- Any of the following are true – at least one condition must be true.
- All of the following are true – every condition must be true.
- None of the following are true – no condition is true.
- Not all of the following are true – at least one condition must be false.

This is followed by a set of specific conditions. **If there are no conditions in the set at all, the action will not execute anything** except the Until action which will execute the actions once. Neither side of the If Then Else will execute.

The available conditions are listed on the wiki Conditions page.

Collections

Keyboard Maestro includes a For Each action which perform a sequence of actions repeatedly, once for each element of a collection.

Any time you have to deal with a set of things (lines, files, numbers, etc), you are probably thinking of a collection for which the For Each action is the answer.

The available collections are listed on the wiki Collections page.

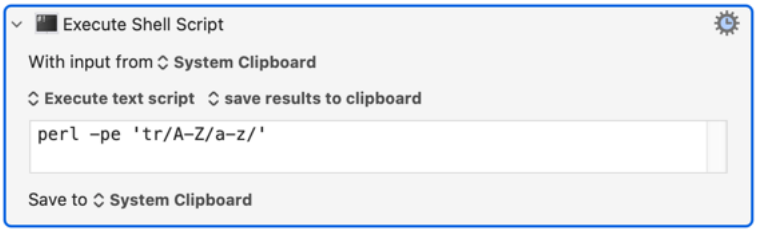
Filters

Keyboard Maestro includes a variety of filters that can be applied to either the system clipboard, Named Clipboards, or variables. For clipboards, the filters preserve style information to whatever degree is possible. Use the Filter action to apply filters.

The filters are:

- Remove Styles (clipboards only, make the clipboard plain text).
- Set line endings to Mac (CR), Unix (LF) or Windows/DOS (CRLF).
- Trim Whitespace. (Remove leading and trailing whitespace from a string.)
- Hard wrap or unwrap paragraphs.
- Lowercase (all characters), Lowercase First (just the first character).
- Uppercase (all characters), Uppercase First (just the first character).
- Capitalize (all words) or Title Case (intelligently uppercase certain first letters).
- Change quotes to Smart, Dumb or French quotation marks.
- Encode HTML or non-ASCII HTML entities.
- Encode HTML with numeric entities.
- Decode HTML entities.
- Generate an HTML list.
- Percent Encode [https://developer.mozilla.org/en-US/docs/Glossary/percent-encoding] or Decode a URL. Percent Encode will encode **all** non-alphanumeric characters.
- Encode for Regular Expression.
- Encode or decode Base64.
- Calculate MD5.
- Quote for AppleScript, JavaScript, Shell (bash) Script, Swift, JSON or Process Tokens.
- Convert a JSON object string to Compact or Pretty format.
- Get parent of a path.
- Get the filename component of a path.
- Get the basename of the path (ie the filename without directory or extension).
- Get or delete the path extension.
- Get the display name of a path.
- Expand tilde (~) paths, or abbreviate with a tilde.
- Resolve symlinks, or standardize the path.
- Get the URL scheme, host, port, user, password, path, fragment or query.
- Sort, reverse sort, or shuffle lines.
- Delete or bullet (•) control characters.
- Calculate an expression and return the result, see the Calculations section.
- Process Text Tokens and return the result, see the Tokens section.
- Get the value of a named Variable or Named Clipboard.
- Count the characters, words or lines and return the result.

We will likely expand the list of possible filters, so if you have specific filtering needs that you think might be of general interest, please let us know. You can also process text using an AppleScript or shell script, for example the shell script:



is roughly equivalent to the Lowercase filter, except that it only works with ASCII characters.

Dictionaries

Keyboard Maestro includes permanently stored dictionaries that you can use or set.

A dictionary is a named set of mappings from a key name to a value. You can have multiple dictionaries, each with their own name (so really, it is a set of mappings from a dictionary name and a key name to a value).

Dictionary names start with a alphabetic character, followed by any number of alphanumeric, space or underscores. Dictionary names are case insensitive.

Key names can be anything, although leading and trailing white space is stripped off. Key names are case insensitive.

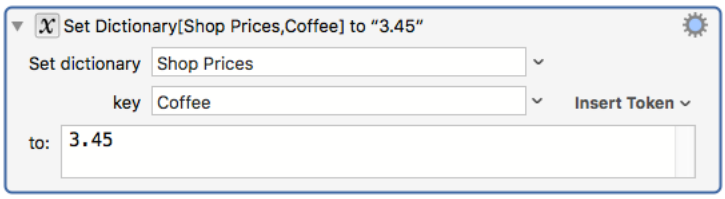
Dictionary values are plain text and can include leading or trailing white space.

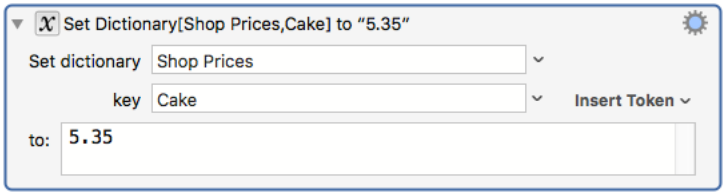
For example, you might have a dictionary named "Shop Prices", with keys being the item names and values being the cost of the item.

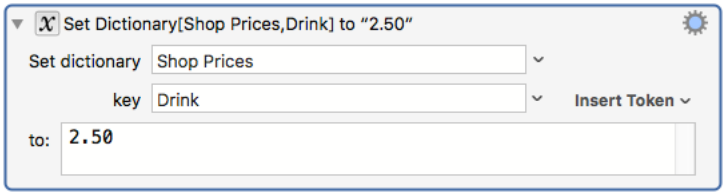
Dictionary	Key	Value
Shop Prices	Coffee	3.45
Shop Prices	Cake	5.35
Shop Prices	Drink	2.50

Setting Dictionary Values

Use the Set Dictionary Value action to set a dictionary value.

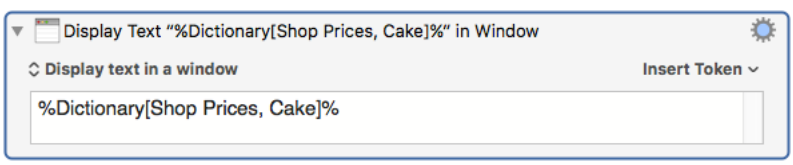






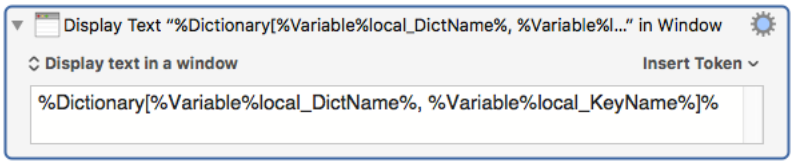
Accessing Dictionary Values

You can use the %Dictionary% token to access Dictionary values.




```
%Dictionary[Shop Prices,Cake]%
```


```
%Dictionary[Shop Prices,%Variable%local_KeyName%]%
```




```
%Dictionary[%Variable%local_DictName%,%Variable%local_KeyName%]%
```

You can list the dictionaries with the Dictionaries collection, and you can list all the keys within a dictionary with the Dictionary Keys collection.

AppleScript

You can read and write dictionary values from AppleScript.

snippet.applescript

```
tell application "Keyboard Maestro Engine"
        set dictionaryNameList to name of dictionaries
end tell
```

snippet.applescript

```
tell application "Keyboard Maestro Engine"

   set newDict to make new dictionary with properties {name:"Shop Prices"}

   # If you don't set a key/value pair the newly created dictionary above goes up in smoke.
   tell newDict
      make new dictionary key with properties {name:"Coffee", value:"3.50"}
   end tell

end tell
```

snippet.applescript

```
tell application "Keyboard Maestro Engine"
        set dictKeyList to dictionary keys of dictionary "Shop Prices"
end tell
```

snippet.applescript

```
tell application "Keyboard Maestro Engine"
        set value of dictionary key "Coffee" of dictionary "Shop Prices" to "4.50"
end tell
```

See Also

See also the Set Dictionary Value action, the %Dictionary% token and the Dictionaries and Dictionary Keys collections.

JSON

Keyboard Maestro includes comprehensive support for JSON [https://json.org] (JavaScript Object Notation), which is a formalized lightweight data-interchange format.

JSON objects consist of one of:

- An array, surrounded by square brackets and with comma-separated values, eg `[ 1, "2", 3, "4" ]`.
- A dictionary (aka, object), surrounded by curly brackets with comma-separated key-value pairs, eg `{ "one" : 1, "two" : "2" }`. The Key must be a properly formatted, double quoted, string.
- A double quoted string, eg `"hello"`. Strings can contain various backslash-escaped characters, including `"`, `/`, `\`, `b` (backspace), `f` (form feed), `n` (linefeed), `r` (carriage return), `t` (tab), and `u` followed by four hex digits to make a unicode character.
- `true` - boolean true
- `false` - boolean false
- `null` - null (missing or absent value)

JSON terminology refers to dictionaries as "objects", but also the whole thing as a JSON object, so generally within Keyboard Maestro it will be referred to as a JSON dictionary when talking about a specific dictionary formatted part of a JSON object, and object when it could be any of the various forms. A JSON container would be an array or a dictionary.

JSON Paths

Keyboard Maestro introduces the concept of a JSON Path. This is similar to the concept of an XPath, and there are other similar but unrelated JSON path concepts around, and they have similar intentions, but none of these are the same, or have the same syntax.

In Keyboard Maestro, for a variable named `J`, and JSON Path might look like:

```
J[2+3].fieldname{%Variable%FieldName%}
```

This would assume that variable J holds a JSON array, and would extract the fifth element. Then it would assume that contains a dictionary and look for the field named "fieldname". Within that, it would assume that contains a dictionary and look for a field named by the variable FieldName.

After the variable name, the format of the path is any of:

- "[" followed by a numeric calculation, followed by "]". The result of the calculation will be used to look for an indexed element in the array, or for a matching named element in a dictionary.
- "{" followed by token text, followed by "}". The result of the text will be used to look for a matching named element in a dictionary, or for an indexed element in an array.
- "." followed by the field name. Field names are trimmed of leading or trailing whitespace - if your object field names can start or end with white space, you need to use the `{ field name }` notation.

Array indices are 1-based. Negative indices count from the end of the array. The 0 index of an array access will return the count of the number of entries in the array. When evaluating the index, the context sensitive COUNT() function will return the number of elements in the array.

Keyboard Maestro will use arrays and dictionaries more or less interchangeably, so:

- J[3]
- J.3
- J{3}

will all return the third element of an array, or the dictionary value for key "3". And important distinction is that the first one is a numeric calculation, the second one is plain text, and the third one is token text.

Pretty, Compact and Strict Flags

Many of Keyboard Maestro's operations that produce a result from JSON have two flags associated with them: Compact or Pretty; Strict or Not-Strict.

- Compact means all extraneous white space is omitted from JSON results
- Pretty means JSON results will include white space to format the JSON in a relatively readable manner.

For example, if the variable J contains the text `{ "a" : 3, "b" : null }`, then the result of `%JSONValueCompact%J%` will be:

```
{"a":3,"b":null}
```

whereas the result of `%JSONValuePretty%J%` will be:

```
{
  "a" : 3,
  "b" : null
}
```

The exact indentation should not be assumed, it might be any number of spaces or tabs.

The Strict flag relates only to JSON results that are trivial JSON strings, eg JSON `"result"`.

- Strict means even for trivial JSON strings, the result will be a properly double quoted string (eg `"result"`.
- Non-Strict means that for trivial JSON string results, the result will return just the resulting string without any double quotes and with any escaping removed/evaluated. This will be more useful in some situations, but will lose the ability to distinguish between strings, numbers and `null`.

For example, if the variable J contains the text `{ "a" : 3, "b" : "hello\n\"there\"\n" }`, then the result of `%JSONValueStrict%J.b%` will be:

```
"hello\n\"there\"\n"
```

whereas the result of `%JSONValue%J.b%` will be:

```
hello
"there"
```

The Strict/Non-Strict flag will have no affect unless the result is a trivial JSON string - if the result is a JSON object or array, or a number or `true`, `false`, or `null`, the result will be the same either way - objects and arrays will be inherently strict, and numbers and true/false/null will be unadorned.

The default result format is Compact, Non-Strict.

Use

There are actions to set dictionaries or variables from JSON containers, as well as an action to set a field within a JSON object to a specified value.

Notes

JSON objects must be properly legal to be processed. In particular keys to objects must be strings, which means strings must be quoted with double quotes. `{ a : 3 }` is not valid, it must be `{ "a" : 3 }`.

Although `3` and `"a"` are technically legal JSON, in Non-Strict mode with cases of ambiguity, Keyboard Maestro will not consider a value or variable to contain JSON unless it is a container, ie an array or dictionary, which starts (without any leading white space) with a `{` or `[`, and ends with a matching bracket followed by optional white space.

Search Strings

Search Strings are filters which, in addition to the text you type, can include qualifiers that may be applied to searches for Macros and Actions in:

1. Keyboard Maestro Editor, Search field
2. Keyboard Maestro Editor, Smart Group
3. Trigger Macro by Name Action

Keyboard Maestro will search for the text you type, and all of the below qualifiers that you specify, for matches in your list of macros. Each word is searched for separately (and in any order) unless you quote a phrase.

Note that the Trigger Macro by Name Action only searches among active macros.

Search Qualifiers:

The qualifier can generally be shorted to its shortest unique name, as shown in the *Short Form* column.

Qualifier	**Short form**	**Definition**	**Example**
all:	al:	Match everything	`al:` (Otherwise the empty search string matches nothing)
group:	gr:	Match any macro within the named macro group (Use quotes if Name contains spaces)	`gr:"Forum Examples"`
global:	gl:	Match any macro within any macro group that is globally active	`gl:` (aka universal:)
application	ap:	Match any macro that is specific to (or excluded from) the specified application	`ap:Finder`
enabled:	e:	Match any macro that is enabled	`e:` (not necessarily active)
disabled:	d:	Match any macro that is disabled	`d:`
trigger:	t:	Match any macro that has a trigger matching the specified string	`t:;sig` (Typed String of ";sig")

hotkey:	h:	Match any macro with specified HotKey
		Match any macro with any HotKey if no HotKey is specified.
		Modifier Keys must be one of these: `^⌥⇧⌘`
			`h:⌘^C`
			(CMD CTRL C)
			
			`h:`
			(Matches any HotKey)

name:	n:	Match any macro that has a name matching the specified string	`n:PDF`
created:	cre:	Match any macro that was created less that the specified time ago	`cre:1w`
modified:	mod:	Match any macro that was modified less that the specified time ago	`mod:1d`
used:	use:	Match any macro that was used less that the specified time ago	`use:1m`
size:	siz:	Match any macro that has a storage size more than the specified size	`size:10000`
note:	not:	Match any action that contains the text in a Note	`note:custom`

For the created:, modified:, used:, the parameter is a number followed by a letter (s for seconds, m for minutes, h for hours, d for days, or w for weeks).

You can use a negative sign (-) to negate any match (for example, "-name:PDF").

Palettes

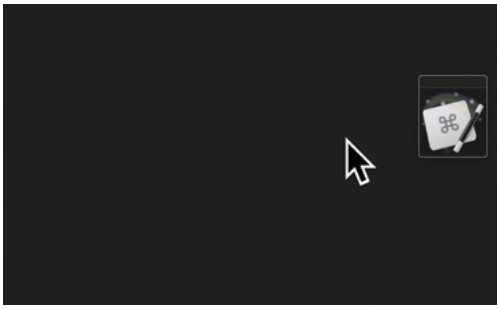
Keyboard Maestro can display a number of Palettes (or Toolbars).

There are four types of palettes:

1. Global Macro Palette
2. Applications Palette
3. Conflict Palette.
4. Macro Group palettes

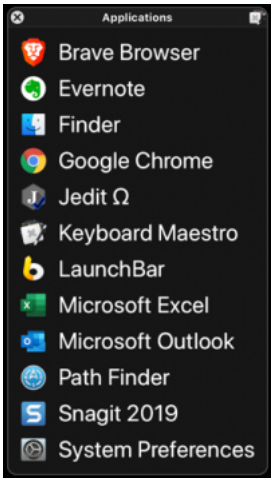
1. Global Macro Palette

Any macro with the Global Macro Palette trigger will be displayed in the *Global Floating Palette*. This palette appears whenever there is one or more active macros with this trigger, so it may appear and disappear depending on which macros are active (remember that Macro Groups control when a macro is active).



2. Applications Palette

The Applications Palette shows the currently running foreground applications.



You can show/hide the Applications Palette in the General preference pane, or by choosing the Status Menu ➤ Show Applications Palette menu.

You can drag files on to the palette icons, and you can control click on them to perform various actions like hiding or revealing in the Finder. You can also hold various modifier keys down while clicking to perform the actions.

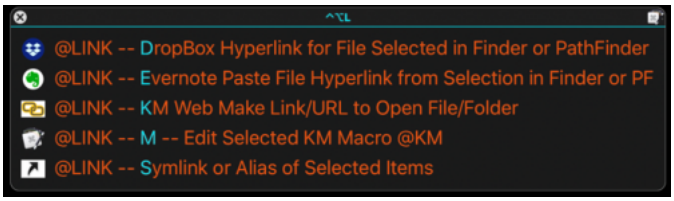
Applications can be excluded from the palette by selecting *Exclude from Applications Palette* in the contextual menu, or by adjusting the excluded applications in the Exclude preference pane.

You can order the applications alphabetically or by launch date by selecting *Sort by Launch Date* in the contextual menu.

3. Conflict Palette

When you use the same trigger (such as a hot key, device key, or typed string trigger) that is assigned to more than one macro, and all of these macros are active, then Keyboard Maestro will display the Conflict Palette listing the triggered macros and let you select the desired macro, either by clicking on it or by typing a letter that narrows that palette down until only one macro

remains at which point that macro is triggered.



You can use the Conflict Palette to limit the number of hot keys you need to remember by give a bunch of similar or related macros the same hot key and then selecting the desired macro with a further keystroke based on the now visible palette.

4. Macro Group Palette

Any Macro Group can be configured to display as a palette. Since Macro Groups can be configured to be active *only* in specific applications, you can design Macro Group to display its palette only when a specific application(s) is frontmost.

See Macro Group Setup Criteria (Advanced) for the various options for displaying palettes.

Control Order of Macros on Palette

You can control the order of macros in a macro palette (or the status menu) by prefixing their name with two characters and a closing parenthesis (eg "01)" - two characters and a closing bracket). The macros will be sorted based on the code, but the code will be stripped off before display in the palette (or status menu).

For a great tool to aid in configuring the Palette for a Macro Group, see the **Palette Organizer [https://forum.keyboardmaestro.com/t/macro-palette-organizer-v1-2-updated/6088]** macro on the forum.

Controlling The Display of Palettes

Display More than One Palette

To have more than one macro palette, create a macro group for each desired palette and configure it to show a palette as desired. Put your macros in there. Create as many of these as you like. You can configure the Macro Group so that the macros are only active while the palette is displayed, so if you only display it occasionally, especially only for one action, then they can have very simple hot keys (like plain letters for example).

Nested Palettes

By using the Show Macro Group or Show Macro Group for One Action actions, you can display a macro group as a palette. This allows you to build nested palettes. For example, a palette could contain a number of macros that simply show different macro groups.

Enabling and Marking Macros

By using the Mark or Unmark Macro action, you can have macros shown in macro palettes appear ticked to indicate some state (for example an application dock that marks or unmarks the macro as the application launches or quits). Alternatively, you can enable and disable macros to have them appear or disappear from a macro palette. Either of these techniques will allow some level of dynamic behavior of palettes.

Appearance and Style

You can control the appearance of the palettes in the Palettes preference pane or in the macro group configuration. You can choose the style, the opacity of the palette, the size of the entries, the number of columns, whether the entries include the icon, the text and/or the trigger, the title bar, and whether the palette shrinks when the mouse is not over it. You can use the styling to build palettes that just display their icon for example, and combined with custom icons for your macros this can create a nice looking icon palette.

Recording

Keyboard Maestro has the ability to create macro action sequences by recording your actions.

For example, to create a macro that simulates keystrokes, rather than create each macro action individually you can enable recording and then simply type the keystrokes.

Keyboard Maestro can record the following actions:

- Moving a window
- Resizing a window
- Miniaturizing a window
- Clicking the mouse
- Typing a Keystroke
- Moving the scroll wheel
- Selecting a menu
- Activating an application
- Quitting an application

There are two ways you can use recording: when creating or editing a macro, or via a Record Quick Macro action.

When you are creating or editing a macro, with the Macro Editor window displayed, simply click the Record button. After a short pause for you to get ready, recording will begin. To avoid the pause, hold the option key down while clicking the Record button).

Once recording starts, demonstrate the task you would like to perform using any of the above actions and Keyboard Maestro will record your actions directly into your macro.

While Keyboard Maestro is recording, it will display the Recording window.

While you are recording, you can pause the recording by clicking the Pause button in the recording window.

Normally, Keyboard Maestro does not record pauses between actions, so the macro will play back at fast speed. However, you can add a 0.25 second pause to your macro by clicking the Clock button, or you can option-click the Clock button to turn on "real time", and Keyboard Maestro will record a pause between each action which will simulate playback at approximately the same speed as you recorded.

When you are finished, click the Record button again to stop recording, or you can stop all recording by clicking on the Recording window.

Typically you will need to make a few adjustments to the Macro Actions to ensure the macro will operate robustly when used. Generally, use recording to create a base sequence of actions and then adjust as necessary.

The other way to use recording is via a Record Quick Macro action. When triggered, the Record Quick Macro immediately starts recording your actions into a private macro. When you have demonstrated the sequence of actions you want, trigger the Record Quick Macro action again. The sequence can now be executed via the specified Hot Key or the Status Menu or Macro Palette. For example, if the Record Quick Macro is triggered by pressing Control-F1, and the specified Hot Key is Option-F1, then if you typed:

Control-F1, h, e, l, l, o, Control-F1

Then each time you press Option-F1, Keyboard Maestro will type "hello" for you. One common use for this is if you want to adjust a sequence of lines in a systematic way. For example, if you had a list of colors, and wanted to change them in to a list of constants, say from this:

```
color Red
color Green
color Blue
```

to

```
const int kRed = "Red";
const int kGreen = "Green";
const int kBlue = "kBlue";
```

You could do this with grep and regular expression, replacing "color (.*)" with "const int k\1 = "\1";", which is fine if you can remember how to do grep with regular expressions, whether it is \1 or $1, and whether the application you are in supports regular expressions or not. But perhaps a simpler way is to just show Keyboard Maestro how to do the first line and then let it do the others with a single keystroke each.

So move the cursor to the start of the first line, press Control-F1, then the sequence:

```
Option-Shift-Right Arrow, Delete, Forward Delete, Command-Shift-Right Arrow, Command-X, c, o, n, s, t,
```

Finish with Command-Left Arrow, Down Arrow to carefully put the cursor at the start of the next line. Now press Control-F1 again to finish the recording, and Option-F1 twice to translate the next two lines.

Record Quick Macros can record the same set of actions that normal recording can, however because you cannot see or edit the recorded actions it is wise to keep them simple, preferably just a sequence of keystrokes. Typically, recorded Quick Macros will be used immediately and not reused, but they are saved and remain available until you record over them.

Macro Library

The macro library is a place where we can provide you with a variety of ready-made macros for optional addition to your macro collection.
To use the library, choose the Window ➤ Macro Library menu to display the macro library. You can then look through the available macros and insert any you'd like to use into your macros. You can then use them as is, or configure the new macros, perhaps changing the hot keys or adjusting the macros to your liking.

Each macro comes with a short description to tell you what it does, so scroll through them to see all the possibilities, and click on them to get more details.

You can also download new potential macros from us or from friends or colleagues or the forum [https://forum.keyboardmaestro.com]. You can also share your own macros with other Keyboard Maestro users by exporting your clever macros.

Keep in mind that macros can do practically anything on your Mac, including cause a huge amount of damage, so you should never execute a macro without verifying the source and better yet, checking exactly what it does.

You can export a set of macros to a macro library file, which you can share with others, and you can import .kmlibrary files into your macro library. Note that currently there is no way to delete imported macro library entries from your Macro Library except by quitting Keyboard Maestro and Keyboard Maestro Engine and removing the files from the ~/Library/Application Support/Keyboard Maestro/Keyboard Maestro Libraries folder.

Generally, for macros you want to share, it is better to export them as macros, or use the Share button at the top the editor window, or choose from the File ➤ Share menu.

For some other example macros, you can also look at the Macro Library on the wiki.

Macro Examples

A typical simple Macro consists of a single Hot Key trigger, such as Control-A, together with a single action, such as the Insert Text by Typing Action that will "type my address" .

For some real examples, see the forum topics:

- List of Example Macros [https://forum.keyboardmaestro.com/tags/example]
- Best Macro List [https://forum.keyboardmaestro.com/t/best-macro-list]
- Best Examples of Keyboard Maestro Macros [https://forum.keyboardmaestro.com/t/best-examples-of-keyboard-maestro-macros]

You can also use the Macro Library by choosing the Window ➤ Macro Library menu to see some built in examples.

Here are a number of example and suggestions for Macros to give you some ideas of how you can get the most out of Keyboard Maestro and your Mac. For tips on how to remember which Hot Key executes which action, see the Remembering Macro Hot Keys section.

Launch Your Most Used Applications

Use function keys to launch or switch to your most used applications. For example, you probably often switch to the Finder, your Email client, your Web Browser, your Word Processor. Consider putting these and other frequently used applications on function keys.

Open Your Most Used Documents

Use Control-Function Keys to open your most used documents. For example, you might have a documentation file or financial details file that you access frequently, consider putting these on Control-Function Keys.

Insert Text Templates

Use Control-Letter and the Insert Text action to type in text for you, such as your name, address, phone number, and so on. Consider restricting these to just the appropriate applications like your Email client or Word Processor by creating a Macro Group for them. Also consider using Typed String triggers for these sorts of macros, for example "=em=" for email address and "=addr=" for address. The text you insert can be typed, pasted as plain text, or can be fully styled text.

Use Hot Keys to Open Financial Accounts

If you keep your finances on your computer, then you probably need to open a document every time you enter a bill or receive a statement. By creating a Hot Key to open the document for you, you can save a few seconds every time – at least it might make receiving a bill slightly less unpleasant! If you have multiple accounts (eg personal, business, association) then this can be even more useful.

Use Hot Keys to Connect to SSH or Web Sites

You could use Hot Keys to connect to your common servers. You might use the Open a URL action, or you could create a Bookmark file for the site and use the Open a File action.

Simulate Bookmarks

Use the Click Browser Link and Browser Form Actions to open web pages, fill in fields, submit forms, follow links. For example, you could use this to log in to the city library for all the members of your family, one in each tab, to easily check what books are due back.

If you are going to use this to enter passwords, use the Set Variable to Keychain Password action to retrieve the password so that it is not stored in plain text in the macros.

Remap Command Keys

If you find yourself pressing a command key in an application and expecting it to do something but it does not work (for example, Command-T for "Replace and Find Again"), use a Macro to make the command key "do the right thing" in that application. Similarly, if you use a function in an application frequently, but it has a convoluted command key or no command key at all, define your own command key by using a Hot Key to select the menu item.

Keep in mind that you can do some menu key remapping in the System Preferences Keyboard preference.

Simulate Missing Features

If you find yourself missing a feature in one application that you are used to in another application (perhaps you switched email clients and a feature is missing), see if you can simulate the feature with a sequence of commands and then use a Hot Key for that. For example, Close Window, Down Arrow, Return to move to next email message, or Command-Left Arrow, Shift-Down Arrow, Command-C, Down Arrow, Command-V to duplicate a line.

Swap Characters

If you often type characters out of order, use a Hot Key to swap them by first placing the cursor between them and then executing:

- Simulate Keystroke Shift-Right Arrow
- Cut to Named Clipboard "Temp"
- Simulate Keystroke Left Arrow
- Paste from Named Clipboard "Temp"

Save a Text Clipping

If you often want to save snippets of text, you could create a Hot Key to save a clipping:

- Copy
- Open File "Clippings.rtf"
- Simulate Keystroke Command-Down Arrow
- Insert Text "== %LongDate% %ShortTime% ==<return>" by Typing
- Paste
- Simulate Keystroke Return
- Simulate Keystroke Return
- Select Menu Item File » Save
- Manipulate Window Close Front Window
- Switch to Last Application (or Quit Specific Application or Command-Q)

Delayed Click

Setup a macro which simply pauses for twenty seconds and then clicks the mouse. Then when you need to print on to an envelope, go all the way through the process, position the mouse over the Print button, execute the Macro, walk over to the printer, insert an envelope and then take the printed envelope back with you.

Insert Boilerplate Text

If you regularly need to insert boilerplate text (eg copyright or file creation text), use an Insert Text macro to insert the text quickly and easily. It can even expand tokens to insert the date or other information.

Apply Text Conversions

If you are regularly translating text from one format to another in an automatic process, perhaps you can automate the whole thing with a macro. For example, converting function declaration in a header file into function definition.

Simulate Workspaces

Create a macro to setup an application to your liking. For example, create multiple tabs in Terminal, each in its own directory, or open multiple documents in TextEdit, each positioned and sized appropriately.

Setup an Application When Launched

If you always do a set of things every time you launch an application (eg arrange the windows in a particular way), use an application Macro Trigger to execute a Macro when you launch the application, then have the Macro do the work for you.

Clean Up After Using an Application

If you always do something after quitting an application (eg unmount a server or disconnect from the Internet), use an application Macro Trigger to execute a Macro when you quit the application. You might need to do a little AppleScripting to perform the action and then use the Execute an AppleScript action.

Launch Scanner Application When Scanner is Connected

Set up a macro that automatically launches your scanner application when your scanner is connected, and quits it again when the scanner is disconnected. This works brilliantly with the ScanSnap scanners – open the lid and the scanner software launches, close it and the scanner software disappears.

Switch Network Location When You Connect

Set up a macro that automatically changes your Network Location when you connect to your home or work wireless network.

Feedback During Macro Execution

A Macro can play a System Beep or use the Speak Text action to speak directly.

You can also use the Alert action to display a window with specified text. This also allows you to stop the macro if you decide not to proceed.

Icon Chooser

Keyboard Maestro includes an Icon Chooser and creator to allow you to select custom icons for your macro groups and macros.

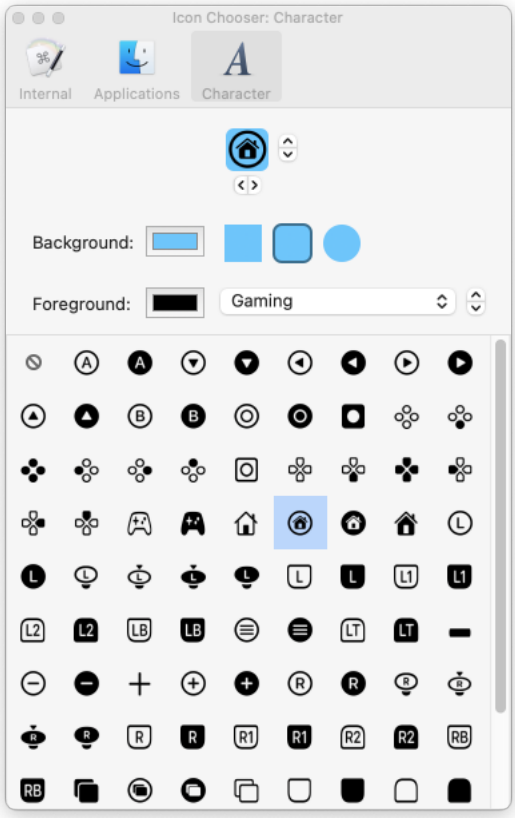
You can display the Icon Chooser by choosing the Window ➤ Icon Chooser menu.

Click on an image well for a macro or macro group (in Edit mode), and then click on an icon in the Icon Chooser to select it.

The Icon Chooser includes three panes with different icons:

- Internal – all the icons Keyboard Maestro normally uses.
- Applications – all the icons Keyboard Maestro can find on your Mac.
- Character – icons you create based on a character.

The Character Icon Choose pane allows you to create your own custom icons by setting a few simple parameters. You can set the background and foreground color, shape, and an optional character:



When you have a Macro or Macro Group icon well selected, and you make any change in the Icon Chooser, or if you click the icon in the Icon Chooser, then the icon for the Macro/Macro Group will be set.

To load the Icon Chooser from an existing Macro or Macro Group, select the icon well of the Macro/Macro Group and then open the Icon Chooser (close it first if necessary).

If an icon well is *not* selected when you open the Icon Chooser, then it will retain the settings that were last used.

Note that icons configured from the Icon Chooser will be very small in storage size in your macros. Icons set any other way will take up more storage in your macros.

To transfer an icon from one macro to another, copy it from the icon well in the source macro and paste it in to the target macro image well.

Application Launcher

The Activate Application Launcher action is essentially a highly specialized macro action that enables you to launch applications. By triggering the macro, the Application Launcher enables you to launch any applications in your Applications or Utilities folder, as well as any recently running applications. Once the launching window appears, you may select the application to launch, and Application Launcher will launch it for you.

The applications are also listed in the Status Menu, so you can launch applications that way if you prefer.

By default, Keyboard Maestro creates a Activate Application Launcher macro in the "Switcher Group" Macro Group, triggered by Command-Control-Tab. You can disable this Macro by selecting the Switcher Group, then selecting the Activate Application Switcher macro and clicking the ✓ button below the Macros list. You can edit this macro to change the trigger to any other desired Hot Key.

Application Switcher

The Activate Application Switcher action is essentially a highly specialized macro action that enables you to launch, switch, hide, and quit applications. By triggering the macro, the Application Switcher enables you to switch between all running applications. Once the switching window appears, you may select the application to activate, and Application Switcher will take you to it.

The Activate Application Switcher action lets you choose from three themes (vertical list, horizontal icons, or icon grid), as well as configure the icon size, color tint, and the sort order.

You can also choose to hide other applications when switching (Keyboard Maestro also has a preference in the General preference pane to always hide other applications when switching).

You can select various applications to always be displayed, even if they are not currently running, perfect for launching frequently used applications. In the Excluded preference pane, you can configure various applications to never be displayed.

While the Application Switcher window is displayed, you can perform various actions:

- Press "q" to mark (or unmark) an application to be quit.
- Press "k" twice to mark an application to be force quit.
- Press "s" or "h" to mark (or unmark) an application to be hidden.
- Press "l" or "z" to mark (or unmark) an application to be launched.
- Press "a" to hide (or show) "always included" applications.
- Press "e" to show (or hide) "always ignored" applications.
- Press "j" to show (or hide) "recently quit" applications.
- Press "c" to select the current application.
- Press "f" to select the Finder.
- Press "d" to switch directly to the current application and hide other applications.
- Press "i" to Get Info on the current application.
- Press "r" to reveal the current application in the Finder.

If you click and hold down on any icon, a menu with these options will appear. Using the menu, you can also configure the switcher to:

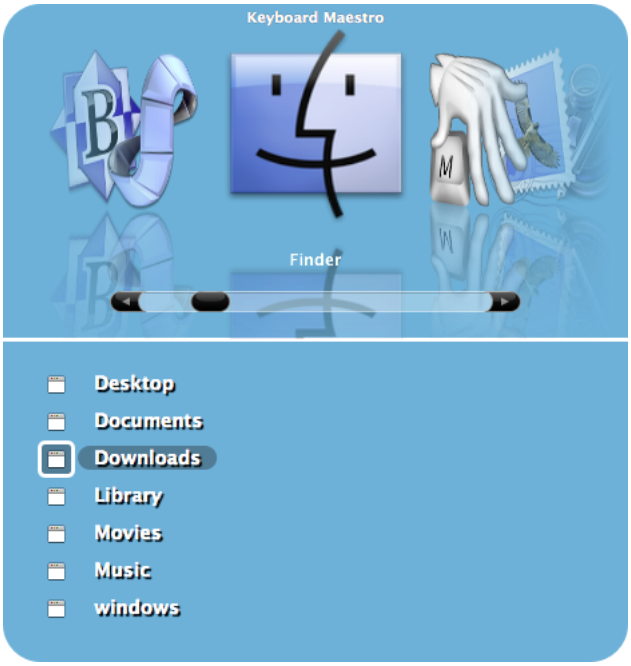
- Switch just the front window instead of all windows.

- Reopen windows when switching.

By default, Keyboard Maestro creates a Activate Application Switcher macro in the "Switcher Group" Macro Group, triggered by Command-Tab. You can disable this Macro by selecting the Switcher Group, then selecting the Activate Application Switcher macro and clicking the ✓ button below the Macros list. You can edit this macro to change the trigger to to any other desired Hot Key avoid replacing the system application switcher.

Window Switcher

Activate Window Switcher is essentially a highly specialized macro action that enables you to show, hide, and minimize windows. By triggering the macro, Window Switcher enables you to switch between all open windows in the current application. Once the switching window appears, you may select the window to activate, and Window Switcher will bring it to the front.



While the Window Switcher window is displayed, you can perform various actions:

- Press "q" to mark (or unmark) an application to be closed.
- Press "s" to mark (or unmark) an application to be minimize.

By default, Keyboard Maestro creates a Activate Window Switcher macro in the "Switcher Group" Macro Group, triggered by Control-Tab. You can disable this Macro by selecting the Switcher Group, then selecting the Window Switcher macro and and clicking the ✓ button below the Macros list.

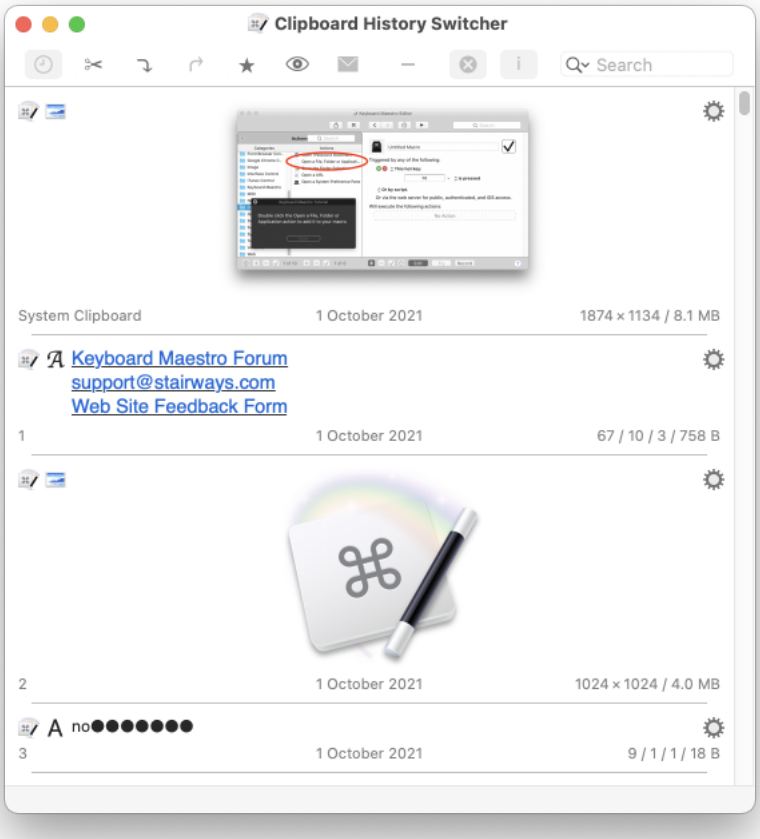
Clipboard History Switcher

The Clipboard History Switcher is in essence a Clipboard Manager. Whenever you copy something from any app, the Keyboard Maestro Engine saves a copy of the System Clipboard at that time to the *Clipboard History* file.

The *Clipboard History Switcher* allows you to:

- Save a copy of the System Clipboard every time you make a copy from anywhere.
- Paste any previous saved clipboard in your History
- View all of the Clipboards in your History
- Perform special operations before you paste, like convert to plain text
- Merge multiple Clipboards into one new Clipboard
- Paste multiple Clipboards with a user-selected delimiter
- Trigger Your Custom Macros to operate on the selected Clipboard.
- Saves up to 200 items (the number is user adjustable in the Preferences).

A typical display of the *Clipboard History Switcher* looks like this:



The Clipboard History Switcher will also shown information such as the size and dimensions of images, and the character, word and line count of text.

How To Use

Display the Clipboard History Switcher

The *Clipboard History Switcher* is displayed using the Activate Clipboard History Switcher Action. You can find a Macro that uses this Action in the *Switcher Group* Macro Group (which comes installed with Keyboard Maestro).

Pasting from the Clipboard History Switcher

- To Paste the Selected Clipboard as is, do any of these:
  - Double-Click on the Clipboard.
  - Press the Return or Enter key.
  - Click on the Paste button.
  - Right-click on the Clipboard and select “Paste”.
  - Drag a Clipboard entry onto App document.

Hold the Shift key while pressing return or double clicking to paste as plain text.

Hold the Option key while pressing return or double clicking to just set the system clipboard.

Command-Numbers will paste the corresponding entry. Command-Shift-Numbers will paste in plain text, and Command-Option-Numbers will just set the system clipboard.

Toolbar

Most of these Toolbar buttons operate on the selected Clipboard.
The buttons from left to right are:

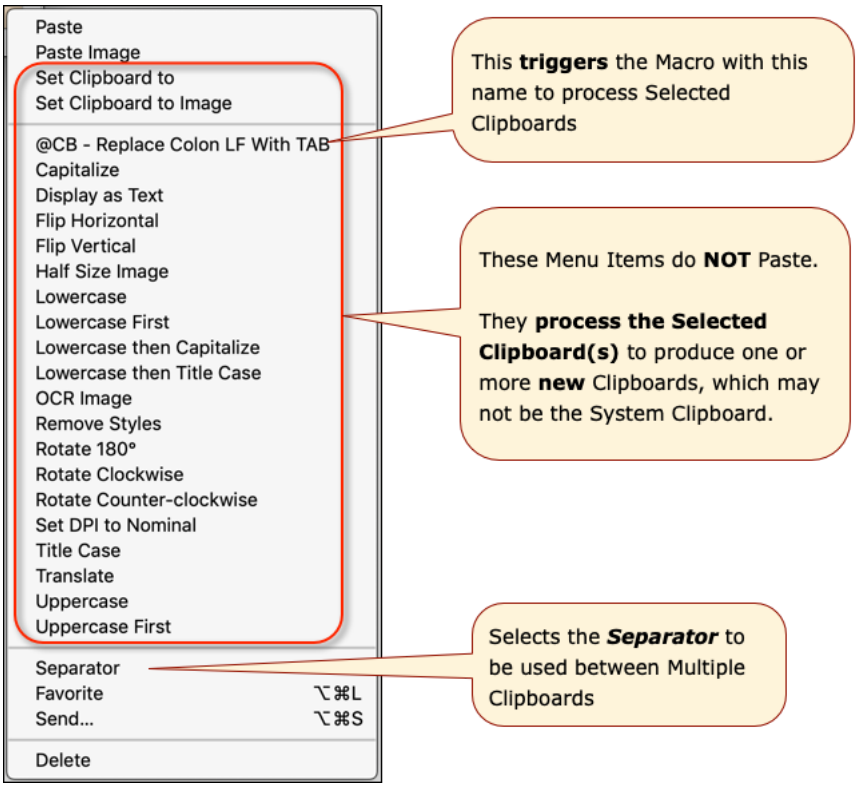
- Toggle between the Clipboard History and the Named Clipboards.
- Cut the current application selection to the Clipboard History.
- Copy the current application selection to the Clipboard History.
- Paste the current Clipboard History selection.
- Mark the current Clipboard History selection as a favorite (so it will never be deleted).
- Quick Look at the current Clipboard History selection.
- Send the current Clipboard History selection to another Mac running Keyboard Maestro.
- Delete the current Clipboard History selection.
- Toggle whether the Clipboard History Switcher will close after pasting.
- Toggle showing additional information.

You can search the Clipboard History with the search field, and limit the display to Favorites-only.

Gear & Context Menu

This menu is available by either clicking on the Gear ⚙ Button on a Clipboard, or by right-clicking anywhere on a Clipboard.

Most of the these menu items are designed to operate on one or more Clipboards, and produce a **new** Clipboard, which can later be pasted if desired.



You can add entries to this menu using the Clipboard Filter trigger.

Operating on Multiple Clipboards

Select two or more Clipboards (using standard macOS multiselection), and then you can:

- Paste them as a single unit, or
- Press ⌘C to merge them to the System Clipboard.
  The Gear ⚙ menu includes a selection for the separator to include between each item when pasting multiple entires. If multiple images are selected, merging them creates a single larger image (horizontally or vertically, depending on the separator).

Other Operations

- Passwords – Clipboard entries that resemble passwords are obscured, deleted after they reach position ten in the clipboard history, and not saved to disk. You can option click on an obscured password to reveal it.
- Send to another Mac
  - You can send clipboards to another Mac running Keyboard Maestro.
    - The receiving Mac will show the received clipboard entries in the clipboard history (assuming the Keyboard Maestro Web Server is enabled on the destination Mac).

Preferences

Excluding Clipboards from Selected Apps

Keyboard Maestro keeps a copy of everything you copy automatically each time you copy something. You can exclude certain applications (for example, badly behaved applications or applications that often have very large clipboards) using the Preferences > Excluded Preferences Panel in the Keyboard Maestro Editor app.

Hidden Preferences

The default Maximum Number of Items stored in the Clipboard History is 200.

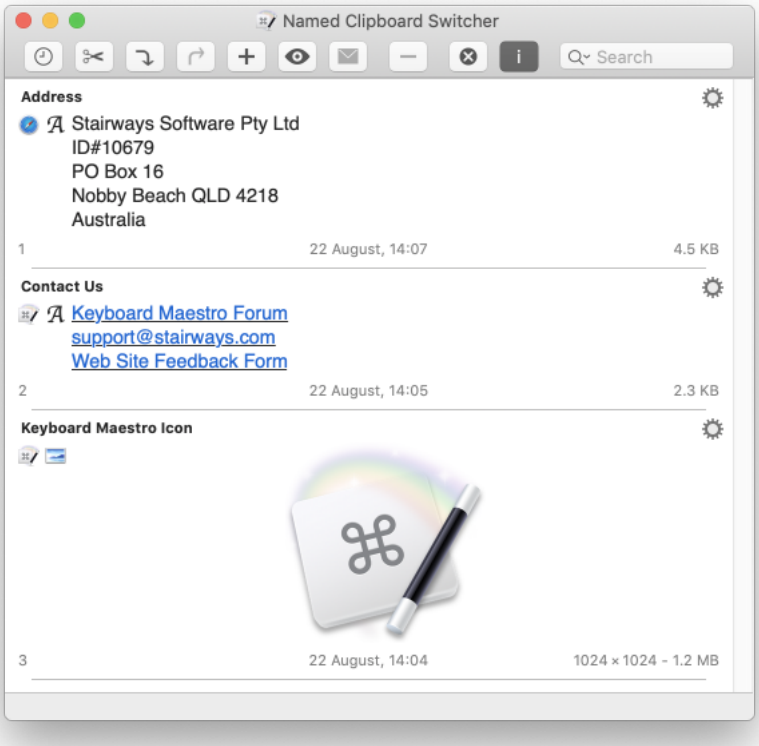
This along with several other features may be changed with the Preferences.

By default, Keyboard Maestro creates a Clipboard History Switcher macro in the "Switcher Group" Macro Group, triggered by Command-Control-Shift-V. You can disable the Macros by selecting them in the Macros window pane and clicking the ✓ button.

Keyboard Maestro also has many macros in the Macro Library, such as macros for Paste Plain Text (Command-Shift-V) and Paste Previous Clipboard (Command-Control-V).

Named Clipboard Switcher

The Named Clipboard Switcher enables you to define any number of named clipboards which can be use to Cut or Copy into and Paste from in any application. To use the Named Clipboard Switcher you simply trigger the Named Clipboard Switcher macro. The Named Clipboard Switcher will present you with a window allowing you to select the named clipboard to use.



You can press arrow keys to scroll through the clipboard entries, or you can use type-ahead to select a named clipboard, and you can use the search field to filter the clipboards.

You can cut or copy the current selection into the selected clipboard, or paste the selected clipboard into the current selection (hold down the shift key to paste as plain text).

You can drag text and images out from the clipboard switcher to other applications.

You can include information about the clipboard entry (index, time copied, size) by toggling the i button. Whether the switcher closes when you paste something is controlled by the x button. The switcher honours the text size setting in the Preferences.

In the ⚙ menu in each clipboard entry, you can Cut, Copy or Paste to/from the item or use it Set the System Clipboard, optionally removing styles or forcing it as an image. You can also Rename or Delete the item.

You can select multiple entries and paste them as a single unit. The ⚙ menu includes a selection for the separator to include between items when pasting multiple entires.

Any active macros that have the Clipboard Filter trigger will be displayed in the ⚙ menu. They can use the Trigger Clipboard facility to operate on the selected clipboard, allowing you to make macros that apply to any selected clipboard (for example, save the clipboard to disk, or uppercase it).

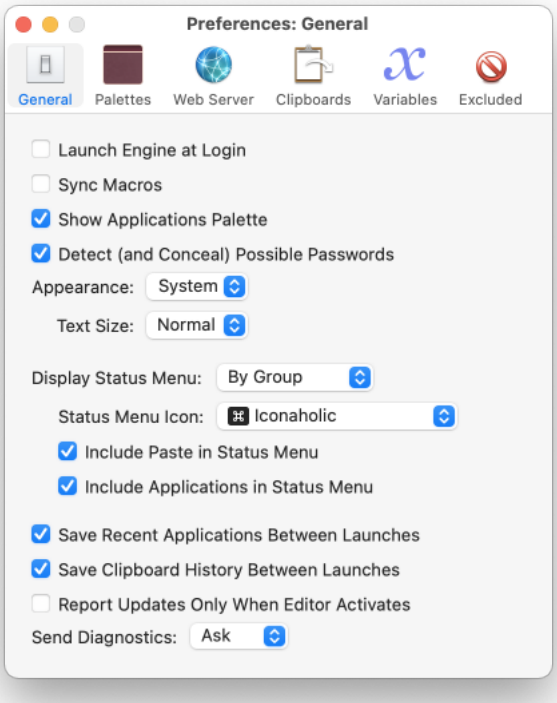
You can toggle to the Clipboard History Switcher by clicking on the button.

By default, Keyboard Maestro creates three Named Clipboard Switcher macros in the "Switcher Group" Macro Group, triggered by Command-Shift-X, C and V (Cut, Copy and Paste). You can disable the Macros by selecting the Switcher Group, then selecting the macros and and clicking the ✓ button below the Macros list.

You can add or delete named clipboards in the Clipboards preference pane.

Preferences

To configure Keyboard Maestro, first launch Keyboard Maestro and choose Preferences from the Keyboard Maestro menu.



The preferences are divided into sections.

General Preferences

In the General preference pane you can:

- enable or disable launching the Keyboard Maestro Engine at Login.
- enable or disable macro syncing.
- Show or hide the Applications Palette.
- configure whether to detect clipboards that resemble passwords and obscure them.
- configure the light/dark appearance (macOS 10.14+).
- configure the default text size used in various places.
- configure whether and how to show the status menu.
- configure the style of the status menu.
- configure whether to include pasting clipboards in the status menu.
- configure whether to include applications in the status menu.
- configure whether to save the recent applications between logins.
- configure whether to save the clipboard history between logins.
- configure whether to only notify about updates when the editor launches.
- configure whether to send diagnostics (crash reports).

Palettes Preferences

In the Palettes preference pane you can adjust the style of the default palette, the Global Macro Palette, the Applications Palette, and the Conflict Palette.

You can configure the Conflict Palette to be placed under the mouse or to have it remain where you place it.

You can configure the Conflict Palette to include entries in the Touch Bar.

Web Server Preferences

In the Web Server preference pane you can:

- enable or disable the web server.
- configure the username, password and port of the web server.
- enable or disable web browser access.
- enable or disable iPhone access.
- enable or disable receiving clipboard entries.
- enable or disable replacing the current clipboard.
- access the web server in your default browser by clicking on Connect.

The web server is disabled by default.

If the web server and web browsing are enabled, then anyone who can connect to your Mac can execute any macro that has a Public Web trigger.

If the web server and web browsing are enabled, and if you have configured a username and password, then anyone who can connect to your Mac and login with the specified username and password can execute any of your macros.

If the web server and iPhone access are enabled, and if you have configured a username and password, then anyone who can connect to your Mac from an iPhone and login with the specified username and password can execute any of your macros.

If the web server and receiving clipboard are enabled, then anyone can send you clipboards which will appear in your clipboard history. By default they do not overwrite the current clipboard, but you can enable that to allow the current clipboard to be directly written remotely.

Macros are still only available when they are enabled and their containing macro group is enabled and active.

You can configure custom styles using the defaults write command to add a custom style, for example:

```
defaults write com.stairways.keyboardmaestro.editor WebServerCustomStyles -string 'body { background:
```

This might be useful if you are controlling multiple Macs and want to differentiate them more clearly.

Clipboards Preferences

In the Clipboards preference pane you can add, remove and rename Named Clipboards and see and change their values.

Named Clipboards store snippets or text or images (or anything the clipboard can hold) and you can copy or paste from them using the Clipboard Switcher or using appropriate macro actions.

You can paste an image into a Named Clipboard by selecting it in the list and pasting an image (if you select part of the text, it will paste the image into the styled text, which works but is probably not what you want).

You can also inspect variables and other values in the Value Inspector.

Variables Preferences

In the Variables preference pane you can add and remove variables and see and change their values.

You do not need to create variables, just using them and giving them a value is all that is required for them to spring into existence. You can also inspect variables and other values in the Value Inspector.

Excluded Preferences

In the Excluded preference pane you can add and remove applications from the global excluded applications list. Excluded applications will not be shown in the Application Switcher, and are (optionally) not hidden or quit by the Hide All Applications or Quit All Applications actions.

You can add and remove applications from the Applications Palette exclusion list. Excluded applications will not appear in the Applications Palette. You can also exclude applications by control or right clicking on them in the Applications Palette and selecting *Exclude from Applications Palette*.

You can also add and remove applications from the clipboard history exclusion list. When an excluded application is at the front, Keyboard Maestro will not automatically record the clipboard history each time it changes. It will still read the current System Clipboard if you explicitly ask for it in a macro or open the Clipboard History Switcher.

Preferences Set by Command Line

You can configure various preferences using defaults write from the Mac OS X Terminal. Some preferences will take effect immediately, but others may require the editor or engine to be relaunched. These preferences are documented here only to provide utility. Their behaviour may change in the future, or have unexpected consequences.

Editor

You can adjust the get mouse location delay with:

```
defaults write com.stairways.keyboardmaestro.editor MouseGetCountdown -int 5
```

You can adjust the spelling of Favorites with:

```
defaults write com.stairways.keyboardmaestro.editor FavoritesDisplayName -string "Favourites"
```

You can turn off automatic completion (autocomplete) with:

```
defaults write com.stairways.keyboardmaestro.editor AutomaticCompletion -bool NO
```

You can limit the maximum undo stack size in the editor with (default 200MB):

```
defaults write com.stairways.keyboardmaestro.editor MaximumUndoSize -int 200000000
```

You can add some custom styling to the detail view displays with:

```
defaults write com.stairways.keyboardmaestro.editor DetailDisplayExtraStyle -string 'div.ActionComment
```

You can set your preferred first day of the week (1=Sunday, 7=Saturday):

```
defaults write com.stairways.keyboardmaestro.editor FirstDayOfWeek -int 2
```

You can disable all animation in the editor with:

```
defaults write com.stairways.keyboardmaestro.editor DisableAnimation -bool YES
```

You can configure the fonts used for the various action text views with:

```
defaults write com.stairways.keyboardmaestro.editor Font-Normal -string "Monaco"
defaults write com.stairways.keyboardmaestro.editor Font-Shell -string "USER"
defaults write com.stairways.keyboardmaestro.editor Font-AppleScript -string "FIXED"
defaults write com.stairways.keyboardmaestro.editor Font-JavaScript -string "SYSTEM"
defaults write com.stairways.keyboardmaestro.editor Font-Swift -string "Times"
defaults write com.stairways.keyboardmaestro.editor Font-HTML -string "Courier:25"
defaults write com.stairways.keyboardmaestro.editor Font-JSON -string "Marker Felt:10"
defaults write com.stairways.keyboardmaestro.editor Font-Comment -string "Palatino:20"
```

The value of this field can be one of "USER", "FIXED" or "SYSTEM" (for the default user, fixed and system fonts), or a font name. This can optionally be followed by a colon and the point size. The default point size is 13. The font size is scaled based on the Large Text setting, and this size is used for the *Normal* setting.

Clipboard

If you wish to effectively disable the Keyboard Maestro Clipboard History, you can do so by setting the history size to zero, and then Keyboard Maestro will only read the clipboard when you explicitly ask it to use the clipboard.

You can set the following preferences for the Clipboard History

Maximum Number of Items (default 200)

```
defaults write com.stairways.keyboardmaestro.engine MaxClipboardHistory -int 200
```

If you set the max number of items to zero, then the Copy action will fail.

Maximum Clipboard Flavor Size (default 100MB)

```
defaults write com.stairways.keyboardmaestro.engine ClipboardFlavorMaximumSize -int 100000000
```

Multiple Clipboard Separator (default Return)

The separator added if you paste multiple items from the Clipboard Switcher can be configured in the gear menu or set to something else with the command:

```
defaults write com.stairways.keyboardmaestro.engine MultipleClipboardSeparator " and "
```

Ignored Clipboard Flavors

You can add additional clipboard flavors to be ignored using:

```
defaults write com.stairways.keyboardmaestro.engine ExtraIgnoredClipboardFlavors "com.whatever.badflav
```

The string should be a bar (|) separated list of flavors (technically a regular expression). These flavors will be excluded from the clipboard when Keyboard Maestro reads the clipboard.

You can also write to `IgnoredClipboardFlavors`, which will set the entire list of ignored clipboard flavors, but that is a bad idea since the flavors that are ignored, are ignored for a reason, and overwriting the list will produce negative results.

Maximum Position of Concealed (Password) Items (default 10)

```
defaults write com.stairways.keyboardmaestro.engine MaxConcealedPosition -int 10
```

Regular Expression Which Matches Text You Think Should or Should Not Be Concealed as Passwords

```
defaults write com.stairways.keyboardmaestro.engine LooksLikePassword -string "^[a-zA-Z0-9]+$"
defaults write com.stairways.keyboardmaestro.engine LooksLikeNotPassword -string "^[a-zA-Z0-9]+$"
```

Include Microsoft PDFs in Clipboards

Microsoft applications include a PDF flavor with every copy, and starting up the PDF rendering engine is slow, making the first copy after launching a Microsoft application. Keyboard Maestro excludes PDF flavors from clipboards copied in Microsoft applications by default, but you can turn this off with:

```
defaults write com.stairways.keyboardmaestro.engine ExcludeMicrosoftPDFFlavor -bool NO
```

Include Microsoft Image Formats

Microsoft applications also include flavors for every possible image format with every copy. Keyboard Maestro excludes secondary image flavors like BMP when there is a PNG or TIFF flavor available. You can turn this off with:

```
defaults write com.stairways.keyboardmaestro.engine ExcludeMultipleImageFlavors -bool NO
```

AppleScript OSA Command

You can set the command line tool that is used to execute AppleScripts

```
defaults write com.stairways.keyboardmaestro.engine OSAScriptCommand -string "/usr/bin/arch -i386 /usr
```

Browser Used in Actions & Tokens

Change which Safari-based Browser will be targeted by the "Safari" Actions, Tokens, and Functions.

These preferences change AppleScript Application name that is used to control the respective browser in the form of `tell application "<browser name>"`. Note that the Safari browser must still behave like Safari, and the Chrome browser must behave like Chrome, or the actions will not work. Do not bother trying to set them to Firefox or something like that as it will not work. All this allows you to do is select a specific version of Safari or Chrome respectively.

You can Safari-based Browser targeted in Actions, etc, as well as the name of the Safari browser shown in applications:

```
defaults write com.stairways.keyboardmaestro.engine AppleScriptSafariBundleID -string "com.apple.Safar
defaults write com.stairways.keyboardmaestro.engine BrowserSafariName -string "Safari Tech"
```

or

```
defaults write com.stairways.keyboardmaestro.engine AppleScriptSafariName -string "com.apple.Safari"
```

Change which Chrome-based Browser will be targeted by the various Keyboard Maestro "Chrome Related" Actions, Tokens, and Functions.

ⓘ Note: By using the FrontBrowser Related Tokens, Actions, and Functions, you may not need to change the below. Just use these *FrontBrowser* entities and whatever Browser is currently (or most recently) frontmost will be targeted. This includes all Safari-based Browsers and Chrome-based Browsers.

```
# ~~~ For Brave Browser ~~~
defaults write com.stairways.keyboardmaestro.engine AppleScriptGoogleChromeBundleID -string "com.brave
defaults write com.stairways.keyboardmaestro.engine BrowserGoogleChromeName -string "Brave Browser"

# ~~~For Chrome Canary ~~~
defaults write com.stairways.keyboardmaestro.engine AppleScriptGoogleChromeBundleID -string "com.googl
defaults write com.stairways.keyboardmaestro.engine BrowserGoogleChromeName -string "Chrome Canary"
```

To restore the defaults back to use Google Chrome, just delete the preferences:

```
# ~~~ Restore to Google Chrome ~~~
defaults delete com.stairways.keyboardmaestro.engine AppleScriptGoogleChromeBundleID
defaults delete com.stairways.keyboardmaestro.engine BrowserGoogleChromeName
```

Add additional Browsers to the Front Browser Actions, Tokens, and Functions.

The current list of browsers supported in Keyboard Maestro as a potential front browser include:

- com.apple.Safari
- com.apple.SafariTechnologyPreview
- com.google.Chrome
- com.google.Chrome.canary
- com.brave.Browser
- com.brave.Browser.beta
- com.vivaldi.Vivaldi
- com.microsoft.edgemac

Plus whatever browser is specified as the Safari or GoogleChrome above.

You can add additional browsers supported as the Front Browser by including their bundle IDs in the `AdditionalWebBrowserBundleIDs` preference. You can list as many as you like, separated by spaces or commas.

```
defaults write com.stairways.keyboardmaestro.engine AdditionalWebBrowserBundleIDs -string "com.example
```

ⓘ Note: if the bundle ID includes "safari", it is considered a Safari browser, otherwise it is considered a Chrome-based browser.

Engine Animation

You can disable all animation in the engine with:

```
defaults write com.stairways.keyboardmaestro.engine DisableAnimation -bool YES
```

Typed String Buffer

You can disable Shift-Space from clearing the Typed String buffer

```
defaults write com.stairways.keyboardmaestro.engine TypedStringClearWithShiftSpace -bool NO
```

and you can disable clicks from clearing the Typed String buffer

```
defaults write com.stairways.keyboardmaestro.engine TypedStringClearWithMouse -bool NO
```

and you can set the idle time for clearing the Typed String buffer (default 5 seconds)

```
defaults write com.stairways.keyboardmaestro.engine TypedStringClearTime -float 5.0
```

Use Unicode for Keystrokes

You can force Keyboard Maestro to simulate keystrokes only as Unicode characters, rather than typing them as you would on the keyboard

```
defaults write com.stairways.keyboardmaestro.engine ForceInsertTextByTypingToUseUnicode -bool YES
```

Hot Key Triggers

You can set the maximum key down trigger time for Hot Key triggers (default 10 seconds)

```
defaults write com.stairways.keyboardmaestro.engine MaxKeyRepeatTime -float 10.0
```

You can configure the timing of a "tap" (Hot Key and USB Device Key triggers) with:

```
defaults write com.stairways.keyboardmaestro.engine MaxTapDownTime -float 1.0
defaults write com.stairways.keyboardmaestro.engine MaxTapUpTime -float 1.0
```

Sounds

You can silence the clipboard transfer sounds with:

```
defaults write com.stairways.keyboardmaestro.engine SilenceClipboardSounds -bool YES
```

And you can silence the recording sounds with:

```
defaults write com.stairways.keyboardmaestro.engine SilenceRecordingSounds -bool YES
```

Recordings

You can adjust the recording delay with:

```
defaults write com.stairways.keyboardmaestro.engine RecordingCountDown -int 5
```

Note: you can option click the Record button to avoid the delay.

You can have all recorded clicks recorded in absolute coordinates by default with:

```
defaults write com.stairways.keyboardmaestro.engine AlwaysRecordAbsoluteClicks -bool YEA
```

By default, clicks are recorded relative to the front window, if that window has remained the front window and retained its size and position for some time. Note that for a short time after recording, you can change the relative field in the action, and the values will update to appropriate values.

Conflict Palette

You can have the conflict palette use sequential hot keys from the first letter with:

```
defaults write com.stairways.keyboardmaestro.engine SequentialConflictPalette -bool YES
```

The default allowable down time is the double click interval, and the default allowable up time is twice the double click interval.

You can set configure whether the Status Menu includes the Paste submenu and how many items it includes with:

```
defaults write com.stairways.keyboardmaestro.engine StatusMenuIncludePaste -bool NO
defaults write com.stairways.keyboardmaestro.engine StatusMenuPasteCount -int 30
```

You can set configure Display Large text color and display period with:

```
defaults write com.stairways.keyboardmaestro.engine DisplayLargeTextColor -string "502033"
defaults write com.stairways.keyboardmaestro.engine DisplayLargeTextDisplayPeriod -float 5.0
```

By default recorded clicks do not restore the mouse location, but you can change that with:

```
defaults write com.stairways.keyboardmaestro.engine RecordedClickRestoresMouseLocation -bool NO
```

Notifications

You can control which notifications are displayed with:

```
defaults write com.stairways.keyboardmaestro.engine "Notification-Information" -bool NO
defaults write com.stairways.keyboardmaestro.engine "Notification-MacroExecution" -bool NO
defaults write com.stairways.keyboardmaestro.engine "Notification-MacroCancelled" -bool YES
defaults write com.stairways.keyboardmaestro.engine "Notification-ActionFailed" -bool YES
defaults write com.stairways.keyboardmaestro.engine "Notification-ReceivedClipboard" -bool YES
```

Software Updates

You can have new versions reported only when the editor activates with:

```
defaults write com.stairways.keyboardmaestro.engine ReportUpdatesWhenEditorActivates -bool YES
```

Prompt With List Show All Limit

You can set the limit for the maximum number of items shown by default in the Prompt With List action with:

```
defaults write com.stairways.keyboardmaestro.engine PromptWithListShowAllLimit -int 100
```

When there is more items than this in the list, by default the action will not show the items as the list may be too long to be practical.

WebKit Inspector for HTML Prompt

You can enable the WebKit inspector with:

```
defaults write com.stairways.keyboardmaestro.engine WebKitDeveloperExtras -bool YES
```

Then you can control-click on elements and use Inspect Element and get an inspector window, including a Console tab.

Custom Palette Theme

You can configure a custom palette theme color style with:

```
defaults write com.stairways.keyboardmaestro.engine CustomPaletteTheme -string "0,90,74, 110,90,74, 25
```

The style includes 3 or 6 HSB colors (being background, text, highlighted letter, selected background, selected text, selected highlight letter) each specified as hue,saturation,brightness. The background brightness is dropped by 33%, but if there are

only three colors, then the selected background color is the background without the 33% drop. You can select the Custom style in the Theme Editor.

Alternatively, you can specify the theme using JSON and including all the details for the theme including optional gradients. The format is:

```
{
    "IncludeTexture":<BOOLEAN>,
    "TitleBackground":<COLORorGRADIENT>,
    "TitleBorder":<COLORorGRADIENT>,
    "BodyBackground":<COLORorGRADIENT>,
    "BodyBackgroundSelected":<COLORorGRADIENT>,
    "Frame":<COLOR>,
    "TitleText":<COLOR>,
    "Text":<COLOR>,
    "TextHighlighted":<COLOR>,
    "TextSelected":<COLOR>,
    "TextSelectedHighlighted":<COLOR>,
}
```

Type	Value
<BOOLEAN>	“true” or “false”
<COLOR>	an array of three numbers, being hue (0-360), saturation (0-100), and brightness (0-100)
<COLORorGRADIENT>	either a <COLOR> or a dictionary {“c1”:<COLOR>, “c2”:<COLOR>, “angle”:<0-360>}

IncludeTexture specifies whether Keyboard Maestro includes the subtle mottling texture on the main background.

For example:

```
defaults write com.stairways.keyboardmaestro.engine CustomPaletteTheme -string '{
"IncludeTexture":false,
"TitleBackground":{"c1" : [180,30,53], "c2" : [180,30,40], "angle":270},
"TitleBorder":[180,30,40],
"BodyBackground":{"c1" : [180,30,50], "c2" : [180,30,38], "angle":270},
"BodyBackgroundSelected":[180,30,60],
"Frame":[180,30,40],
"TitleText":[180,40,92],
"Text":[180,40,92],
"TextHighlighted":[180,40,70],
"TextSelected":[180,40,98],
"TextSelectedHighlighted":[180,40,75],
}'
```

Note that the “-string” is important - otherwise the `defaults` command will interpret the JSON itself and that will not work.

Scripting

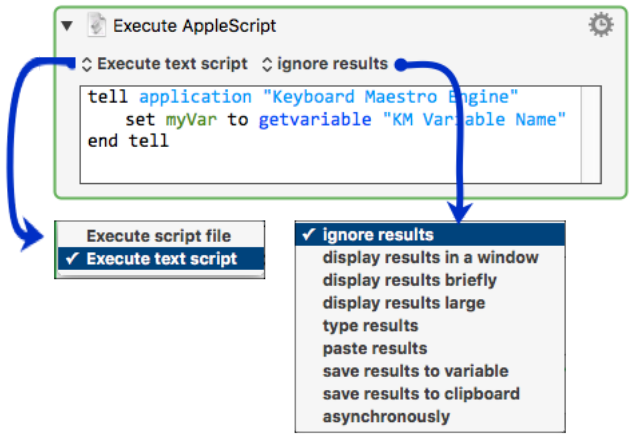
Executing Scripts

The following *Execute Script* Actions are provided to execute a variety of script languages:

- Execute an AppleScript
- Execute an Automator Workflow
- Execute a JavaScript For Automation
- Execute JavaScript in Custom HTML Prompt
- Execute a JavaScript in Browser
- Execute a Shell Script
- Execute a Swift Script
- Apply a BBEdit Text Factory to the Clipboard

You can specify the script to be executed either by reference to a file or as text entered directly in the *Execute Action*. After pasting or typing script text into the Execute Script Action, press the Enter key to compile and format the script.

For example, the Execute_an_AppleScript Action:



There are two setup options:

Setup Option	**Choices** **(Default shown first)**
Script location	Execute text script (type or paste script into Action text field) Execute script file (This may be faster if it is a compiled script file `.scpt`)
Script Results	Ignored. Displayed in a floating window. Displayed briefly in a Notification. Typed in the current text field that has focus. Pasted in text field that has focus. Saved to a Keyboard Maestro Variable. Saved to the System or Named Clipboard. Ignore Results and run *Asynchronously* (the script runs while the macro continues on to the next *Action*).

Getting Script Results

The results of scripts can be displayed, or they can be typed or pasted in to the current application document, or saved into a Keyboard Maestro Variable or the System Clipboard. This allows you to insert text that depends on many factors, such as date calculations, file listings, SQL queries, web pages, or anything else you can imagine. Alternatively, the results can be displayed in Notification Center, in a window, or in large type across the screen. Or you can ignore the results, or execute the script asynchronously, leaving it running in the background.

Using the Clipboard

You can also read and set the clipboard in a script, although the easiest way to set the clipboard to the results of a script is to select the Execute Script Action option to "save results to clipboard". This works *only* for plain text returned by the script. For more complex formats, you will need to set the Clipboard directly in the script.

For more information about using the Clipboard in scripts, see:

- AppleScript Clipboard Commands suite [https://developer.apple.com/library/mac/documentation/AppleScript/Conceptual/AppleScriptLangGuide/reference/ASLR_cmds.htr
- The unix man pages for pbcopy, pbpaste.

Automating Applications and Adding Functionality to Keyboard Maestro

AppleScripts, JavaScript for Automation scripts, Shell scripts, and Swift scripts, give you a powerful way of adding new facilities we have not specifically provided for, as well as controlling other applications.

Web Page Interactions

The Execute a JavaScript in Browser actions enables deep control over a web page, as well as extracting specific data from both the page contents and HTML.

Shell Scripts

Shell scripts can execute any installed scripting language, such as perl, python, ruby or whatever. Be aware that because shell scripts are executed in a non-interactive shell, typically none of the shell configuration files (like .login or .profile) will be executed, which may change the way the shell script behaves.

Variables can be accessed from shell scripts via the environment variables in the form $KMVAR_Variable_Name where KMVAR_ is prefixed, and spaces are converted to underscores. AppleScripts can also access the environment variables using the system attribute command, but note that system attribute is not safe for international characters.

Note that the total size of the variables stored in the environment is limited to 100K, so larger variables may be excluded to ensure the variables do not take up excessive amounts of environment space since this is limited by the system.

Variables whose names start with "ENV_" override regular environment variables (eg "ENV_PATH" overrides the regular "PATH" environment variable).

AppleScript

AppleScripts can perform many tasks on the Mac and control many applications.

You can read and write variables, dictionaries and control the Keyboard Maestro engine with AppleScript:

```
tell application "Keyboard Maestro Engine"
  set calcResult to getvariable "Calculation Result"
  -- If the Keyboard Maestro Variable does not exist, the AppleScript Variable will be set to empty st

  setvariable "Calculation Result" to (calcResult + 10)
  -- If the Keyboard Maestro Variable does not exist, it will be created
end tell
```

For AppleScripts compatible with prior versions of Keyboard Maestro, see Using AppleScript prior to version 7.1.

JXA

JavaScript for Automation (JXA) can access Variables with:

```
var kme = Application("Keyboard Maestro Engine");
var oldValue = kme.getvariable('Calculation Result');
kme.setvariable('Calculation Result', { to: 10 });
```

For details on JXA, see JavaScript for Automation. This provides an introduction to JXA and comparison with AppleScript.

JavaScript

JavaScript in web browsers can access the variable values by using the document.kmvar dictionary, like document.kmvar.Variable_Name (spaces are converted to underscores), but they have no way to write values back to variables, except by returning a result from the script.

User Interaction

AppleScripts and JavaScript For Automation scripts are executed in the background via osascript. This means they are not allowed to do user interaction. You can work around this by asking an application like System Events to do the user interaction for you, for example:

```
tell application "System Events"
  activate
  display dialog "Hello"
end tell
```

The osascript tool will execute in 64-bit mode, which may be a problem if you have old versions of AppleScript extensions installed. However, you can set the command line tool that is used to execute AppleScripts as described in Other Hidden Preferences.

See also the Variables section.

Controlling Keyboard Maestro Engine via Scripting

The primary scripting interface to Keyboard Maestro Engine is the Keyboard Maestro Engine's do script support. You can ask Keyboard Maestro Engine to:

- execute a macro by name
- execute a macro by unique ID
- execute an action given its XML code

Note in most cases you must tell "Keyboard Maestro Engine", not "Keyboard Maestro".

The easiest way is to use the name, for example:

```
tell application "Keyboard Maestro Engine"
  do script "[Name of Your Macro]"
end tell
```

The macro must be defined and enabled, and the macro group must be enabled and currently active.

If there is more than one macro with the same name, you will get an error, so you can use a UID instead of a name.

```
tell application "Keyboard Maestro Engine"
  do script "D0C150C7-8A0C-4837-918A-427E2BCFB6B9"
end tell
```

The do script will not return until the macro is finished executing.

You can pass an optional parameter using the with parameter clause, which you can read in the macro as the %TriggerValue% token.

You can determine a macro's UID by selecting it and choosing Copy UID command in the Copy as sub-menu in the Edit menu.

An even more powerful way to script Keyboard Maestro is to execute specific actions based on their XML code. This allows you to construct any action, including changing the action on the fly, without having to create a macro first. A simple example would be:

```
tell application "Keyboard Maestro Engine"
  do script "<dict><key>MacroActionType</key><string>SwitchToLastApplication</string></dict>"
end tell
```

The easiest way to determine the appropriate XML is to create an example action in an example macro and then choose Copy as XML in the Copy as sub-menu in the Edit menu.

You can read information about the existing macros using the gethotkeys and getmacros commands. See the Keyboard Maestro Engine AppleScript dictionary for more information.

Using Keyboard Maestro Facilities from AppleScript

Keyboard Maestro Engine makes several of its facilities available to AppleScript.

You can ask it to play a sound with:

```
tell application "Keyboard Maestro Engine"
  play sound alias "Harddisk:System:Library:Sounds:Glass.aiff"
end tell
```

You can ask Keyboard Maestro Engine to perform a calculation for you with:

```
tell application "Keyboard Maestro Engine"
  set n to calculate "JULIANDATE()"
end tell
```

or process tokens with:

```
tell application "Keyboard Maestro Engine"
  process tokens "%LongDate%"
end tell
```

The `calculate` and `process tokens` commands can also take an `instance` parameter (v10.0+) to specify the instance for local variable access.

You can ask Keyboard Maestro Engine to find strings in other scripts:

```
tell application "Keyboard Maestro Engine"
   count found in "The Source" for ".e" with regex
end tell
```

And you can search and replace with:

```
tell application "Keyboard Maestro Engine"
        search "3+4" for "(\\d+)" replace "%CalculateFormat%CALCULATE(\\1)%Currency%" with regex and p
end tell
```

Controlling Keyboard Maestro via Scripting

Macro Groups, Smart Groups, Macros, Triggers, Actions are all available via AppleScript. So you have deep control over controlling the Keyboard Maestro editor itself, allowing you to automate creating macros in a variety of different ways.

You can disable or enable a Macro Group (or similarly Macro) from AppleScript with:

```
tell application "Keyboard Maestro"
  set enabled of macro group "Macro Group Group Name or UID" to true/false
end tell
```

Alternatively you can use the Set Macro Enable action.

You can rename, create, delete, duplicate macro groups and macros. You can add and remove actions. You can get or set the selection.

You can start editing a Macro or Macro Group from AppleScript with:

```
tell application "Keyboard Maestro"
  editMacro "Macro/Macro Group Name or UID"
end tell
```

See the Keyboard Maestro AppleScript dictionary for more information.

Controlling Keyboard Maestro Engine via Scripting

You can adjust Keyboard Maestro Engine windows with:

```
tell application "Keyboard Maestro Engine"
    set bounds of window "Clipboard History Switcher" to {1000, 50, 2000, 1200}
end tell
```

URL Schemes

Editor

Another way you can control Keyboard Maestro editor is using the "`keyboardmaestro`" URL scheme, which supports the following formats:

Example Editor URL Command	Description
`keyboardmaestro://u=support%40stairways.com/s=ABCDEFGH0123456789`	Enter your username/serial number.
`keyboardmaestro://m=Activate%20Application%20Switcher` `keyboardmaestro://m=D2F427A1-51E3-4719-820B-4C25B6FF7329`	Edit a specific macro or macro group. You may used either the Macro Name, or UUID.
`keyboardmaestro://q=Activate`	Filter macros with this keyword.
`keyboardmaestro://g=All%20Macros/q=Activate`	Select a macro group and filter macros with this keyword.
`keyboardmaestro://a=Execute`	Filter actions with this keyword.
`keyboardmaestro://c=All%20Actions/a=Execute`	Select action category and filter actions with this keyword.

Triggers

You can trigger a macro (that is Active and Enabled) using the "`kmtrigger`" URL scheme. The URL uses this format:

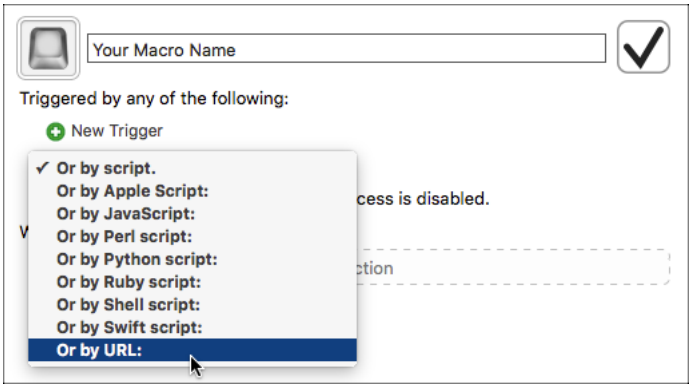
```
kmtrigger://macro=<Macro Name or UUID>[&<Trigger Value>]
```

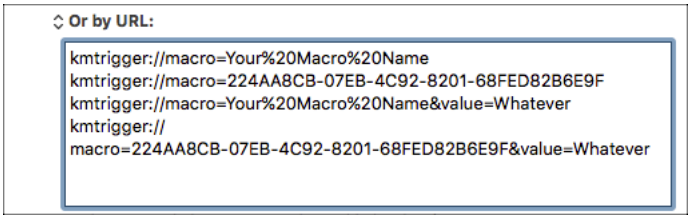
where

- `<Macro Name or UUID>` is either the Macro Name, OR, its UUID
- `<Trigger Value>` is the *optional* Trigger Value
- `[ ]` are *not* included. Used only to show *optional* parameters
- `&` must precede the Trigger Value, if it is provided.

Note both Macro Name and Trigger Value must be URL encoded.

You can easily get the Trigger URL by clicking on the "Or by Script" Trigger option, and selecting "Or by URL".





Example Trigger URL Command	Description
`kmtrigger://macro=Your%20Macro%20Name`	Using the Macro Name
`kmtrigger://macro=224AA8CB-07EB-4C92-8201-68FED82B6E9F`	Using the Macro UUID
`kmtrigger://macro=Your%20Macro%20Name&value=Your%20Trigger%20Value`	Using Macro Name with Trigger Value
`kmtrigger://macro=224AA8CB-07EB-4C92-8201-68FED82B6E9F&value=Your%20Trigger%20Value`	Using Macro UUID with Trigger Value

The `kmtrigger` URL will only work on the local Mac, but you can use the Remote trigger to trigger macros with a URL even from remote locations via the Keyboard Maestro trigger server.

Status Menu Icons

Keyboard Maestro allows you to select the status menu icon.

The current lovely status menu icons were done by Iconaholic [https://www.stairways.com/action/linkthru?iconaholic], and you can also use the Classic finger tapping icons. As well as that, you can get more status menu icons from our web site [https://www.stairways.com/action/linkthru?getmorestatusmenuicons] or the forum [https://forum.keyboardmaestro.com/c/statusmenuicon] and you can create and optionally contribute your own versions. You can drop a new status menu icon .zip archive on the Keyboard Maestro application dock icon to install it.

The format is fairly simple, you need a folder with the name of the icon (which must be unique relative to other contributed icons), and in that folder you have a sequence of files named StatusItem and then then the animated sequence StatusItem1, StatusItem2, etc. You can have as many as you like. If the icon is a template icon (black and clear only), then add the word "Template" to the name. And finally the extension can be either .tiff or .png. The file should be either a 16×16 icon, or ideally a retina tiff file with both a 16×16 and 32×32 at 144dpi (you can create the retina-ready tiff files using tiffutil -cathidpicheck, or you can use a tool like Opacity.app).

The two default icon folder structures look like this:

- Classic
  - StatusItem.tiff
  - StatusItem1.tiff
  - StatusItem2.tiff
  - StatusItem3.tiff
  - StatusItem4.tiff
  - StatusItem5.tiff
  - StatusItem6.tiff
- Iconaholic
  - StatusItemTemplate.tiff
  - StatusItem1Template.tiff
  - StatusItem2Template.tiff
  - StatusItem3Template.tiff
  - StatusItem4Template.tiff

Note that the "Template" appears at the end, after the animation index, and there are no spaces anywhere. You can have spaces in the folder name.

Then you archive the folder into a zip file named NameStatusMenuIcon.zip (eg ClassicStatusMenuIcon.zip) and drop it on the Keyboard Maestro application dock icon or email it to us [mailto:mailto:support@stairways.com] with a cover letter indicating your permission for it to be distributed, as well as your name to be published with an optional URL if desired.

Plug In Actions

You can develop custom Macro Actions using a facility known as *Plug In Actions*. After you install a *Plug In Action* in your local Keyboard Maestro Support folder, you can use them like the built-in Actions.

Third Party Plug In Actions

You can download Plug In Actions developed by others from these sources:

1. Keyboard Maestro Web Site [https://www.stairways.com/action/linkthru?thirdpartyactions]
2. Keyboard Maestro Forum [https://forum.keyboardmaestro.com/c/plugin]

How to Install

Plug In Action Install Files are `.zip` Archive Files

1. **Initial Install:** Drop on the Keyboard Maestro app Dock icon
   - This will create a Plug In Action sub-folder by the same name as the `.zip` file
   - Do *not* unzip the file
2. **To Update** An Existing Plug In Action
   a. First Manually Delete or Move the Plug In Action Sub-Folder from:
   `~/Library/Application Support/Keyboard Maestro/Keyboard Maestro Actions/` folder
   b. Then Drop the `.zip` file onto the Dock Icon

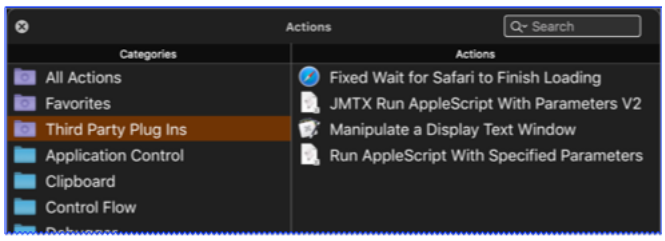
How To Use a Plug In Action

Plug In Actions are used just like the built-in Keyboard Maestro Actions.

After you have installed a Plug In Action, you may need to restart both the Keyboard Maestro Editor and Engine in order for the new Plug In Action to be recognized by Keyboard Maestro.

1. In the Keyboard Maestro Editor, goto *File* > *Quit Engine*.
2. Quit and re-launch the Keyboard Maestro Editor. This will also start the *Engine*.

To insert a Plug In Action into your Macro, use any of the normal methods to insert Actions:

1. Display the *Actions Panel* from the menu or with the shortcut ⌘K, and select the *Third Party Plugins* folder:
   - 

2. Menu *Edit* > *Actions* > *Third Party Plugins*
3. *Insert Action by Name*, using shortcut ⌘^A.

Enter the Plug In Action form fields (parameters) that are shown on the action form.

Building a Plug In Action

How to Create a Plug In Action

There is no easy way to create a *Plug In Action*. There is no one editor (except maybe BBEdit) that can be used to create all of the files that are needed. Probably the best way to get started is to examine an existing Plug In Action, like this one:

- Fixed Wait for Safari to Finish Loading Plug In Action [https://forum.keyboardmaestro.com/t/fixed-wait-for-safari-to-finish-loading-plug-in-action/13343] by @PeterNLewis.

Steps to Create a Plug In Action

1. **Folder:** Create a folder on your local drive, using the same name that you will use for the Plug In Action, put the following files in this folder.
2. **Plist:** Create a file named `Keyboard Maestro Action.plist`, which is a standard Cocoa Property list [https://developer.apple.com/library/archive/documentation/General/Conceptual/DevPedia-CocoaCore/PropertyList.html] file that defines the Plug In Action form.
   - See Plug In Action Plist Definition below.
   - The Plug In Action Plist is an XML which defines the Action form you see in the Keyboard Maestro Editor, and identifies the name of the script file that will be run when the Plug In Action is executed in a triggered macro.
3. **Script:** Create a script file named `Action.scpt`, or other suitable name (like the name of your Plug In Action), which can be an AppleScript, JXA, or Shell Script, to process the data from the Plug In Action form.
4. **Icon:** Create `icon.png` which is the icon that will be displayed in the Keyboard Maestro Editor. This file is optional.
5. **Zip Install File:** When you are finished with all of these files, create a `.zip` archive from the Plug In Action Folder.

Plug In Action Folder

A *Plug In Action* is Contained in a Folder, Whose Name:

1. Should generally closely match the action name
2. Must be made up of only ASCII alphanumerics, underscores and spaces.
3. Must be unique among all plug in actions.
4. Is stored in the Keyboard Maestro Macros.plist to reference the plugin action.

The Plug In Action folder contains a set of files, including:

1. Keyboard Maestro Action.plist – an XML file describing the action.
2. A script file
   - whose name must be made up of only ASCII alphanumerics or underscores, plus an ASCII alphanumeric extension.
   - It may be a shell script or an AppleScript.
   - If it is a shell script, it will be made executable automatically.
3. Icon File: 64×64 png icon (optional).

The Plug In Action Folder must be stored as a `.zip` archive file for installation.

Plug In Action PList

The format of the Keyboard Maestro Action.plist is a Cocoa Property list [https://developer.apple.com/library/archive/documentation/General/Conceptual/DevPedia-CocoaCore/PropertyList.html] containing a dictionary with the following keys and values:

Key	Description
Name	**Name of the action**. (which appears in the Category/Actions list)
Script	**Name of the script file** which will be run when the Plug In Action is executed. The name can only use ASCII alphanumerics or underscores, plus an ASCII alphanumeric extension
Parameters	**An array of parameters to the script** • Each entry is a dictionary as described below. • These parameters will appear as form fields in the Plug In Action in the Keyboard Maestro Editor.

Key	Description
	• They allow the user of the Plug In Action to enter the data to be processed by the script. • **Optional**
Icon	**Name of the icon png file**, made up of only ASCII alphanumerics or underscores plus .png **Optional**
Title	**The title displayed on the action** • Which can include %Param%XYZ% tokens. • It should usually not include other tokens. • If it is missing, the Name will be used. • **Optional**
Timeout	**The default timeout in seconds** • Entered as a *number* • Set it to 0 if the action needs no timeout • The default is 99 hours • Like all Actions, Timeout can be changed in the *Editor* • **Optional**
Author	The author of this action **Optional**
URL	URL for the author or this action **Optional**
Help	A short (Tool Tip) explanation of this action **Optional**
Results	**What to do with the output of the script if any**. Possible Values: • None • Window • Briefly • Typing • Pasting • Variable • Clipboard Multiple values can be used, separated by a bar (|), the first specified value is the default **Optional**

Plist Parameters

Parameter Key	Description
Label	**Name of the parameter**. • The same rules as Keyboard Maestro Variable Names apply. • The label is displayed to the user and used to pass the parameter to the script. • Obviously, the label must be unique amongst all parameters
Type	**Type of the parameter**. Must be one of the following: *(Text in italics is info, and not part of the Type)*

Parameter Key	Description
	• String *(single line)* • TokenString • Calculation • Text *(multi-line)* • TokenText • Checkbox *(0 or 1)* • PopupMenu • Hidden The Type specifies how the value is displayed to the user and what processing is applied before it is passed to the script. Hidden types are text token processed, but are not displayed in the editor
Menu	Values of the popup menu, separated by | **Optional** unless Type is *PopupMenu*.
Default	Default value of the Parameter when the Plug In Action is inserted in the Macro. **Optional**

Each parameter in the Parameters array is a dictionary with the following keys:

Warning: Keys are case sensitive.

Retrieving Parameters in a Script

Parameters are passed to the script via *environment variables* with names starting with KMPARAM_, similar to how variables are passed to shell scripts with the Execute Script action.
So a parameter named "My Text" would be in an environment variable KMPARAM_My_Text.
Note that `spaces` in the variable names *must* be replaced with `underscores` in your script.

Methods To Retrieve Parameters Using AppleScript

```
--------------------------------------------------------------
--»     1.  Use AppleScript "system attribute"
--------------------------------------------------------------
-- This method is "not safe for international characters" (and emoji)

set myText to system attribute "KMPARAM_My_Text"

--------------------------------------------------------------
--»     2.  Use Shell Script with "echo"
--------------------------------------------------------------
-- This method works with emoji and international characters
-- but multi-line text (as from a TokenText form field)
-- will be flattened into a single line

set myText to do shell script "echo $KMPARAM_My_Text"


--------------------------------------------------------------
--»     3.  Use Shell Script with "printenv"
--------------------------------------------------------------
-- This works with emoji, international characters and multi-line text
-- however printenv returns an error if KMPARAM_My_Text doesn't exist.
-- You can catch this error with a try command.

try
  set myText to do shell script "printenv KMPARAM_My_Text"
on error -- Parameter does NOT exist
```

```
  set myText to ""
end try
```

Reload to Use Updated Plug In Actions

In normal use, once a plug in action is read, it will stay in memory and changes will not be noticed (although the script will be executed each time, so changes to that will be noticed). To cause the editor and/or engine to notice changes to the plug while in development, use AppleScript to reload the macros:

```
tell application "Keyboard Maestro" to reload
tell application "Keyboard Maestro Engine" to reload
```

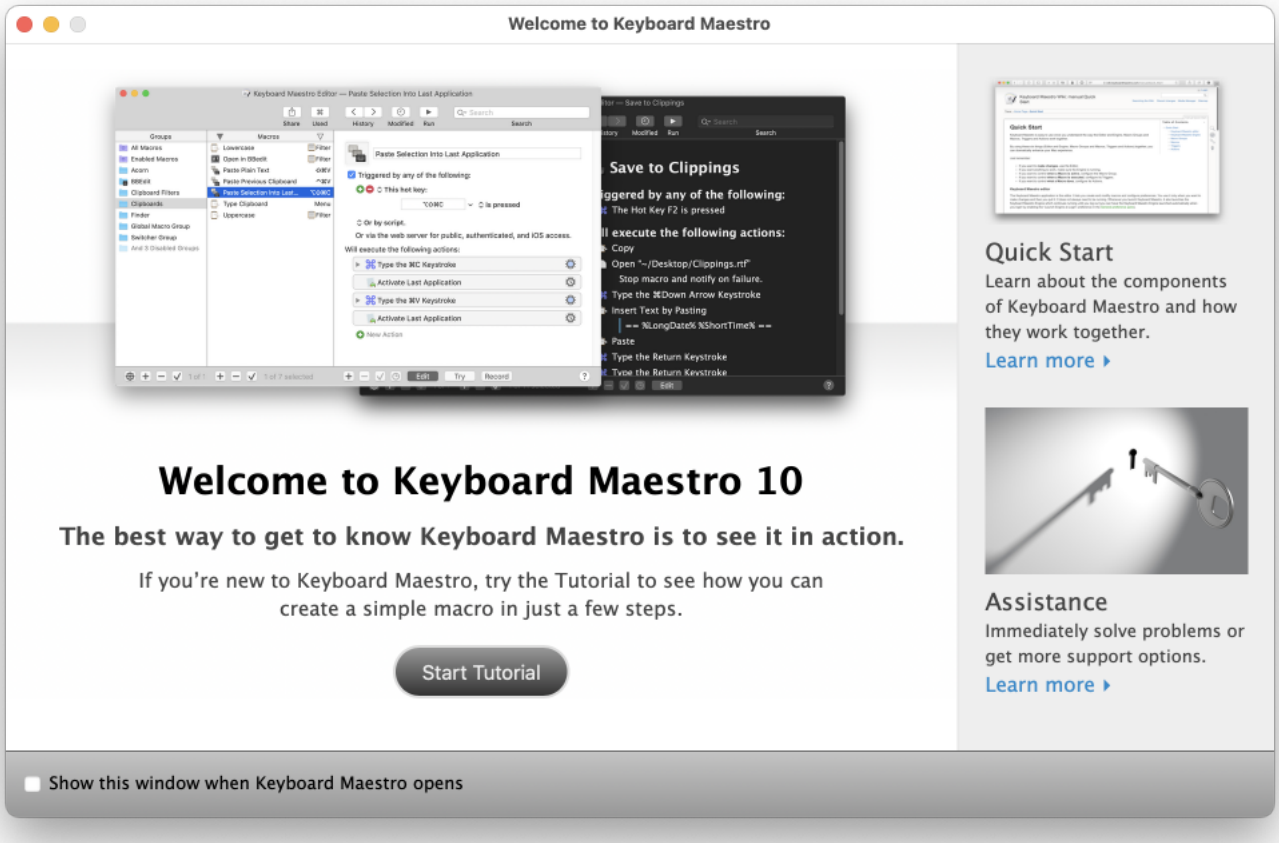
Plug In Action Failure

If the Plug In script fails, the action will fail, potentially aborting the macro.

Windows

Welcome Window

This window welcomes new users and gives you some options for learning about Keyboard Maestro.

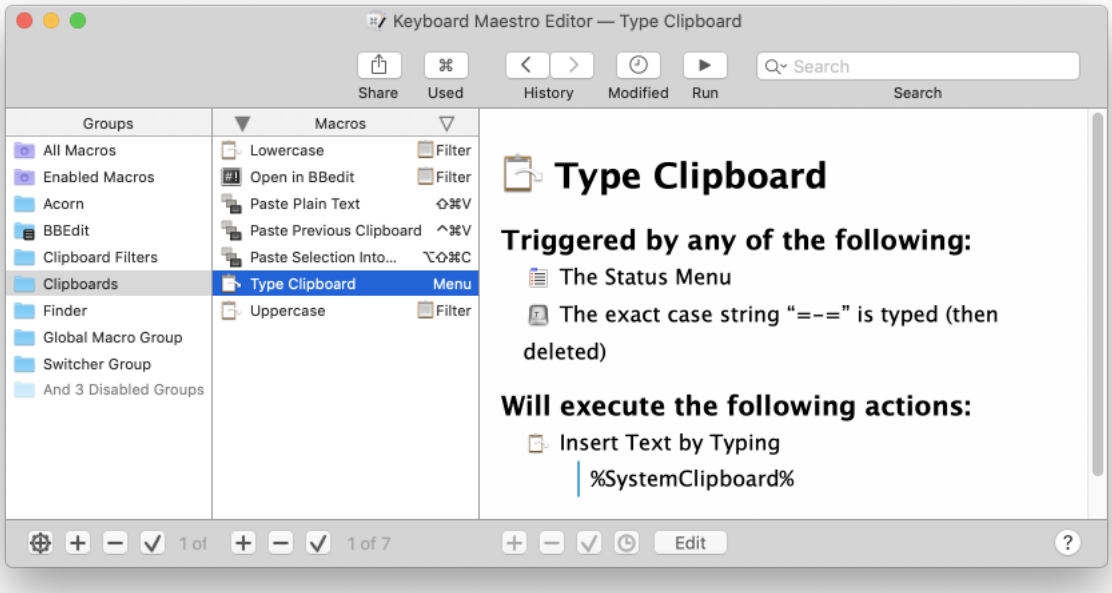


If you are new to Keyboard Maestro, start the tutorial and Keyboard Maestro will walk you through creating a simple macro.

Macros Window

This window lets you manipulate Macros and Macro Groups, creating new ones, deleting old ones, enabling and disabling them, editing them and so on.

You get this window pane by launching Keyboard Maestro.



The window contains a list of Macro Groups and their associated Macros.

You can create a new Macro Group by clicking the + button below the Groups list.

You can create a new Macro by selecting a macro group and then clicking the + button below the Macros list.

You can see the selected Macro Group or Macro in the right hand column, and edit it by clicking the Edit button.

You can select the All Macros meta-Group to show all Macros, and you can use the search field to filter down the list of macros.

You can rename a Macro Group or Macro by double clicking it and changing the title.

You can delete a Macro Group by selecting it and clicking the - button below the Groups list.

You can delete a Macro by selecting it and clicking the - button below the Macros list.

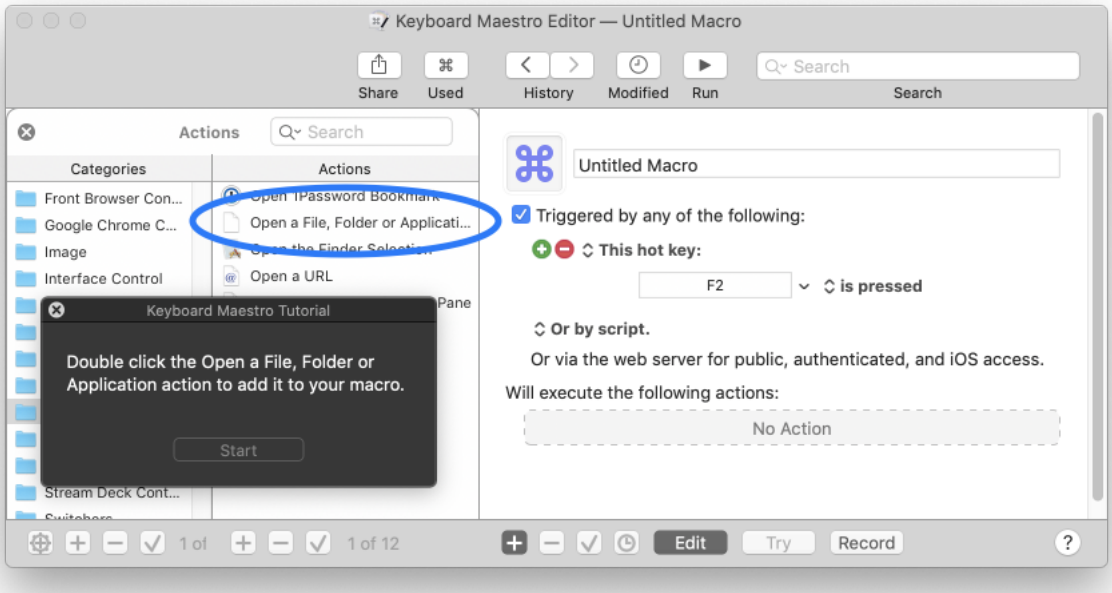
Similarly, you can enable or disable Macro Groups or Macros by clicking their respective ✓ button.

You cannot delete, rename or modify the Global Macro Group.

See also the Macro Groups, Macros and Macro Editor Window sections.

Tutorial

If you are new to Keyboard Maestro, start the tutorial by clicking the Start Tutorial button in the Welcome window or by choosing the Help ➤ Tutorial menu and Keyboard Maestro will walk you through creating a simple macro.



Follow the instructions. Keyboard Maestro will hilight the location of the various buttons to help you quickly create a macro. You can even use the tutorial as a wizard to create a hot key triggered macro to perform any of Keyboard Maestro's many actions.

Macro Group Editor

To edit a Macro Group, select it and click the Edit button. Its details will be shown in the right hand column. You can edit its name, control which applications it is available in, and how it will be activated.

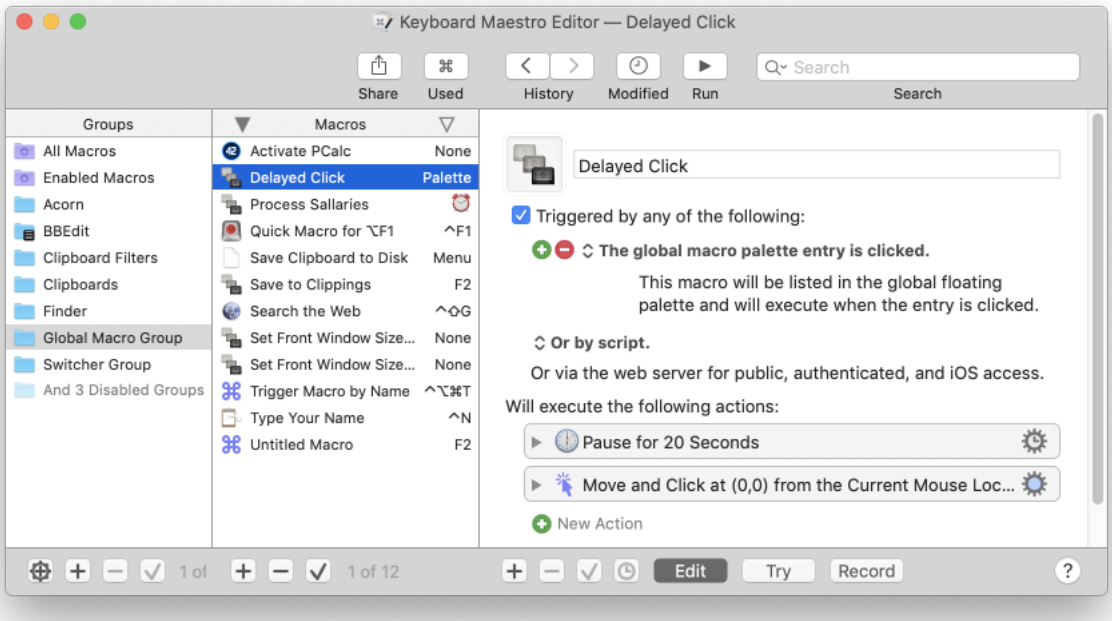
Typically a group's macros would be available everywhere (available in all applications), or it might be specific to a particular application (available in the following applications) in which case you might name the Macro Group after the application.

You can also configure the macro group to be activated only after a Hot Key press (either for a single use or toggled on and off), and whether to display the macros in a floating palette.

See also the Macros section.

Macro Editor Window

To edit a Macro, select it and click the Edit button. Its details will be shown in the right hand column. You can edit its name, add or remove triggers, and configure its action list.



To add a trigger, click the green ⊕ button and select the type of trigger. To remove a trigger, click the red ⊖ button.

To see how to execute this macro via a script, select from the "Or by script"" menu. How you can execute the macro remotely is also displayed.

To add an action, click the New Action button, or equivalently the + button below the detail view. This will show the lists of possible actions. Double click one or more of them to add actions to the action list for this macro.

You can also Copy and Paste actions, as well as drag them around to rearrange them. Use the - button and ✓ button to delete or enable/disable the selected actions.

You can try the selected actions by clicking the Try button.

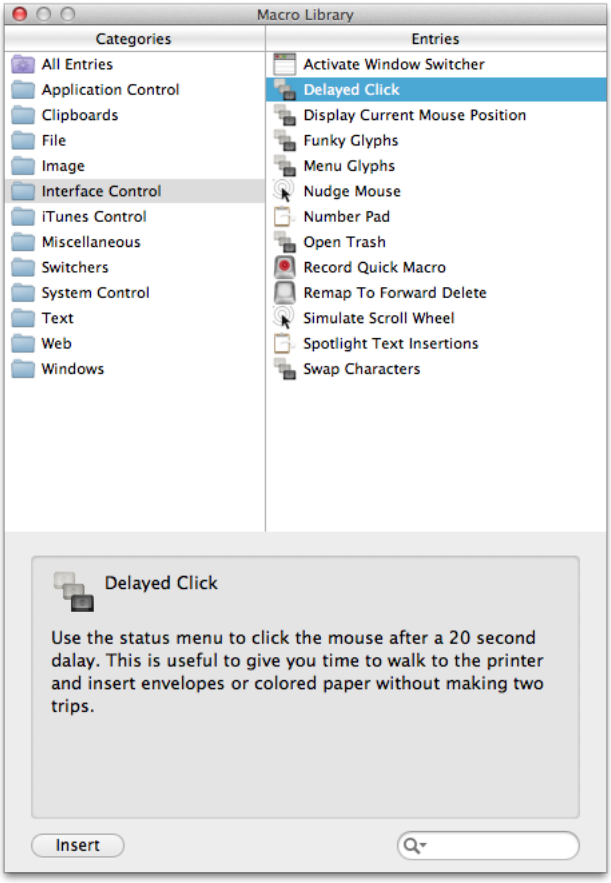
You can click the Record button to record your actions.

To learn more about creating or editing Macros, see the Macros section.

Macro Library Window

This window contains example and template macros you can add to your macro collection. You can use the macros as is, or edit them to customize them for your particular needs.

You get this window by choosing the Window ➤ Macro Library menu.



Each entry represents one or more macros, usually in a single macro group, but occasionally in more than one macro group. You can learn about them by selecting them, and then you can insert them into your macros by clicking the Insert button or by dragging them to a particular macro group (dragging is not available if the library entry represents more than one macro group as you can't drag to two macro groups).

You can add macros to your library by choosing the File ➤ Export as Macro Library menu and selecting the Add to My Macro Library checkbox. You can get Macro Library entries from us or from other Keyboard Maestro users and add them to you library by double clicking them or by choosing the File ➤ Import to Macro Library menu.

Remember to use caution when installing a macro or macro library from anyone – macros can potentially do a lot of damage and compromise the security of your Mac, so only install macros from trusted sources. When importing macros from a file they will be imported disabled (either the parent macro group will be disabled, or if it exists already, the macros themselves will be disabled) unless you hold the Option key down.

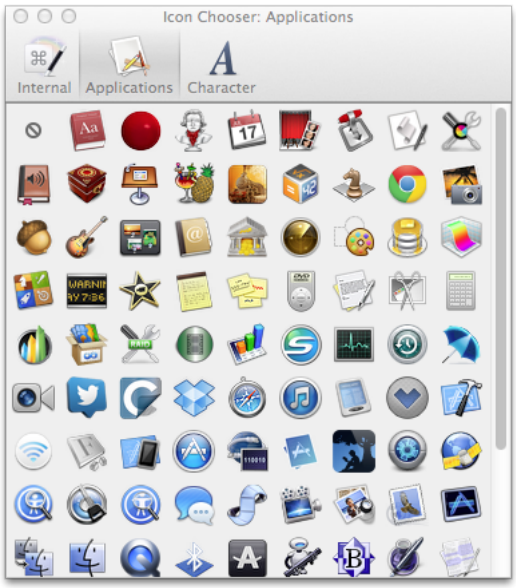
Icon Chooser Window

This window contains icons you can use to customize your macro and macro groups. You can click on an icon well in the macro or macro group editor and then show this window and select or create an icon for it.

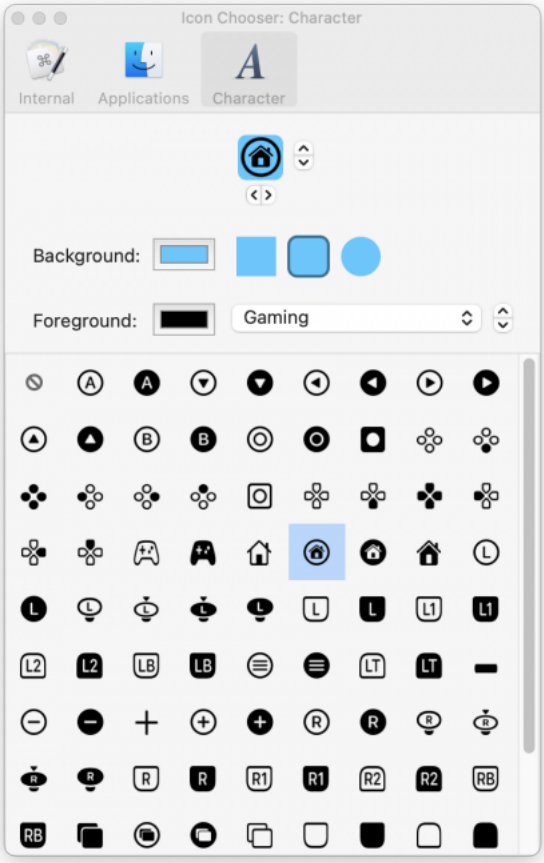
You get this window by choosing the Window ➤ Icon Chooser menu.

There are three types of icons available. Internal Keyboard Maestro icons:

application icons available on your Mac:



and icons you create by choosing a shape, color and optional character:



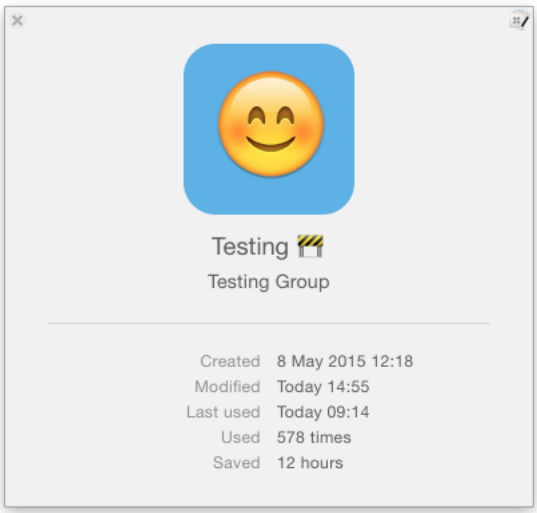
When creating icons, you can reference lots of characters, including SF Symbols in macOS 11+.

Alternatively, you can copy an image from anywhere and paste it in to the icon well (although Keyboard Maestro stores a small reference to the Icon Chooser icons, so that is much more efficient that storing the image in your macros as copy & paste will).

Macro Inspector Window

This window shows you information about the selected macro or macro group.

You get this window by choosing the Window ➤ Macro Inspector menu.



Mouse Display Window

This window shows you the mouse location relative to any corner of the main screen or front window.

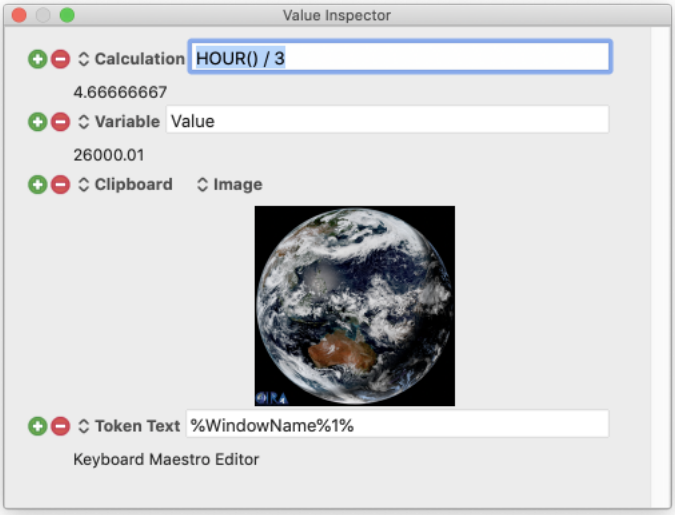
You get this window by choosing the Window ➤ Mouse Display menu.

You can click the button and then five seconds later Keyboard Maestro will lock the coordinates. You can click the button to copy the values, and you can change the relative corner even while the display is locked.

Value Inspector Window

This window shows you the value of the specified variables, clipboards, calculations or tokens.

You get this window by choosing the Window ➤ Value Inspector menu.

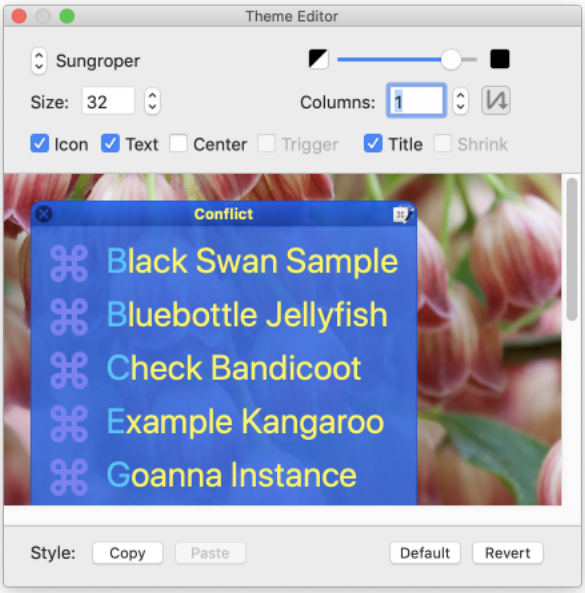


The contextual menu (control or right click in the window) will allow you to toggle whether the window is displayed only while the Keyboard Maestro editor is at the front, or whether the window floats above all applications.

Palette Theme Editor

This window allows you to configure the appearance of the various macro palette.

You can choose a style, the opacity of the palette, the size of the entries, the number of columns, whether the entries include the icon, the text and/or the trigger, and whether the palette shrinks when the mouse is not over it.



Recording Window

This window shows you when Keyboard Maestro is recording your actions.

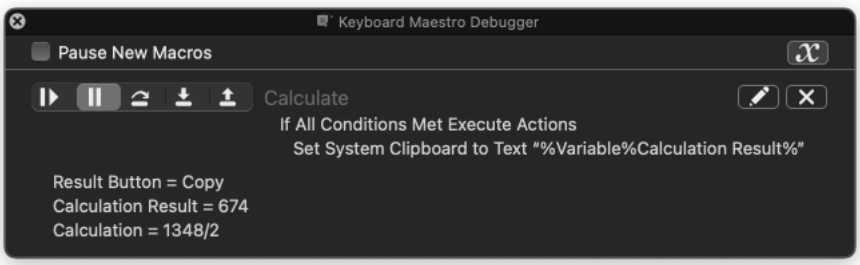
You get this window by clicking the Record button in the Macro Editor window or by triggering a Record Quick Macro action.

Clicking on this window will stop all recording, or you can pause recording, or add a 0.25 second Pause action to the current recording. Option-clicking the Clock button will turn on "real time" recording and Keyboard Maestro will record a pause between each action to simulate playback at approximately the same speed as the recording.

To learn more about recording, see the Recording section.

Macro Debugger

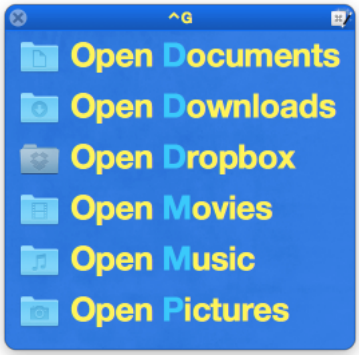
You get this window by choosing the Status Menu ➤ Start Debugging menu by triggering a Debugger Start action.



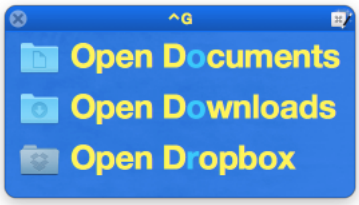
To learn more about debugging, see the Macro Debugger section.

Conflict Palette

You get a conflict palette when a Hot Key would trigger two or more actions.



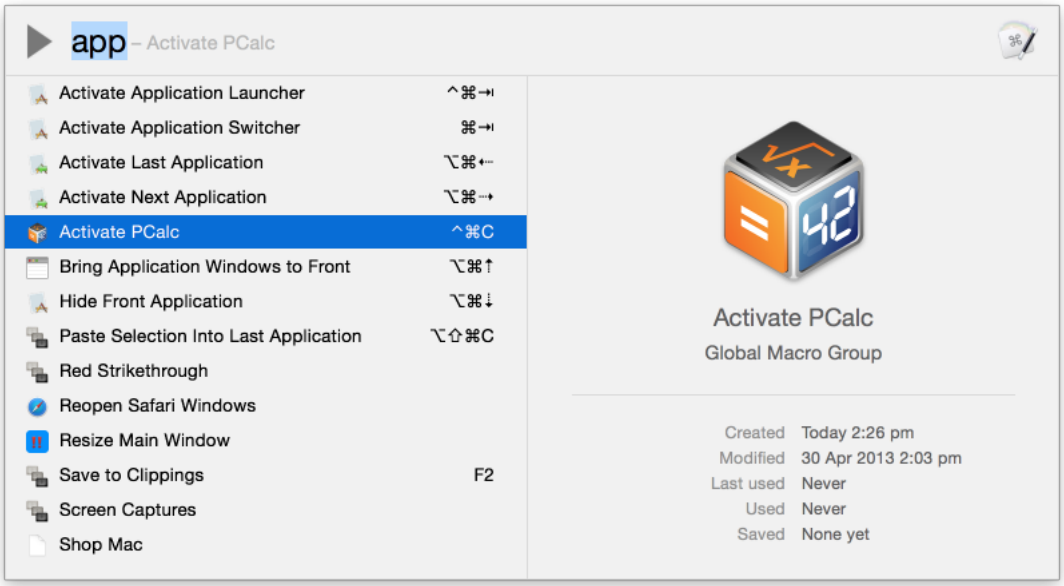
Note how the distinguisihing characters are highlighted – pressing those keys will select the desired trigger, or filter the list so that only those macros remain:



this continues until only one macro remains which is immediately executed.

Trigger Macro by Name

You get this window by executing the Trigger Macro by Name action.



Type a string to filter the macros.

Note that the filtering is not just by macro name.

Application Launcher Window

This window lets you launch applications.

You get this window by triggering the Activate Application Launcher macro.

To learn more about the Application Launcher, see the Application Launcher section.

Application Switcher Window

This window lets you switch between active applications, as well as quit or hide applications or quickly launch frequently used applications.

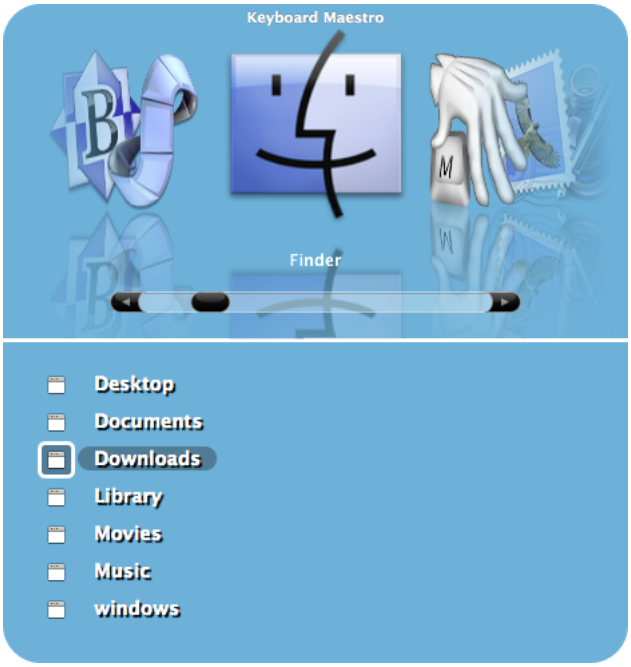
You get this window by triggering the Activate Application Switcher macro.

To learn more about the Application Switcher, see the Application Switcher section.

Window Switcher Window

This window lets you switch between windows in the current application, as well as close or minimize windows.

You get this window by triggering the Activate Window Switcher macro.

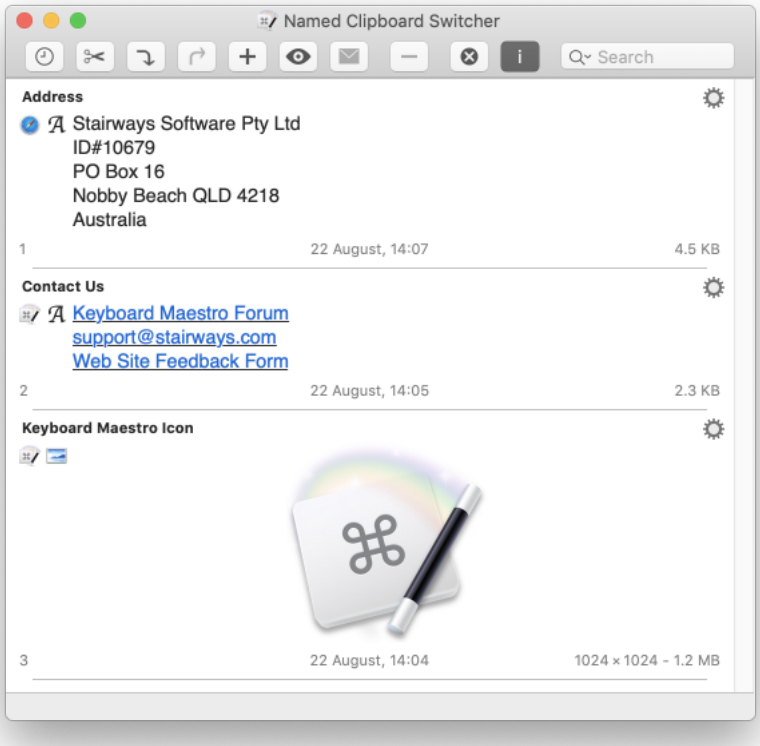


To learn more about the Window Switcher, see the Window Switcher section.

Named Clipboard Switcher Window

This window lets you select between named clipboards to Cut, Copy or Paste to/from.

You get this window by triggering one of the Activate Clipboard Switcher macros.



Select something and select a named clipboard and click the Cut button or Copy button to cut/copy to a named clipboard. Select a named clipboard and click the Paste button button to paste a named clipboard. Click the + button to create a new named clipboard. Select a named clipboard and click the Quick Look button to view it. Select a named clipboard and click the - button to delete it.

Click the x button to toggle whether the window should close after an action.

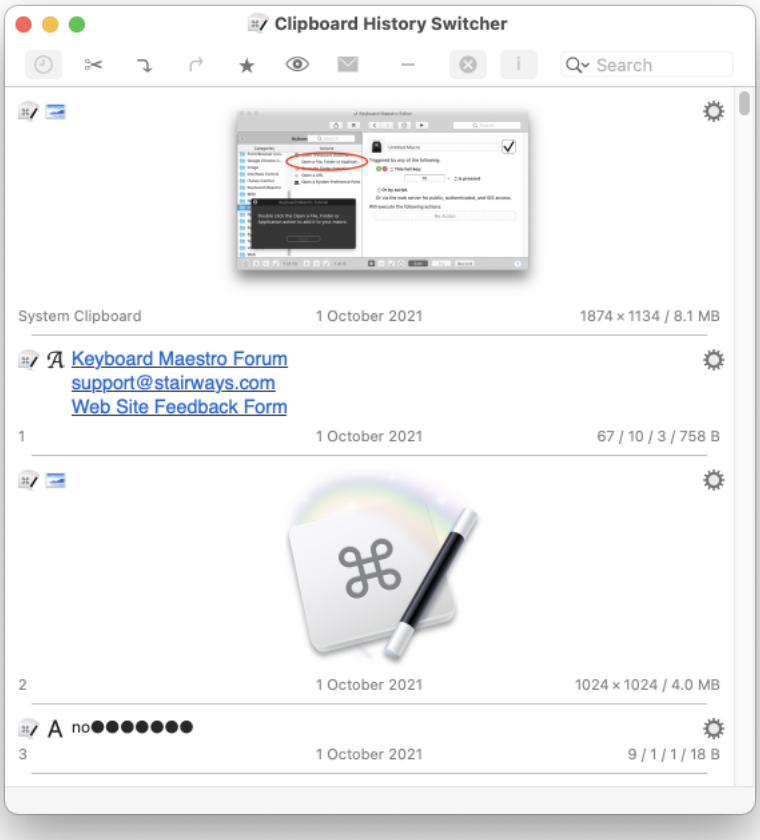
Use the search field to filter the named clipboards.

To learn more about the Clipboard Switcher, see the Named Clipboard Switcher section.

Clipboard History Switcher Window

This window lets you paste from your clipboard history of items that you have previously cut or copied.

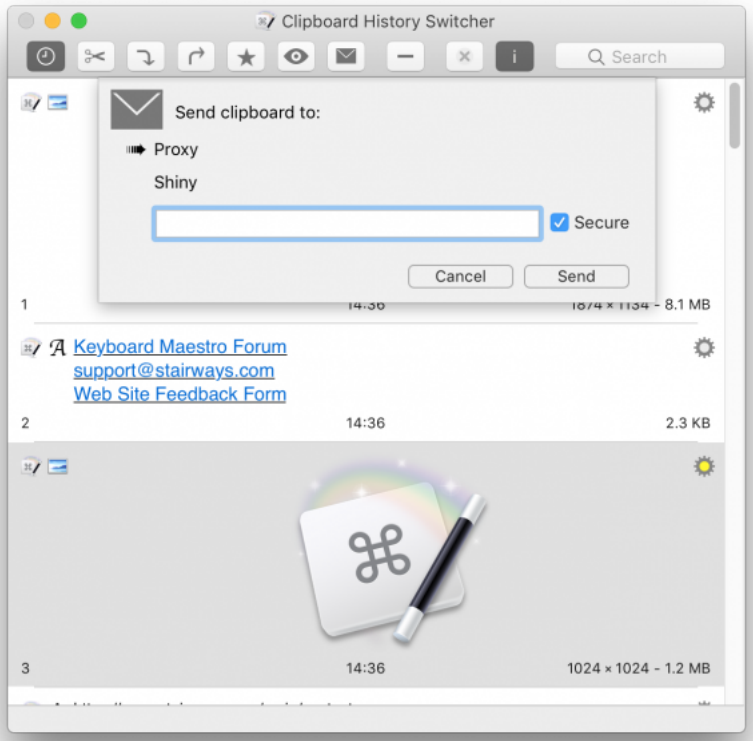
You get this window by triggering the Clipboard History Switcher macro.



Cut or copy something and it will appear in this list. Select an item and click the Paste button button to paste a named clipboard. Select an item and click the ★ button (or press Command-L) to mark it as a favorite, or click the ✉ button (or press Command-S) to send it to another Mac. Select an item and click the - button to delete it (this is useful if you want to delete a password or other sensitive information).

To view an item more fully, select it and click the Quick Look button (or press Space) to display the Quick Look window.

To send a clipboard to another Mac running Keyboard Maestro, click the ✉ button to display the sending window.



Select the desired local destination, or type a host:port destination to send it to a remote Mac, and click the send button. Keyboard Maestro will keep trying to send to the Mac even if it can't connect right now, so as long as both Macs are connected to the Internet eventually, the clipboard should get through. While a clipboard is being sent, it is marked with a progress icon. If a clipboard is marked to be sent in the future, it is marked with a . Once it has been successfully sent, it is marked with a . If it fails to be sent, and Keyboard Maestro has given up, it is marked with a !. Clipboards that have been received are marked with a ✉.

Click the x button to toggle whether the window should close after an action.

Use the search field to filter the named clipboards.

To learn more about the Clipboard History Switcher, see the Clipboard History Switcher section.

Preferences Window

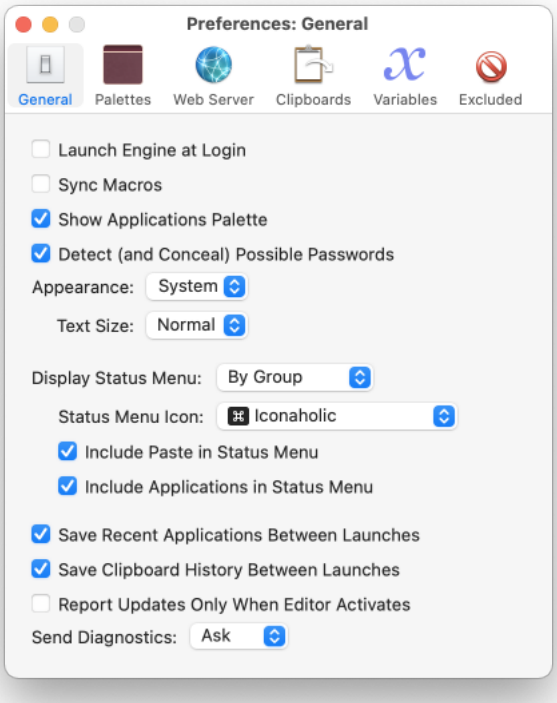
This window lets you configure Keyboard Maestro.

You get this window by launching Keyboard Maestro and choosing the Keyboard Maestro ➤ Preferences menu.

To learn more about the Preferences, see the Preferences section.

Preferences General Pane

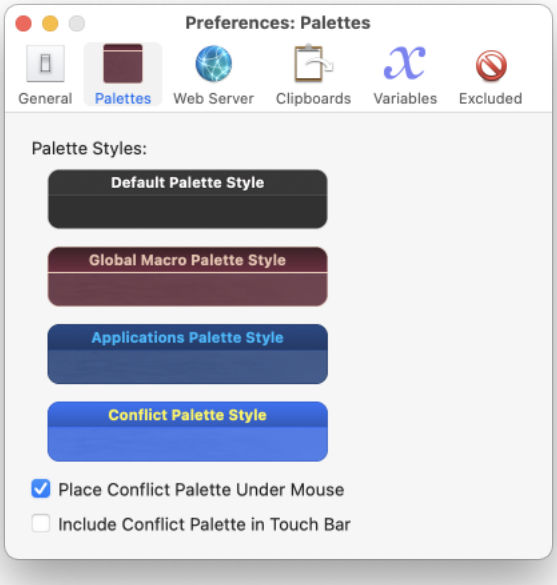
This window pane lets you configure general preferences.



Preferences Palettes Pane

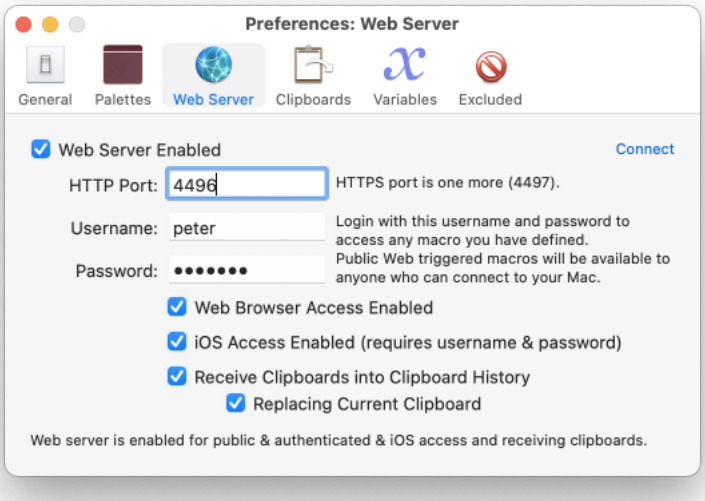
This window pane lets you configure the default palettes style, as well as the global and conflict palette styles.

You can also configure whether the conflict palette appears under the mouse.



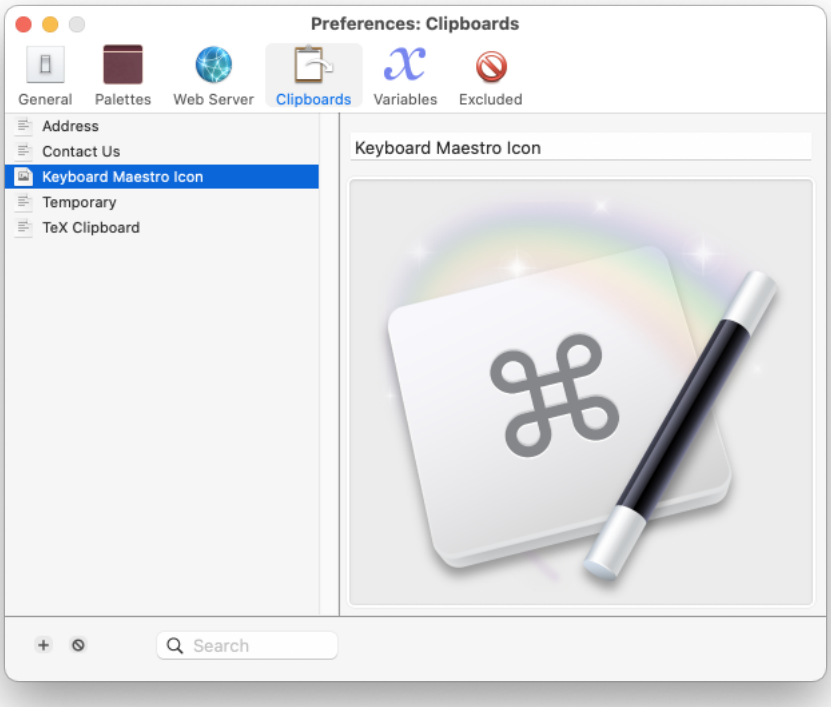
Preferences Web Server Pane

This window pane lets you configure the built-in web server which enables remote execution of macros.



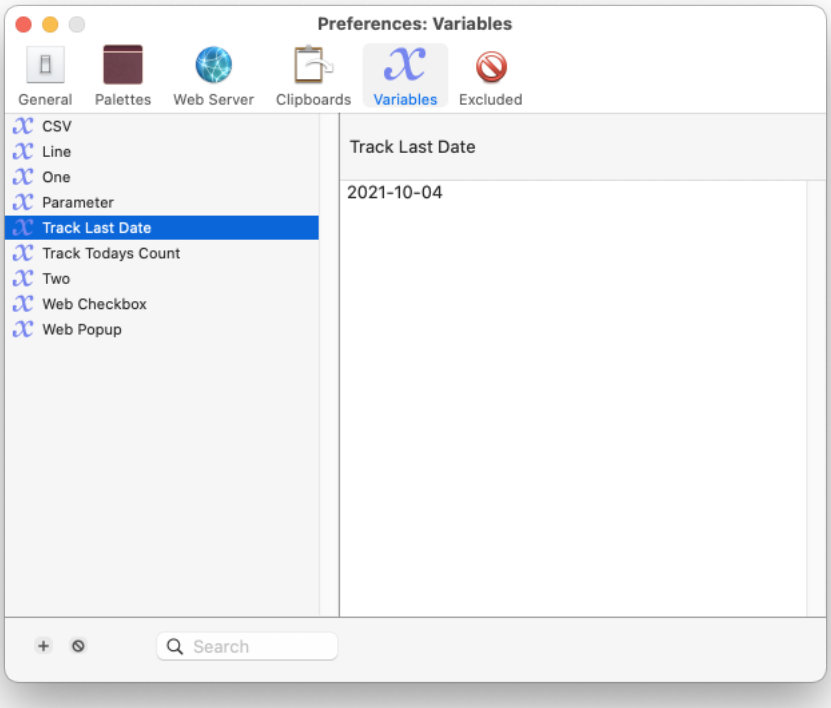
Preferences Clipboards Pane

This window pane lets you add, remove and rename Named Clipboards and see and change their values.



Preferences Variables Pane

This window pane lets you add and remove variables, and see and change their values.

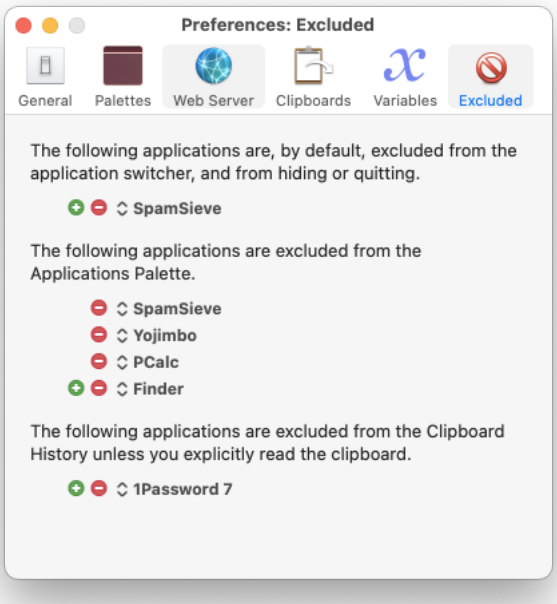


Variables listed in the Variables Panel include:

- Variables with values.
- All variables referenced directly in any action (although not as tokens).
- Minus all variables whose value has been set to "%Delete%" (this is provided for people who prefer not to see certain variable names in the pane or in variable insertion menus).

Preferences Exclude Pane

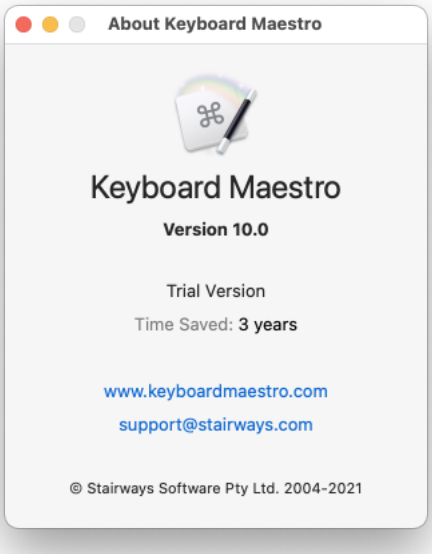
This window pane lets you add and remove applications from the global Excluded Applications list.



About Window Pane

This window shows you the version of this copy of Keyboard Maestro, to whom it is registered, and allows you to visit the web site.

You get this window by launching Keyboard Maestro and choosing the Keyboard Maestro ➤ About Keyboard Maestro menu.



Menus

Below is a complete list of all menus. Note that in some cases you must hold down the Option Key ⌥ (or Alt or Opt) to see the menu item. For example, the "Expand All Actions" item in the Actions menu.

Keyboard Maestro

The *Keyboard Maestro* menu contains menu items relating to the Keyboard Maestro application as a whole.

About Keyboard Maestro

The *About Keyboard Maestro* command in the *Keyboard Maestro* menu displays the About Keyboard Maestro window.

Purchase Keyboard Maestro

The *Purchase Keyboard Maestro* command in the *Keyboard Maestro* menu lets you purchase Keyboard Maestro online.

Purchase Keyboard Maestro Upgrade

The *Purchase Keyboard Maestro Upgrade* command in the *Keyboard Maestro* menu lets you purchase an upgrade to Keyboard Maestro online.

Register Keyboard Maestro

The *Register Keyboard Maestro* command in the *Keyboard Maestro* menu displays the serial number entry window allowing you to enter your username (email address) and serial number. Make sure you enter them exactly as sent to you. If your email address shows up as "Registered to" in the About Keyboard Maestro box, then you are already registered.

Check For Update

The *Check For Update* command in the *Keyboard Maestro* menu checks to see if there are any updates to Keyboard Maestro and offers to download and install them if there are.

Preferences

The *Preferences* command in the *Keyboard Maestro* menu displays the Preferences window.

Services

The *Services* command in the *Keyboard Maestro* menu is used to perform Mac OS X Services which are shared functions available across multiple applications. You can learn more about Mac OS X Services from your Mac OS X documentation, and you can install new services which will work with Keyboard Maestro. Keyboard Maestro includes full support for Services, so relevant Services on your system are available in Keyboard Maestro. You can also export a macro as a Text Service to have it available here (and for other applications).

Hide Keyboard Maestro

The *Hide Keyboard Maestro* command in the *Keyboard Maestro* menu will hide the Keyboard Maestro application and all its windows. Click on Keyboard Maestro's Dock icon or choose Show All to show Keyboard Maestro again. Note that generally you should just quit the Keyboard Maestro editor when you are not modifying your macros.

Hide Others

The *Hide Others* command in the *Keyboard Maestro* menu will hide all other applications. Choose Show All to show them again.

Show All

The *Show All* command in the *Keyboard Maestro* menu will show all hidden applications.

Quit Keyboard Maestro Editor

The *Quit Keyboard Maestro Editor* command in the *Keyboard Maestro* menu will Quit Keyboard Maestro. The Keyboard Maestro Engine will remain running and all enabled Keyboard Maestro features will continue to operate (unless you have specifically quit the Keyboard Maestro Engine).

File

The *File* menu is where you import or export Macros or launch or quit the Keyboard Maestro Engine.

New Smart Group

The *New Smart Group* command in the *File* menu creates and starts editing a new smart group.

New Macro Group

The *New Macro Group* command in the *File* menu creates and starts editing a new macro group.

New Macro

The *New Macro* command in the *File* menu creates and starts editing a new macro.

New Editor Window

The *New Editor Window* command in the *File* menu creates a new editor window. Closing the last editor window will quit the Keyboard Maestro editor (the Keyboard Maestro Engine will continue operating as usual).

Close

The *Close* command in the *File* menu closes the front window.

Close All

The *Close All* command in the *File* menu closes all the windows and quits the Keyboard Maestro editor (the Keyboard Maestro Engine will continue operating as usual). Hold the Option key down to select this menu item.

Run Macro

The *Run Macro* command in the *File* menu runs the currently selected macro.

Share

The *Share* menu in the *File* menu lets you share the selected macro group, macro or actions with any appropriate sharing service on your Mac, including sharing to the forum [https://forum.keyboardmaestro.com]. If you hold the Shift key down before starting your menu selection you can share just an image of the selection, which will allow you to share to services that only allow selection of images.

Export Actions

The *Export Actions* command in the *Export* sub-menu in the *File* menu exports the selected actions to a file that you can import later or on another Mac or send to others.

Export Macros

The *Export Macros* command in the *Export* sub-menu in the *File* menu exports the selected macros to a file you can import on another Mac. This is one way to transfer macros from one Mac to another or to share macros with friends or colleagues.

If a Macro Group is selected and **no** macros are selected then the entire Macro Group is exported as a single `.kmmacros` file.

Export as Macro Library

The *Export as Macro Library* command in the *Export* sub-menu in the *File* menu exports the selected macros to a library file that you can share with others. If you create any interesting macros please consider sending them to us and we will make them available on our web site or in a future version of Keyboard Maestro.

See also the Macro Library section.

Note that you should generally only export as library if you want to import the macros multiple times. Otherwise just exporting the macros is sufficient.

Export as Folder

The *Export as Folder* command in the *Export* sub-menu in the *File* menu exports the selected macros to a folder containing macro group folders and individual macro export files. You can use this to preserve your macros in a version history structure or for archiving purposes.

Export All Macros as Folder

The *Export All Macros as Folder* command in the *Export* sub-menu in the *File* menu exports all of your macros to a folder containing macro group folders and individual macro export files. You can use this to preserve all of your macros in a version history structure or for archiving purposes.

Export as Trigger File

The *Export as Trigger File* command in the *Export* sub-menu in the *File* menu lets you save a file that will trigger a macro when opened (that is, a file you can open in the Finder to trigger a macro).

Export as Text Service

The *Export as Text Service* command in the *Export* sub-menu in the *File* menu lets you save a Text Service that will trigger a macro when selected in the Services menu. The incoming text is available in the %TriggerValue% token.

Export as Finder Quick Action

The *Export as Finder Quick Action* command in the *Export* sub-menu in the *File* menu lets you save a Finder Quick Action that will trigger a macro when selected in the Quick Action contextual menu. The incoming file path is available in the %TriggerValue% token.

Import Actions

The *Import Actions* command in the *Import* sub-menu in the *File* menu lets you select a saved action file and imports the actions it contains into the current macro.

Import Macros Safely

The *Import Macros Safely* command in the *Import* sub-menu in the *File* menu lets you select a saved macro file and imports the macros it contains, disabling them (either disabling created macro groups, or disabling the contained macros as appropriate).

If you do not completely trust the source of the macros, you should import them this way.

Import Macros

The *Import Macros* command in the *Import* sub-menu in the *File* menu lets you select a saved macro file and imports the macros it contains. Hold the Option key down to select this menu item. Macros imported this way could trigger immediately, so use this command only if you completely trust the source of the macros.

Import to Macro Library

The *Import to Macro Library* command in the *Import* sub-menu in the *File* menu lets you import a shared macro library file into your macro library. Macros in your library are not active, but can be added into one or more macro groups to become active.

See also the Macro Library section.

Revert Macros

The *Revert Macros* menu in the *File* menu lets you revert to a previous version of your macros. If you find you have really messed up your macros, you can restore your macros to a backup version of how they were when you first launched Keyboard Maestro, or how they were yesterday or even several days ago.

Start Syncing Macros

The *Start Syncing Macros* command in the *File* menu lets you start syncing your macros with another Mac.

See also the Macro Syncing section.

Reveal Macro Sync File

The *Reveal Macro Sync File* command in the *File* menu lets you see where your macro sync file is.

See also the Macro Syncing section.

Stop Syncing Macros

The *Stop Syncing Macros* command in the *File* menu lets you stop syncing your macros.

See also the Macro Syncing section.

Launch Engine

The *Launch Engine* command in the *File* menu lets you start the Keyboard Maestro Engine manually. The Keyboard Maestro Engine performs all the Macro, Application Switcher, Window Switcher and Clipboard Switcher functions even while Keyboard Maestro itself is not running. It is launched automatically as a Startup Item when you login (assuming you have enabled that in the Preferences window) or any time you launch Keyboard Maestro. If it is not running for any reason you can start it manually with this command. This menu item only appears while the Keyboard Maestro Engine is not running.

Quit Engine

The *Quit Engine* command in the *File* menu lets you quit the Keyboard Maestro Engine. The Keyboard Maestro Engine performs all the Macro, Application Switcher, Window Switcher and Clipboard Switcher functions even while Keyboard Maestro itself is not running. It is launched automatically as a Startup Item when you login (assuming you have enabled that in the Preferences window). If you quit the Keyboard Maestro Engine these functions will no longer operate. This menu item only appears while the Keyboard Maestro Engine is running.

Edit

The *Edit* menu contains menu items relating to text and selections.

Undo

The *Undo* command in the *Edit* menu undoes the previous command.

Redo

The *Redo* command in the *Edit* menu redoes the previous undone command.

Cut

The *Cut* command in the *Edit* menu copies the current selection to the system clipboard and then deletes the selection.

Copy

The *Copy* command in the *Edit* menu copies the current selection to the system clipboard.

Copy as

The *Copy as* sub-menu in the *Edit* menu menu allows you to copy the selection in a variety of formats.

Copy as Text

The *Copy as Text* command in the *Copy as* sub-menu in the *Edit* menu menu allows you to copy the selected macro or actions as styled text.

Copy as Image

The *Copy as Image* command in the *Copy as* sub-menu in the *Edit* menu menu allows you to copy the selected macro or actions as an image.

Copy as XML

The *Copy as XML* command in the *Copy as* sub-menu in the *Edit* menu menu allows you to copy the selected macro or actions as XML.

Copy UUID

The *Copy UUID* command in the *Copy as* sub-menu in the *Edit* menu menu allows you to copy the selected macro or macro group's UUID (Universally Unique ID).

Copy as Execute a Macro Action

The *Copy as Execute a Macro Action* command in the *Copy as* sub-menu in the *Edit* menu menu allows you to copy the selected macro as an Execute a Macro action.

Copy as Execute a Subroutine Action

The *Copy as Execute a Subroutine Action* command in the *Copy as* sub-menu in the *Edit* menu menu allows you to copy the selected macro as an Execute a Subroutine action.

Copy as Mark Macro Action

The *Copy as Mark Macro Action* command in the *Copy as* sub-menu in the *Edit* menu menu allows you to copy the selected macro as a Mark Macro action.

Copy as Set Macro Enabled Action

The *Copy as Set Macro Enabled Action* command in the *Copy as* sub-menu in the *Edit* menu menu allows you to copy the selected macro as a Set Macro Enable action.

Copy as Set Macro Group Enabled Action

The *Copy as Set Macro Group Enabled Action* command in the *Copy as* sub-menu in the *Edit* menu menu allows you to copy the selected macro as a Set Macro Group Enable action.

Copy as Toggle Macro Group Action

The *Copy as Toggle Macro Group Action* command in the *Copy as* sub-menu in the *Edit* menu menu allows you to copy the selected macro as a Toggle Macro Group Activation action.

Paste

The *Paste* command in the *Edit* menu pastes the current system clipboard into the current selection.

Delete

The *Delete* command in the *Edit* menu deletes the current selection.

Select All

The *Select All* command in the *Edit* menu selects all text or items.

Deselect All

The *Deselect All* command in the *Edit* menu deselects all text or items. Hold the Option key down to select this menu item.

Duplicate

The *Duplicate* command in the *Edit* menu duplicates the selected items.

Make Alias

The *Make Alias* command in the *Edit* menu makes an alias to the selected macros. It creates a new macro that just executes the original macro.

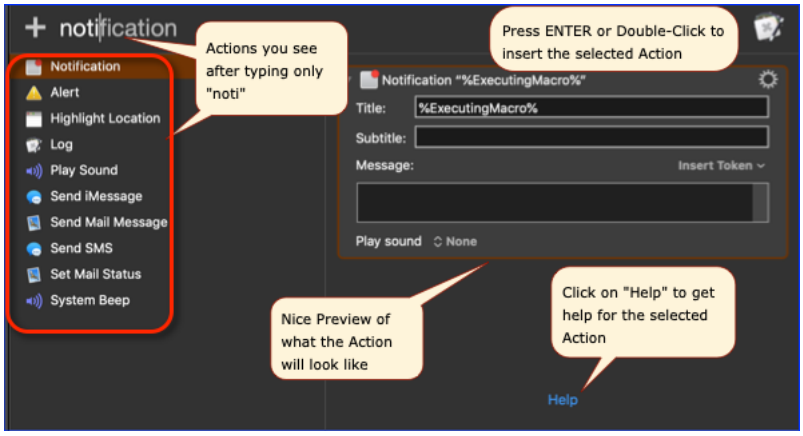
You may also use ⌘L or Drag/Drop with ⌘⌥ to another Macro Group.

Insert Action

The *Insert Action* command in the *Edit* menu lets you insert any of the many actions available in Keyboard Maestro.

Insert Action by Name

The *By Name* command in the *Edit* ➤ *Insert Action* menu lets you insert any of the many actions available in Keyboard Maestro by name.



Hold the Option key down while double clicking or pressing Return to insert the action above the selected action.

Insert Function

The *Insert Function* sub-menu in the *Edit* menu lets you insert any of the many functions available in Keyboard Maestro. The functions will be inserted, together with an example of any parameters it takes.

Most numeric fields in Keyboard Maestro can process an expression containing these functions. These fields are indicated either by having up/down steppers, or a C in the field while editing.

Hold the Option key down to get help on the function.

Insert Function by Name

The *By Name* command in the *Edit* ➤ *Insert Function* menu lets you insert any of the many functions available in Keyboard Maestro by name.

Insert Token

The *Insert Token* sub-menu in the *Edit* menu lets you insert any of the many Tokens available in Keyboard Maestro. The token will be inserted, together with an example format of any parameters it takes.

Most text fields in Keyboard Maestro can process Tokens. These fields are indicated by having a T in the field while editing.

Hold the Option key down to get help on the token.

Hold the Shift key down to get show the raw token.

Insert Token by Name

The *By Name* command in the *Edit* ➤ *Insert Token* menu lets you insert any of the many tokens available in Keyboard Maestro by name.

Insert Variable

The *Insert Variable* command in the *Edit* menu lets you insert any of your variables.

Insert Variable by Name

The *By Name* command in the *Edit* ➤ *Insert Variable* menu lets you insert any of your variables by name.

Insert ICU Date Field

The *Insert ICU Date Field* sub-menu in the *Edit* menu lets you insert any of the many fields available in %ICUDateTime% token.

Hold the Shift key down to get show the raw field.

Insert ICU Date Fieldby Name

The *By Name* command in the *Edit* ➤ *Insert ICU Date Field* menu lets you insert any of the many fields available in %ICUDateTime% token in Keyboard Maestro by name.

Automatic Completions

The *Automatic Completions* command in the *Edit* menu lets you toggle automatic text completion. When off, you can still complete variables, tokes, and functions by pressing F5.

Find

The *Find* menu in the *Edit* menu lets find things in Keyboard Maestro.

The *Find…* menu item will select the search field in the Keyboard Maestro editor window.

The *Find in All Macros…* menu item will select the search field in the Keyboard Maestro editor window and select the All Macros Smart Group to search all the macros.

View

The *View* menu contains menu items relating to display and actions.

Sort Macros by Name

The *Name* command in the *View* ➤ *Sort Macros by* menu sorts the macros in the main window by name.

Sort Macros by Trigger

The *Trigger* command in the *View* ➤ *Sort Macros by* menu sorts the macros in the main window by trigger. This is useful to see what hot keys are available, or to group all Typed String triggers together.

Sort Macros by Date Created

The *Date Created* command in the *View* ➤ *Sort Macros by* menu sorts the macros in the main window by the date created.

Sort Macros by Date Modified

The *Date Modified* command in the *View* ➤ *Sort Macros by* menu sorts the macros in the main window by the date modified (most recently modified at the top). This is useful to see what macros you have recently modified. You can also use the ◷ button at the top of the macro editor to show the recently modified macros.

Sort Macros by Date Used

The *Date Used* command in the *View* ➤ *Sort Macros by* menu sorts the macros in the main window by date used (most recently executed at the top). This is useful to see what macros you have recently executed, especially if you think one might have been executed inadvertently. You can also use the ⌘ button at the top of the macro editor to show the recently used macros.

Sort Macros by Size

The *Size* command in the *View* ➤ *Sort Macros by* menu sorts the macros in the main window by approximate storage size, with the largest at the top. Typically it is not the number of actions in a macro that makrs a macro large, instead it is usually included images. If your macro file gets very large, this can slow the performance of the Keyboard Maestro editor (though generally it will not affect the performance of the Keyboard Maestro Engine). Sorting the macros by size can find the larger macros.

Sort Macros by Use Count

The *Use Count* command in the *View* ➤ *Sort Macros by* menu sorts the macros in the main window by use count (most used macros at the top).

Sort Macros by Time Saved

The *Time Saved* command in the * View ➤ Sort Macros by* menu sorts the macros in the main window by time saved (most time saved at the top).

Disabled to Bottom

The *Disabled to Bottom* command in the *View* ➤ *Sort Macros by* menu sorts the disabled macros to the bottom of the macro list.

Select Groups Column

The *Select Groups Column* command in the *View* menu focuses on the Macro Groups column.

Select Macros Column

The *Select Group Column* command in the *View* menu focuses on the Macros column.

Select Macro

The *Select Macro* menu in the *View* menu lets you select a recently edited or used macro.

Select Macro by Name

The *By Name* menu in the *View* ➤ *Select Macro* menu lets you select a macro by name.

Select Macro Last Used

The *Last Used* menu in the *View* ➤ *Select Macro* menu lets you select the most recently triggered macro.

Select Macro Last Edited

The *Last Edited* menu in the *View* ➤ *Select Macro* menu lets you select the most recently edited macro.

Select Macro Previous Edited

The *Previous Edited* menu in the *View* ➤ *Select Macro* menu lets you select the previous edited macro (edited macros form a list, and this moves backward along the list).

Select Macro Next Edited

The *Next Edited* menu in the *View* ➤ *Select Macro* menu lets you select the next edited macro (edited macros form a list, and this moves forward along the list).

Select Last Aborted Action

The *Select Last Aborted Action* menu in the *View* menu selects the macro and the last action that failed and aborted a macro.

Reveal Parent Group

The *Reveal Parent Group* command in the *View* menu selects the parent macro group of the currently selected macros. If you are editing a macro and the parent macro group is not the only selected macro group, then the parent macro groups's name is shown in the top right corner of the window. Clicking on this name will also reveal the parent macro group.

Show Disabled Macro Groups

The *Show Disabled Macro Groups* command in the *View* menu shows all macro groups (enabled or disabled) in the Macro Groups column. You can hide all the disabled Macro Groups with the *Hide Disabled Macro Groups* command.

Hide Disabled Macro Groups

The *Hide Disabled Macro Groups* command in the *View* menu hides all disabled macro groups in the Macro Groups column. Instead, a single placeholder group is shown that "contains" all the disabled Macro Groups. You can double click the Disabled Macro Groups placeholder to temporarily reveal the disabled macro groups. You can show all the disabled Macro Groups with the *Show Disabled Macro Groups* command.

Evaluate Condition Results

The *Evaluate Condition Results* command in the *View* menu toggles whether the editor evaluates conditions in macros it is displaying. If it does, macros with conditions will display their current value in real time in the editor (eg "(currently true)"). If not, they will display "(click for result)" and you can click on that to enable the Evaluate Condition Results facility.

Note that evaluating results may have a performance issue if the results are complex to evaluate (eg searching the screen).

This facility is turned off when you import a macro safely to ensure that scripts in conditions do not get to execute before you have a chance to verify their safety.

Enable

The *Enable* command in the *View* menu enables Macro Groups, Macros or Actions as appropriate.

Disable

The *Disable* command in the *View* menu disables Macro Groups, Macros or Actions as appropriate.

Start Editing Macros

The *Start Editing Macros* command in the *View* menu turns on edit mode. Edit mode allows you to modify macros and macro groups. You may prefer to leave it on permanently, or you may like to turn it off when you are not making changes to give a more concise and visually appealing view of the macros and macro groups.

Stop Editing Macros

The *Stop Editing Macros* command in the *View* menu turns off edit mode.

Rename

The *Rename* command in the *View* menu edits the name of the currently selected Macro Group or Macro.

Record

The *Record* command in the *View* menu toggles recording on and off. It is available while editing a macro. Hold the option key down to record without a delay.

Record Without Delay

The *Record Without Delay* command in the *View* menu toggles recording on and off immediately (without the normal short countdown). It is available while editing a macro. Hold the shift key down to select the menu.

Enter Full Screen

The *Enter Full Screen* command in the *View* menu will expand the window to full screen.

Actions

The *Actions* menu contains menu items relating to actions. These items have analogs in the action gear ⚙ menu and contextual menu for actions, and some menu items appear only in the appropriate menu.

Show Actions

The *Show Actions* command in the *Actions* menu shows the action list. It is available while editing a macro. Double click or drag actions from the action list to add them to your macro. This menu toggles to Hide Actions when the action list is already showing. Alternatively you can choose *Insert Action by Name* from the *Edit* menu to insert actions by name or use the *Insert Action* menu in the *Edit* to select actions.

Try

The *Try* command in the *Actions* menu tries the selected actions. It is available while editing a macro.

Enable

The *Enable* command in the *Actions* menu enables actions.

Disable

The *Disable* command in the *Actions* menu disables actions.

Rename

The *Rename* command in the *Actions* menu renames actions.

Set Color

The *Set Color* menu in the *Actions* menu sets the color of actions.

Note: A blue striped color is reserved to mark the action as a match for the current search.

Set Note

The *Set Note* command in the *Actions* menu sets the notes of actions.

Paste Above

The *Paste Above* command in the *Actions* menu pastes previously copied actions above the selected actions.

Paste Below

The *Paste Below* command in the *Actions* menu pastes previously copied actions below the selected actions.

Paste Replacing

The *Paste Replacing* command in the *Actions* menu pastes previously copied actions, replacing the selected actions.

Add to Favorites

The *Add to Favorites* command in the the *Actions* menu adds the currently selected action(s) as a Favorite action which you can insert later. The favorite action saves all the current action configuration.

Engroup

The *Engroup* menu in the *Actions* lets you enclose the selected actions within a container action such as a *Group*, *Repeat*, *If Then Else* or other control flow action. You can also turn the selected macros into a sub-macro.

Degroup

The *Degroup* command in the *Actions* removes the selected actions from their parent container action, and removes the container action if it is now empty.

Set Action Timeout

The *Set Action Timeout* command in the *Actions* menu sets the timeout for the selected action.

Timeout Aborts Macro

The *Timeout Aborts Macro* command in the *Actions* menu controls whether the macro is aborted if the action is timed out.

Notify on Timeout

The *Notify on Timeout* command in the *Actions* menu controls whether you are notified if the action is timed out.

Failure Aborts Macro

The *Failure Aborts Macro* command in the *Actions* menu controls whether the macro is aborted if the action fails.

Notify on Failure

The *Notify on Failure* command in the *Actions* menu controls whether you are notified if the action fails.

Expand Action

The *Expand Action* command in the *View* menu expands (discloses) the selected actions. Hold the option key down to expand all actions in a given sublist.

Expand All Actions

The *Expand All Actions* command in the *View* menu expands (discloses) the selected actions and all other actions in the same sublist. Hold the option key down to select this command.

Collapse Action

The *Collapse Action* command in the *View* menu collapses the selected actions. Hold the option key down to collapse all actions in a given sublist.

Collapse All Actions

The *Collapse All Actions* command in the *View* menu collapses the selected actions and all other actions in the same sublist. Hold the option key down to select this command.

Parent Action

The *Parent Action* command in the *View* menu moves the selection to the containing action.

Enter Action

The *Enter Action* command in the *View* menu moves the selection to the contained action.

Help

The *Help* command in the *View* menu displays the wiki help for the selected action.

Window

The *Window* menu contains menu items relating to windows.

Minimize

The *Minimize* command in the *Window* menu minimizes the front window.

Zoom

The *Zoom* command in the *Window* menu zooms the front window.

Keyboard Maestro Editor

The *Keyboard Maestro Editor* command in the *Window* menu brings the frontmost Keyboard Maestro macro editing window to the front.

Macro Library

The *Macro Library* command in the *Window* menu shows or hides the macro library.

See also the Macro Library section.

Icon Chooser

The *Icon Chooser* command in the *Window* menu shows or hides the Icon Chooser.

See also the Icon Chooser section.

Macro Inspector

The *Macro Inspector* command in the *Window* menu shows or hides the Macro Inspector.

See also the Macro Inspector Window section.

Mouse Display

The *Mouse Display* command in the *Window* menu shows or hides the Mouse Display, which shows the mouse location in real time.

See also the Mouse Display Window section.

Value Inspector

The *Value Inspector* command in the *Window* menu shows or hides the Value Inspector, which shows values of variables, tokens, calculations, or clipboards that you can select.

See also the Value Inspector Window section.

Bring All to Front

The *Bring All to Front* command in the *Window* menu brings all Keyboard Maestro windows to the front.

Help

The *Help* menu contains menu items relating to Help.
You can use the search field in this menu to find help on the wiki.

Interactive Help

The *Interactive Help* command in the *Help* menu starts the in-application assistance system. If you're having trouble with something not working the way you expect, this system may help you find out where the problem is, or offer suggestions for other sources os assistance.

Keyboard Maestro User Manual

The *Keyboard Maestro User Manual* command in the *Help* menu displays the Keyboard Maestro user manual on the Keyboard Maestro Wiki.

Keyboard Maestro Quick Start

The *Keyboard Maestro Quick Start* command in the *Help* menu displays the Keyboard Maestro quick start help web page, which quickly gets you up to speed in using Keyboard Maestro.

Welcome to Keyboard Maestro

The *Welcome to Keyboard Maestro* command in the *Help* menu displays the Welcome message, giving you a quick overview of what support resources are available for helping you get started using Keyboard Maestro.

Keyboard Maestro Web Site

The *Keyboard Maestro Web Site* command in the *Help* menu takes you to the Keyboard Maestro web site.

Keyboard Maestro Wiki

The *Keyboard Maestro Wiki* command in the *Help* menu takes you to the Keyboard Maestro wiki.

Keyboard Maestro Forum

The *Keyboard Maestro Forum* command in the *Help* menu takes you to the Keyboard Maestro forum.

Stairways Software Web Site

The *Stairways Software Web Site* command in the *Help* menu takes you to the Stairways Software web site.

Tutorial

The *Tutorial* command in the *Help* menu starts the in-application tutorial. The tutorial will walk you through creating a simple macro. By varying the actions slightly, you can create a variety of macros that are triggered by hot keys and that open various documents.

Practice Gestures

The *Practice Gestures* command in the *Help* menu starts the gesture system and lets you practice gestures, to ensure you can reliably trigger Gesture triggered macros.

ICU Date Format References

The *ICU Date Format References* command in the *Help* menu takes you to the ICU Date Format References web site.

ICU Regular Expression References

The *ICU Regular Expression References* command in the *Help* menu takes you to the ICU Regular Expression References web site.

Regular Expression Unicode Properties

The *Regular Expression Unicode Properties* command in the *Help* menu takes you to the Regular Expression Unicode Properties web site.

MacSparky Field Guide

The *MacSparky Field Guide* command in the *Help* menu takes you to the MacSparky Field Guide [https://learn.macsparky.com/p/km], a high quality, tech-savvy, video tutorial on using Keyboard Maestro from David Sparks.

Elgato Stream Deck Details

The *Elgato Stream Deck Details* command in the *Help* menu takes you to the Stream Deck page, which includes details about installing the plugin and working with a Stream Deck from Keyboard Maestro.

Third Party Licenses

The *Third Party Licenses* command in the *Help* menu displays the licenses folder containing the third party licenses for code used in Keyboard Maestro.

Open Logs Folder

The *Open Logs Folder* command in the *Help* menu displays the Keyboard Maestro Logs folder.

Open Preferences Folder

The *Open Preferences Folder* command in the *Help* menu displays the Keyboard Maestro Preferences folder.

Report Bugs or Feature Requests

The *Report Bugs or Feature Requests* command in the *Help* menu allows you to report any bugs you find or and features you would like implemented (hat tip to the wonderful image editor, Acorn [https://www.stairways.com/action/linkthru?acorn]).

Service and Support

The *Service and Support* command in the *Help* menu displays the service and support details, including how to contact us and where you can get more assistance.

Status Menu

The *Status Menu* menu appears on the right side of the menu bar showing (by default) the Keyboard Maestro icon (although you can control the icon in the General preference pane). It is generally visible any time the Keyboard Maestro Engine is running (unless you turn it off in the General preference pane). You use it to control the Keyboard Maestro Engine, to execute your Status Menu triggered macros, and to switch to applications.

Launch Keyboard Maestro Editor

The *Launch Keyboard Maestro Editor* command in the *Status Menu* menu launches Keyboard Maestro so you can edit your macros. You can also edit your macros directly by holding down the option key while choosing them in the *Status Menu* menu or in a palette.

Paste

The *Paste* menu in the *Status Menu* menu includes the recent text items in the clipboard history so you can easily paste in past items using this menu.

Hold the Shift key down to paste as plain text, and the Option key down to just set the clipboard without pasting.

Activate Clipboard History Switcher

The *Activate Clipboard History Switcher* command in the *Status Menu* ➤ *Paste* menu lets you activate the Clipboard History Switcher.

Your Macros

Your macros that include the Status Menu Trigger and that are currently active will be listed in this menu. You can select them to trigger them.

This is great for less frequently used macros that you will not remember a hot key or other trigger, as well as for macros that *feel like* they should be menu commands, such as actions that *perform* a complicated task.

Show Applications Palette

Toggles the display of the Applications Palette.

Recent

The *Recent* menu in the *Status Menu* menu contains a list of all recently quit applications.

Running

The *Running* menu in the *Status Menu* menu contains a list of all running applications.

Applications

The *Applications* menu in the *Status Menu* menu contains a list of all applications.

Utilities

The *Utilities* menu in the *Status Menu* menu contains a list of all utility applications.

Cancel

The *Cancel* command in the *Status Menu* menu contains a list of all currently running macros and allows you to cancel them.

You can also cancel all currently running macros by holding all the modifiers (Command, Control, Option, Shift) down and clicking on the *Status Menu* menu.

Start Debugging

The *Start Debugging* command in the *Status Menu* menu opens the macrodebugger window and starts debugging.

See also the Macro Debugger section.

Quit Keyboard Maestro Engine

The *Quit Keyboard Maestro Engine* command in the *Status Menu* menu quits the Keyboard Maestro Engine.

Tips

Remembering Macro Hot Keys

Hot Key triggers are only useful if you can remember which key does what.

Consider using mnemonic keys. For example, in your email client, you might define a set of Macros to Insert Text action, so use Control-A for your Address, Control-S for your Signature, Control-N for your Name, and so on.

Be consistent in your choice of Hot Keys. For example, use function keys to launch applications, Control-Function Keys to open documents, Control-Letter to insert text, and so on.

Consider using a single hot key for a set of related functionality. Keyboard Maestro will display a conflict palette and let you select from the choices.

Keyboard Maestro also interoperates with KeyCue – if you use both applications and hold the control key down KeyCue will display all your active Hot Keys.

Use the Conflict Palette

If you have two or more macros with the same Hot Key, pressing that key will display the conflict palette, listing all the conflicting macros. You can then press further keys to filter the list until one remains which is then immediately executed.

Use the Trigger Macro by Name Action

Another way to reduce having to remember Hot Keys is to use the Trigger Macro by Name action to trigger macros based on their name.

Use the Insert Menus

In the editor, there are menus in the Edit menu to inserting actions, functions and text tokens and variables. These can be very useful for both learning about what is available and quickly inserting elements. You can also use text completions to insert functions in numeric fields and tokens and text token fields.

If you hold down the Option key while selecting from these menus you can get help on any of the actions, functions or tokens.

There are also *By Name* versions to quickly insert actions, functions, tokens or variables by name.

Use the Insert Action by Name

Once you are confident using Keyboard Maestro, consider using the Insert Action by Name instead of the Action Selector. When you know the generally available actions, this is usually more efficient.

Pay Attention to the Gear Icon

The gear ⚙ menu in the top right of the menu shows you a variety of different things depending on exactly how it looks.

If the action has a timeout, then a clock face will appear in the middle of the gear. If the timeout has been left as the default, the time will be three o'clock, otherwise the time will be an impossible 4.5 o'clock.

If the action has extra configuration options, the gear menu will have a blue center.

The gear menu itself also shows you whether the item has notes, and whether it will abort or notify you if the action fails.

And finally, the gear menu includes a Help command which will take you to the wiki help on the action.

Use Function Keys for Global Hot Keys

It is quite hard to come up with global Hot Keys that will not conflict with those keys used by any application (a conflict is not really a problem, the Macro Hot Key will simply override the application, but this is not always desirable). It is best to use function keys, especially in conjunction with modifiers, as global Hot Keys since they tend not to be used by most applications.

Use the Number Pad

Remember that the number pad is available (and distinct from the numbers on the main keyboard).

Troubleshooting

See the wiki Troubleshooting page for how to resolve any problems we have anticipated.

Around 80% of all support emails we receive are related to Apple's security permissions, including:

- Translocation
- Accessibility
- Screen Recording
- Secure Input Problem

Unfortunately Apple's permission system has been filled with bugs since Mojave, including things like:

- Checkboxes not showing up in the permission system.
- Checkboxes not being able to be ticked.
- Checkboxes showing as ticked, but not actually granting the permissions.
- The system telling Keyboard Maestro it has permission, when it really does not.

Remember to unlock the System Preferences which is required for changing some (but not all) permissions.

Thankfully, you can pretty much always resolve these by sufficient bashing on the permission system:

- Restart.
- Use `tccutil` to reset the permission system.
- Quit and relaunch Keyboard Maestro or the Keyboard Maestro Engine.
- Ensure you move the Keyboard Maestro.app to the Applications folder **using the Finder**.

Occasionally you can resolve the issues by re-installing the Keyboard Maestro.app:

- Quit Keyboard Maestro Engine
- Quit Keyboard Maestro (editor)
- Delete the Keyboard Maestro.app
- Download [https://download.keyboardmaestro.com] a fresh copy of Keyboard Maestro.
- **Using the Finder**, move the Keyboard Maestro.app to the Applications folder.
- Launch Keyboard Maestro.app

This will not affect any existing macros.

In-built Assistance

If you are finding a macro is not firing when you expect it to, or something weird is happening when you don't expect it to, or choose Help ➤ Interactive Help menu) to bring up an interactive assistance system which might help you narrow down the problem.

Keyboard Maestro can detect some of the System Permission problems described above, when the system is not lying to it.

How do I get more help?

For more information about a specific Keyboard Maestro feature consult the Keyboard Maestro User Manual, post a question to the Keyboard Maestro Forum [https://forum.keyboardmaestro.com/], visit the Keyboard Maestro [https://www.keyboardmaestro.com/] web site or the Keyboard Maestro Wiki or contact us [https://contact.stairways.com/].

Generally the best place to ask about specific macros is the forum [https://forum.keyboardmaestro.com/] since there are many helpful people there and so you will get more varied and quicker answers.

While we are happy to help you learn how to use Keyboard Maestro, we cannot generally help with writing specific macros - use the forum for that.

We always respond to email, however email is no longer a guaranteed medium and spam filters can delete your message to us or our message to you. Messages sent using the feedback form will always get to us, emails sent to us will pretty much always get to us, but if you do not receive a response within one business day check your spam filters to see if they have trapped our reply. If you use the feedback form and want a reply, make sure you enter your email address!

Support

For sales enquiries, customer service, technical support, or to contact project management, email us at support@stairways.com [mailto:support@stairways.com] or use our Web Site Feedback Form [https://www.stairways.com/form/feedback].

Extensive online documentation on Keyboard Maestro can be found in the Keyboard Maestro User Manual and on the wiki at Keyboard Maestro Wiki.

You can join the Keyboard Maestro Forum [https://forum.keyboardmaestro.com/] online community consisting of the developers and Keyboard Maestro users — this is the best place to get help with specific macros.

For ideas, see the Macro Examples section, or the Macro Library section of the Keyboard Maestro Wiki.

You can download Keyboard Maestro from https://download.stairways.com/ [https://download.stairways.com/]. You can download old versions of our applications from the archive site at https://files.stairways.com/ [https://files.stairways.com/].

You can purchase Keyboard Maestro at https://purchase.stairways.com/ [https://purchase.stairways.com/].

You can look up your current or previous license status and serial numbers, and get information about discounted upgrades from https://enquiry.stairways.com/ [https://enquiry.stairways.com/].

For more information about anything to do with Keyboard Maestro visit https://www.keyboardmaestro.com/ [https://www.keyboardmaestro.com/].

Glossary

Collection	An ordered list of things (like "Lines" in a Variable or File) that you can iterate through using the For Each action
Clipboard	The system clipboard is where you store items when you Copy and Paste. When you Copy an item, it is temporarily stored in the Clipboard and when you Paste, the item is copied from the Clipboard into your currently selection
Clipboard History	Normally the system stores only one clipboard. Keyboard Maestro keeps a history of your system clipboard, ensuring you never lose data on the clipboard and allowing you to copy and paste multiple items,
Clipboard Switcher	is a feature of Keyboard Maestro that allows you to copy or paste to/from a set of Named Clipboards
Conflict Palette	is a palette that lets you disambiguate your selection when a macro trigger triggers multiple macros simultaneously
Excluded Applications	is the set of applications that should not appear in the Application Switcher list, allowing you to hide applications you rarely want to switch to. These applications are also ignored when hiding other applications
Global Macro Group	a predefined Macro Group that always exists and is the default location for new Macros
GUID	see UUID
Hot Key	A keystroke that acts as a Macro Trigger to start the execution of Macro Actions in a Macro
KeyCue	software from ergonis that displays command keys and can also display Keyboard Maestro Hot Keys (more info [https://www.stairways.com/action/linkthru?keycue])
Keyboard Maestro Engine	The process that enables your Macros, Application Switcher, Window Switcher, Named Clipboard Switcher and web server to work even after you quit Keyboard Maestro
Mac OS X	see macOS
macOS	The Apple Macintosh Operating System. Formerly known as "OS X".
Macro	Macros are used to automate your workflow, procedure, or process on your Mac. A Macro consists of • A set of steps called Actions. • One or more Triggers to execute the Macro, if it is Active.
Macro Action	an action you wish to perform, such as opening a file, typing some text, controlling Safari, and so on

Macro Group	a set of Macros which can be restricted to only a defined set of applications
Macro Palette	a floating palette containing any active Macros that have a Macro Palette trigger. The palette only appears in applications with at least once active Macro Palette triggered Macro
Macro Trigger	an event, such as a Hot Key, application launch, time of day, that starts the execution of a Macro
Michael Kamprath	the original developer of Application Switcher and Keyboard Maestro
Named Clipboard	Keyboard Maestro provides a set of named clipboards where you can permanently store information (text, logos, graphics, etc)
Notification Center	Mac OS X Mountain Lion introduced a system wide notification center which shows you alerts in the upper-right corner of your screen, without interrupting what you're doing
OS X	Apple's Macintosh operating system versions 10.0 and up
Palette	a floating window containing macros that you can trigger. There are several different kinds of palettes and you can style them in various ways
Program Switcher	the premier application management utility for Classic Mac OS, written by Michael Kamprath it was in part the inspiration for Keyboard Maestro and forms one of the components of Keyboard Maestro
Quick Macro	a macro recorded on the fly in another application. see the Recording section
Record Quick Macro	the action that when triggered records a Quick Macro. see the Recording section
Regular Expression	aka RegEx, a way of matching strings based on patterns. When Keyboard Maestro refers to "matching" it means by regular expression, in particular ICU Regular Expressions [https://www.stairways.com/action/linkthru?icuregex]
Sandboxed	an Apple technology that limits what an application can do and how it can interface with the rest of the system and other applications. This is great for security for applications that work with just their own data, but is impossible with workflow applications like Keyboard Maestro
Shortcut	a Shortcut is another name for a Macro (it is also another name for an Alias but that is a different context to the normal Keyboard Maestro Macro context)
Smart Group	a saved search set of Macros
Sorting Characters	You can control the sorting order of macros by prefixing two characters and a closing parenthesis to the name, eg "01)My Macro". The prefix will be removed before displaying in the macro palette or status menu, but will be used to control the order of the macros shown.
Status Menu	the icon menus at the right of your menu bar
Text Tokens	See Tokens
Tokens	tokens allow you to insert dynamic text, such as the current date or time, into various text fields, see the Tokens section
Tokenized	see Tokens
Toolbar	see Palette
UID	see UUID
UUID	a Universally Unique IDentifier for Macros and Macro Groups and other purposes that remains the same even if you rename or modify a macro or macro group, or import a Macro built on another Mac. Also known as a GUID [https://en.wikipedia.org/wiki/Globally_unique_identifier].
Variables	Like most programming languages, Keyboard Maestro allows you to create Variables [https://en.wikipedia.org/wiki/Variable_(computer_science)] to store data for use later on in the same Macro, or in other Macros. Variables can be set from many actions. You can set variables to specific text, to the result of a calculations, from user input, from the Keychain, by searching other variables, from the clipboard or Named Clipboards, as the result of scripts, and from many other sources. For more details, see Variables.
Z-order	refers to the order of windows from frontmost to furthest back

Administrative Details

Requirements

Keyboard Maestro 10 requires macOS 10.13 or later.

Distribution

You may distribute this application in any way you wish as long as you only distribute the unmodified Keyboard Maestro package, as downloaded from www.stairways.com [http://www.stairways.com]. You may not break Keyboard Maestro up into its component files and distribute parts of it separately.

History

Following on the success of Application Switcher for Classic Mac OS, Michael Kamprath wrote Keyboard Maestro for Mac OS X and released it in early 2002. Incorporating an impressively powerful hot key macro facility, as well as Application and Clipboard Switching facilities, it rapidly became an indispensable tool for many Mac OS X users, including us here at Stairways Software.

Development continued on version 1 through the end of 2002, and then work began on version 2. The first beta of 2.0 was released in early 2003 and development continued until the 2.0b6 beta released in May 2003. After that, life and work got in the way. Keyboard Maestro languished for over a year as Michael found that he did not have the time or energy to continue development.

Around May 2004, we contacted Michael as a concerned user to query the long delay in the eagerly awaited 2.0 release. When we learned that Michael was considering abandoning the application we offered to purchase it from him to ensure that we would not lose this valuable tool, as well as to continue the fine tradition that he had started.

On June 30, 2004 the deal was struck and Stairways Software acquired all the rights to Keyboard Maestro. Our aim was to resolve the outstanding issues with Keyboard Maestro and release 2.0 as soon as possible, which we did in September 2004. Keyboard Maestro 2 introduced many new Macro Triggers (such as Application, Time of Day, and so on), Macro Groups to allow easy control over when macros are active, and many new actions.

Development of Keyboard Maestro competed for resources with development of Interarchy until the latter was sold to lead developer Matthew Drayton in early 2007. After a short break, development on Keyboard Maestro 3 started in earnest and resulted in many new features, including improved and streamlined user interface, recording, new triggers, built-in web server, new actions, and numerous minor enhancements. Keyboard Maestro 3 was released in April 2008 followed by a succession of releases over the rest of 2008.

Development of Keyboard Maestro 4 began in late 2008 and was released in late 2009. Version 4 was a complete rewrite of the user interface, bringing with it a modern look and feel reminiscent of various modern Apple applications. Further minor releases were made through 2010, followed by the initial release of Keyboard Maestro's baby brother Switcher Maestro and the Mac App Store version in January 2011.

By that point, development of Keyboard Maestro 5 was well under way and was released in July 2011. Keyboard Maestro 5 built on the solid user interface of version 4, but added depth and breadth of power with almost no addition of complexity. Keyboard Maestro added such powerful features as control flow, conditions, variables, and calculations as well as many new actions, and enhancements to the application and clipboard history switchers. Further minor releases were made over the next year, adding things like a For Each action, File actions and Image actions.

Development of Keyboard Maestro 6 began in mid 2012 and was released in May 2013. Version 6 kept the user interface largely unchanged, while adding significant new features, including macro syncing, macro debugging, plug in actions, full support for styled text, support for controlling Safari and Google Chrome web browsers, trigger by name facility. Keyboard Maestro 6 also sported a stylish new icon from Iconaholic [https://www.stairways.com/action/linkthru?iconaholic] as well as customizable status menu icons and full Retina graphic support. It was followed by a succession of releases throughout 2013 and 2014.

Development of Keyboard Maestro 7 began in late 2014 was was released in July 2015. Version 7 concentrated on streamlining the editor, adding things like Smart Groups, auto-completion, Insert Action by name and menu, disclosure folding for sub-action lists, renaming, coloring, and adding notes to actions and more. Version 7 also added themed palettes, and the usual plethora of new actions and triggers. Macro Groups could now be targeted at specific windows, and macros could be triggered by window, folder or clipboard changes. It was followed by a succession of releases throughout 2016.

Development of Keyboard Maestro 8 began in late 2016. Version 8 concentrated on further refinements to the editor, including AppleScript support, in-built assistance, variable text sizes, machine learning, dragging, Touch Bar support, expansion of various action menus and more. Also new is enhanced MIDI support, Local/Instance variables, Dictionaries, Gesture, Cron and Remote triggers, and a bunch of new actions, conditions, collections and tokens.

Development of Keyboard Maestro 9 began in mid 2018. Version 9 concentrated on adding Dark Mode support across all of Keyboard Maestro, multiple editor windows to the editor, OCR support, wide ranging JSON support, and extending support for regular expressions. Version 9 also brings support for Stream Deck, and has a hardened runtime and is notorized for additional security. There is also a plethora of new actions and lots of other refinements.

Development of Keyboard Maestro 10 began in early 2021. Version 10 added further refinements to the editor, including saved Favorite actions and lots of extensions to contextual menus. The Engine also received some new facilities like displaying macro groups in the menu bar and more control over engine windows. There were a variety of new triggers and actions, including the "long press" option for Hot Key triggers and Paste by Name, Prompt for Screen Location, displaying progress, Try/Catch/Throw actions, and more.

Going forward, we plan to develop Keyboard Maestro aggressively, bringing it to new levels of both power and ease of use in the long tradition of both macOS and Stairways Software.

Credits

Thanks to Michael Kamprath for all his work producing Keyboard Maestro.

Thanks to Alan Gentle for many example Macro ideas.

Thanks to Philippe Martin for some great beta testing.

Thanks to Dan Benjamin for doing the voice overs on the tutorial videos.

Thanks to Noah Kadner [https://www.callboxlive.com/] the voice overs on the intro video.

Thanks to Rakesh Kumar for the set of Switcher Macros.

Thanks to Sam Stephenson and the Prototype Core Team for the Prototype JavaScript Framework.

Thanks to Jono Hunt for the brilliant Iconaholic [https://www.stairways.com/action/linkthru?iconaholic] icon and other help.

Thanks also:

- to Jerry Krinock [https://www.sheepsystems.com/] for NS(Attributed)String+Geometrics.
- to John Gruber [https://daringfireball.net/], Aristotle Pagaltzis [http://plasmasturm.org/] for Title Case.
- to Kelan Champagne [http://kelan.io/] for the YRKSpinningProgressIndicator [http://kelan.io/2009/yrkspinningprogressindicators/].
- to Matt Gemmell [https://mattgemmell.com/] – Magic Aubergine for MGAnimatedView and MGTemplateEngine.
- to Michael Ash [https://www.mikeash.com/] for MAKVONotificationCenter.
- to Matthew Ball for MBCoverFlow [https://github.com/mattball/MBCoverFlowView].
- to Rainer Brockerhoff [https://www.brockerhoff.net/] for RBSplitView [https://brockerhoff.net/src/rbs.html].
- to Keith Blount [https://www.literatureandlatte.com/freestuff.php] for KBWebArchiver.
- to Brian D K Jones for VDKQueue.
- to James Berry for Tag.
- to Flying Meat [https://flyingmeat.com] for FMDB.
- to Random Ideas for NSTimer+Blocks.
- for OpenCV [https://opencv.org/].

- for JSON_Checker [https://github.com/douglascrockford/JSON-c].
- for Leptonica [http://leptonica.org/]
- for SocketRocket.
- for Tesseract.

Thanks to Ken, Corentin, Stephen, Brad and others for their great assistance with beta testing.

Thanks to Jim Underwood for outstanding assistance on the forum and wiki. Jim passed away in 2021 and he will be sorely missed.

Thanks to Christopher Stone and ComplexPoint and many others for their great help on the forum.

Thanks to Andy for great help editing this documentation.

Thanks also to the many others who have provided input and support over the past decade.

Warranty

This application should do what we have described in this document. If it does not, you can simply stop using it. If you purchase it, and within 30 days find that it does not do what we have described here, then you can request a refund and your money will be refunded and we will cancel your license.

Licenses

Keyboard Maestro is licensed on a per user basis and individual users may use it on up to five Macs. You must purchase a license for each user using Keyboard Maestro. A license is for the current major version and generally includes minor updates to it. There will likely be an upgrade fee for any future major versions.

Fine Print