

The 8th™ Programming Language

Word list version 24.06

Copyright © AHT Associates LLC, All Rights Reserved

8th™ is a trademark of AHT Associates LLC

Words by namespace

The following is an exhaustive list of the built-in words in *8th*, arranged by namespace, including descriptions of the words and their SEDs. This information is the same as that provided by the console help.

2fa

Namespace: **2fa**

Description: Two factor authentication

word	sed/description
gen-secret	-- b (as of 21.02)
	<i>needs 2fa/totp</i>
	Generate a secret key for the TOTP algorithm to use. Returns a 16 byte random buffer.

word	sed/description
gen-url	m -- url (as of 21.02)
	<p><i>needs 2fa/totp</i></p> <p>Takes a <i>map</i> with the parameters to be converted to a QR code or base64 encoded for Authy or Google Authenticator, etc.</p> <p>Valid keys are:</p>
validate-code	key code ndigits -- T (as of 21.02)
	<p><i>needs 2fa/totp</i></p> <p>Takes the TOTP secret and the code entered by the user, and validates the code</p>

Application

Namespace: **app**

Description: Application

word	sed/description
8thdir	-- s
	<p>Returns the name of the (OS specific) directory where 8th stores data; this is almost the same as app:datadir , but is always the ".8th" directory.</p>
asset	s -- b
	<p>Read an "asset", which is a resource bundled with the application. Returns a buffer containing the resource content. The parameter s refers to a "relative path". That path is relative to where the script or application appdata is located. So if the script is in src/app.8th then the asset "image/a.png" is in "src/image/a.png". For a compiled application, the appdata is created with bin/build .</p>
atrun	opt --
	<p><i>needs win/atrun</i></p> <p>given a <i>map</i> opt, arrange to launch the indicated application at a specific time. the keys in the <i>map</i> are: "exe" - a <i>string</i> which is the name of the executable to run (required); "args" - an <i>array of strings</i> which has the command-line arguments to the application; "when" - a <i>date</i> which indicates the time at which the application should be run (required); "dir" - the directory where the exe should run (not effective on linux/rpi).</p>

word	sed/description
atrun	opt --
	<p><i>needs mac/atrun</i></p> <p>Given a <i>map</i> opt, arrange to launch the indicated application at a specific time. The keys in the <i>map</i> are: "exe" - a <i>string</i> which is the name of the executable to run (required); "args" - an <i>array</i> of <i>strings</i> which has the command-line arguments to the application; "when" - a <i>date</i> which indicates the time at which the application should be run (required); "dir" - the directory where the exe should run (not effective on Linux/RPI).</p>
atrun	m --
	<p><i>needs lin/atrun</i></p> <p>given a <i>map</i>, arrange to launch the indicated application at a specific time. The keys in the <i>map</i> are:</p> <ul style="list-style-type: none"> • exe: <i>string</i> which is the name of the executable to run (required) • args: <i>array of strings</i> which has the command-line arguments to the exe • when: <i>date</i> which indicates the time at which the application should be run (required) • dir: <i>string</i> directory where the exe should run (not effective on linux/rpi). <p>Requires "atd" be installed and running.</p>
basedir	-- s
	Returns a string which is the name of the (OS specific) application-accessible directory.
basename	-- s (as of 23.04)
	Returns the name of the executable 8th is named as. "8th" , unless compiled as a standalone program.
config-file-name	s -- s (as of 21.08)
	<p><i>needs utils/config-file</i></p> <p>Given a base-name, return the OS-specific path to a "private" configuration file It is the result of app:privdir plus the base-name you give.</p>
current	-- (as of 18.01)
	Launches a system-defined browser to the canonical 8th "current" page, with your user id and version of 8th. Will give you the option to refresh your build if you like.
datadir	-- s
	Returns the OS specific name of the directory where this application stores data.

word	sed/description
display-moved	n -- (as of 23.08)
	DEFERRED Invoked when the primary monitor has moved or changes position with another monitor. n is the index of the affected display.
exename	-- s
	Returns a string which is the running executable's full path name.
localechanged	-- (as of 21.08)
	DEFERRED Invoked when the user changes the system locale. After this message is received, you can get the new current locale using os:lang and os:region , or all the user's chosen locales with os:locales .
lowmem	-- (as of 20.01)
	DEFERRED Invoked if the OS is running low on memory. Free up resources if you can.
main	--
	If you define a word with this name, your application will start there. Otherwise, the default version will run, which just invokes G:cold and the REPL.
name	-- v (as of 21.06)
	A var which contains the name of the current application. The default value is "8th", so you probably want to change it, e.g. "myappname" app:name ! .
onback	-- (as of 24.05)
	DEFERRED Invoked when the "back button" is pressed (Android only). The default d: behavior invokes bye .
oncrash	s -- (as of 18.06)
	If the application crashed, this word is invoked with a message.
opts!	s x -- (as of 19.07)

word	sed/description
	<p>Stores the value x under the key s for retrieval by app:opts@ .</p> <p>See also: app:opts@</p>
opts@	s -- x (as of 19.07)
	<p>Returns the value v stored in the key s by app:opts! . Values are application-global, and are typically used to pass information to libraries upon loading. Ex: G:help uses the "help.width" key.</p> <p>See also: app:opts!</p>
orientation	n -- (as of 16.13)
	<p>DEFERRED</p> <p>Invoked (on mobile devices) when the device orientation has changed. n is 1 for 'upright', 2 for 'upside down', 3 for 'rotated clockwise' and 4 for 'rotated anti-clockwise'.</p>
orientation!	n -- (as of 20.04)
	On mobile devices (Android only for the moment) sets the permitted orientation: 1 for 'upright', 2 for 'upside down', 3 for 'rotated clockwise' and 4 for 'rotated anti-clockwise'.
pid	-- n (as of 17.06)
	Returns the application's OS-specific process-id.
post-main	-- (as of 20.01)
	<p>DEFERRED</p> <p>For non-mobile applications only. Invoked just after app:main exits.</p>
pre-main	-- (as of 20.01)
	<p>DEFERRED</p> <p>For non-mobile applications only. Invoked prior to app:main .</p>
privdir	-- s (as of 21.08)
	Returns the os-specific "private dir" (e.g. "/etc/" on POSIX systems)
raise	n -- s -- (as of 19.09)

word	sed/description
	<p>Sends ("raises") the given signal to the app. If it's a number, that signal number is raised; if it's a string, then the signal of that name is raised. Only signals which 8th traps by default can be named.</p> <p>Ex: "USR1" app:raise works, but "KILL" app:raise does not.</p> <p>See also: app:signal app:trap</p>
read-config	s -- m (as of 21.08)
	<p><i>needs utils/config-file</i></p> <p>Given the config-file base-name, return a buffer containing the contents of the file (or null, if there's nothing there)</p>
read-config-map	s -- m (as of 21.08)
	<p><i>needs utils/config-file</i></p> <p>Given the config-file base-name, return a map with values in the JSON formatted map in that file. Returns an empty map if the file does not contain one.</p>
read-config-var	s name -- x (as of 21.08)
	<p><i>needs utils/config-file</i></p> <p>Given the config-file base-name, and the name of a variable (a key in the map therein), return the value of the key.</p>
request-perm	s -- T (as of 20.05)
	<p>ANDROID: request the OS-specific permission (e.g. "android.permission.RECORD_AUDIO"). Returns true if the permission was granted.</p>
restart	-- (as of 17.06)
	<p>Restarts the application from the beginning, launching a new process.</p>
resumed	--
	<p>DEFERRED</p> <p>Mobile only. Invoked when the application is about to be resumed. TOS is true if app has been resumed, false if preparing to resume. Take care of restoring application state here, saved during app:suspended .</p> <p>See also: app:suspended</p>

word	sed/description
signal	s -- (as of 19.09)
	<p>DEFERRED</p> <p>Invoked when the OS signal s has been received. Traps SIGINT, SIGTERM, and SIGABRT on all platforms. On other than Windows, it also traps SIGHUP, SIGUSR1, and SIGUSR2. The signal name is passed in s (e.g. "INT"). For signals trapped using app:trap and which are unknown to 8th, the numeric value of the signal is converted to a string and passed in. The default behavior is to print the name of the signal received and quit. A signal handler is not guaranteed to be invoked in any particular task, so be careful!</p> <p>See also: app:trap app:raise</p>
standalone	-- v (as of 17.10)
	A var which holds the value true if the app is running as a built, standalone, application. It is false otherwise.
standalone!	T -- (as of 24.06)
	Sets app:standalone . If true , w:forget s words which are security risks. The build utility inserts true app:standalone! in built applications.
subdir	s -- s' (as of 17.01)
	Returns the name of the subdirectory of the (OS specific) directory where 8th stores data. s may or may not exist on the system, this word simply returns a composed name.
suspended	T --
	<p>DEFERRED</p> <p>Mobile only. Invoked when the application is about to be suspended. TOS is true if the app has been suspended, false if preparing to suspend. Take care of saving application state here, in preparation for app:resumed .</p> <p>See also: app:resumed</p>
sysquit	-- T
	<p>DEFERRED</p> <p>Invoked if the system requests that the application shut down. The default simply returns true , which means "quit". If you do not want to allow the system to close the application, you should return false instead.</p> <p><i>Note: denying the OS doesn't mean you will succeed...</i></p>
terminated	-- (as of 20.01)

word	sed/description
	DEFERRED Invoked if the OS is terminating the application for some reason.
ticks	-- n (as of 24.06)
	Gets the number of 'ticks' the app has been running, since the app was loaded. See also: t:ticks
timeout	n -- (as of 22.01)
	<i>needs utils/timeout</i> Creates a background task which waits for n seconds before invoking app:timed-out (which does nothing, by default) and then invoking 1 die .
trap	n T -- (as of 19.09)
	Tell 8th to trap the signal whose number is n (this will be OS-specific). If T is true, the signal should be trapped; otherwise, it should be ignored. The deferred word app:signal will be invoked if the signal is intercepted. The acceptable values of n are OS-specific. See also: app:signal app:raise

Array

Namespace: **a**

Description: Arrays are sequentially ordered containers accessed by numeric index

word	sed/description
!	a ix x -- a
	Put the item x in the array at index ix . If the index is larger than the largest current index, the array will be resized and any empty spots created will be filled with null . If ix is negative, then the element at the index from the end of the array will be modified. For example, an index of "-1" will modify the last element. See also: a:@
+	a1 a2 -- a1

word	sed/description
	Append contents of array a2 to a1 , modifying a1 .
-	a ix -- a
	Remove the item at index ix from the array a , moving other items over to fill the gap: a is modified. If you just want to remove an item from the array without changing the position of other elements, simply store null in the position of the item you want to remove. If ix is less than 0 the item that far from the end of the array is removed.
/	a ix -- a1 a2 a w -- a1 a2 (as of 21.01)
	Splits an existing array into two parts at the index. If the index is less than 0, splits at that index from the end of the array. If given a word , then that word determines which array (a1 or a2) to split the item to. In that case the SED of the word must be ix x -- T where it returns true for a1 , or false for a2 .
2each	a1 a2 w -- a1 a2 (as of 18.08)
	<p>Iterates over two arrays, invoking w for each corresponding element in them. w 's SED is: n x1 x2 -- ; that is, it is passed both the index of the items and the items themselves. The arrays are not available on the stack while it is running, to avoid consistency problems. w must consume its inputs. Will stop iterating on break or once the shorter of the two inputs is exhausted. Obeys G:step .</p> <p><i>NOTE: You must not modify the array being iterated, while it is being iterated! Modifying the array while it is being iterated may throw an exception.</i></p> <p>See also: a:each a:each! a:x-each</p>
2map	a1 a2 w -- a' (as of 18.08)
	<p>Similar to a:map , but takes two arrays and invokes w on the corresponding elements of them. If the arrays are not of equal size, will stop after processing the shorter one; thus, the result will be of the same length as the shorter of the two inputs.</p> <p>See also: a:map a:2map=</p>
2map+	a1 a2 w n1 n2 -- a' (as of 18.08)

word	sed/description
	<p>Similar to a:2map , but takes additional parameters:</p> <ul style="list-style-type: none"> • n1 : the number of items of a to pass in each time • n2 : number of items to advance before creating the next slice <p>The SED of w is a3 a4 -- x , where a3 is an n1 sized slice of a1 , and a4 of a2 .</p> <p>By default, the initial slice index increments by 1 each iteration; by using step (inside w) you can make that any value you wish. The result is a new array with the values returned by w on each iteration.</p> <p>Ex: [1,2,3,4] [10,20,30,40] (2 a:close) 2 1 a:2map+ yields: [[[1,2], [10,20]], [[2,3], [20,30]], [[3,4], [30,40]]]</p> <p>See also: a:2map</p>
2map=	a1 a2 w -- a1 (as of 18.08)
	<p>Same as a:2map , but modifies the original array.</p> <p>See also: a:map a:2map</p>
<>	a ix1 ix2 -- a (as of 22.07)
	Swaps the array items at those indices. Ex: [10,20,30] 1 2 a:<> returns [10,30,20] .
=	a1 a2 w -- a1 a2 T (as of 18.04)
	Compares the two arrays using the word w to compare each item. The SED for w is: x1 x2 -- T , where T is true if the items are equal, otherwise false. Arrays must have equal length to be considered equal, and each compared element must have the same type.
@	a ix -- a x a a1 -- a a2
	<p>Returns the item in the array at index ix . If ix is negative, get the item from the end of the array, e.g. "-1" corresponds to the last element. If ix is out of bounds, null is returned (though null might also be a valid element! Use a:exists? to distinguish). If the index is an array of numbers, returns another array with values corresponding to the indices in a1 .</p> <p>See also: a:! a:exists?</p>
@?	a ix x -- a x'

word	sed/description
	<p>Same as a:@ , but if the item at that index doesn't exist, supplies the default value x . If x is an array, then if ix is <i>also</i> an array, the value returned will be from the corresponding element of the x array (last element used for all corresponding elements beyond the original's bounds).</p> <p>See also: a:! a:exists? m:@?</p>
[]!	a n x -- a (as of 20.08)
	<p>Same as a:! , but if the index n doesn't exist or is not an array, a new array is created, the existing value if any is pushed to it, and the new value is pushed into it. Basically, accumulates values to the index.</p> <p>Ex: [10,20,30] 1 21 a:[]! results in [10,[20,21],30]</p>
_@	a n -- x (as of 19.07)
	<p>Same as a:@ , but removes the array from the stack.</p> <p>See also: a:@</p>
all	a w -- a T
	<p><i>needs python/any</i></p> <p>Iterates the <i>array</i>, invoking w on each one. If all of the items return true from w then returns true; otherwise returns false. Short-circuits on first 'false' result.</p>
any	a w -- a T
	<p><i>needs python/any</i></p> <p>Iterates the <i>array</i>, invoking w on each one. If any of the items returns true from w then returns true; otherwise returns false. Short-circuits on first 'true' result.</p>
bsearch	a x w -- a ix (as of 16.01)
	<p>Search for the item x in a <i>sorted</i> array, using the comparison word w (which was used for the sort!), whose SED is: x1 x2 -- n , where n is a negative number if the first item is less than the second, 0 if they're equal, and positive if the first is greater than the second.</p> <p>Returns the index of the matching item, or null if there was no match.</p> <p>Note: The array must have been sorted in ascending order using the same comparator w in order for this to work correctly!</p>
centroid	a -- a'

word	sed/description
	<p><i>needs array/centroid</i></p> <p>Given an array of points, where each point is an array of x,y coordinates, return an array which consists of the centroids of those points. May be any dimension as long as each point is the same dimension.</p>
clear	a -- a
	Remove all elements from the array. Similar to repeatedly invoking a:pop drop .
close	x1 x2 x3 ... xN n -- a
	<p>Create an array from n items on the stack. This is the inverse of a:open . If n is 0 or negative, will only consume TOS (e.g. n).</p> <p>See also: a:open</p>
cmp	a1 a2 w -- a1 a2 n (as of 23.08)
	<p><i>needs array/cmp</i></p> <p>Compare two arrays. Similar to 'a:=' , but given a comparator word (like s:cmp or n:cmp) The first non-equal element terminates the comparison, and 'n' is the result of the comparator on that element. Result 'n' is 0 if arrays compare 'equal', '-1' if a1 is 'less'</p>
diff	a1 a2 w -- a1 (as of 22.02)
	Removes from a1 all elements which are in a2 . The w word should return true if the items are equal.
dot	a1 a2 w1 w2 -- x
	<p>Generalized "dot product" of the two arrays a1 and a2 . If they are not the same length, returns null . Otherwise, invokes the word w1 on each corresponding item in each array, and then invokes the word w2 on each pair of results, accumulating them into a result x . Ex: [1,2] [3,4] ' n:* ' n:+ a:dot results in the number 11.</p>
each	a w -- a

word	sed/description
	<p>Iterate over the array, invoking w for each element in it. The SED of w is ix x -- , where ix is the index of the item x in the array.</p> <p>While running, the array is <i>not</i> available on the stack, to avoid consistency problems. w must consume <i>both</i> items.</p> <p><i>NOTE:</i> You must <i>not</i> modify the array being iterated, while it is being iterated! Modifying the array while it is being iterated may throw an exception.</p> <p>See also: a:each!</p>
each!	a w -- a (as of 19.04)
	<p>Same as a:each , but only passes the item, omitting the index, to the word. The SED of w is x -- .</p> <p><i>NOTE:</i> You <i>must not</i> modify the array being iterated, while it is being iterated! Modifying the array while it is being iterated may throw an exception.</p> <p>See also: a:each</p>
each-par	a w n -- a (as of 23.09)
	<p><i>needs array/parallel</i></p> <p>Splits the array into n sub-arrays and invokes a:each! on each of them in a separate task, then recombines them.</p>
each-slice	arr wrd sz -- arr
	<p><i>needs array/each-slice</i></p> <p>Split <i>array</i> arr into slices of size sz elements, and feed each slice to the <i>word</i> wrd. That callback gets the stack slicenumber slice on its stack.</p>
exists?	a ix -- a T
	<p>Returns true if there is a value defined for the given index in the array. This is necessary because requesting an arbitrary index from an array will always succeed, returning null if the index does not exist as well as if the value stored is actually null . If the index is negative, it is taken as that many from the end.</p>
filter	a w -- a'

word	sed/description
	<p>Creates an array, whose elements are those from the initial array for which the word w returns true . The SED of w is x -- T , where x is each element of a in turn. It must consume each item it is given.</p> <p>While this word is running, the original array is <i>not</i> on the stack so it is possible to access items which were under it.</p> <p>See also: a:reduce a:map</p>
filter-par	a w n -- a' (as of 23.09)
	<p><i>needs array/parallel</i></p> <p>Splits the array into n sub-arrays and invokes a:filter on each of them in a separate task, then recombines them.</p>
generate	w n1 n2 -- a w -- a (as of 18.08)
	<p>Loops over the numeric range [n1, n2] , invoking w on each number. The result in TOS is pushed onto the resultant array.</p> <p>If given only a word, then that word is invoked repeatedly with a SED of n -- x where n is the current index (starting at 0) and x is what TOS is after w is invoked. The repetition stops after break is invoked.</p>
group	a w -- m (as of 18.08)
	<p>Group elements in an array, creating a map whose values are arrays containing items with the same grouping. The word used to group the items takes an array element and returns an item which is the group to which that element belongs, SED x -- s .</p>
indexof	a x w -- a ix (as of 16.01)
	<p>Search for the item x in the array, using the comparison word w , whose SED is: x1 x2 -- T . It takes two items of the given type and returns true if they match, or false otherwise.</p> <p>Returns the index of the matching item, or null if there was no match.</p>
insert	a1 a2 ix -- a3
	<p>Create a new array by inserting the contents of array a2 into the array a1 at offset ix . If the offset is negative, it means "from the end of the array". If ix is greater than the current number of items in a1 , the missing items will be null .</p> <p>See also: a:+</p>
intersect	a1 a2 w -- a3 (as of 22.02)

word	sed/description
	Produces the "intersection" of two arrays over the comparator word. The resultant array contains only those values common to the input arrays (without duplication).
join	a sep -- s
	<p>Inverse of s:/ . Joins an array of strings, inserting the string sep between each pair of items; return new string s .</p> <p>See also: s:/</p>
len	a -- a n
	Returns the length of the array, e.g. the highest index occupied plus 1 (indices start at 0). The actual number of items contained may be less than the array's length, because one can insert an item at any index (subject to system memory constraints).
len'	a -- n (as of 24.03)
	Same as a:len but does not leave the array on the stack.
len2	a1 a2 -- a1 a2 n1 n2 (as of 24.03)
	Given two arrays, returns their respective lengths. Much faster and more efficient than the hard way.
map	a w -- a'
	<p>Creates an array, whose elements are formed by executing the word w for each element of the original array. The SED of w is x -- x' , where x' may be the same as x . The mapping must consume the element given, producing a single result on TOS, which becomes the corresponding element in the new array.</p> <p>While this word is running, the original array is <i>not</i> on the stack so it is possible to access items which were under it.</p> <p>See also: a:filter a:reduce</p>
map+	a w n1 n2 -- a' (as of 18.08)

word	sed/description
	<p>Similar to a:map , but takes additional parameters:</p> <ul style="list-style-type: none"> • n1 : the number of items of a to pass in each time • n2 is the number of items to advance before creating the next slice <p>The SED of w is a -- x , where a is an n1 sized array slice of the original array. By default, the initial slice index increments by 1 each iteration; by using step (inside w) you can make that any value you wish.</p> <p>See also: a:map</p>
map-par	a w n -- a' (as of 23.09)
	<p><i>needs array/parallel</i></p> <p>Splits the array into n sub-arrays and invokes a:map on each of them in a separate task, then recombines them.</p>
map=	a w -- a (as of 18.05)
	<p>Same as a:map , but modifies the original array.</p> <p>See also: a:map</p>
maxlen	a -- n (as of 23.01)
	<p><i>needs array/maxlen</i></p> <p>Returns the maximum (character) length of an array of strings</p>
mean	a -- a n
	<p><i>needs math/mean</i></p> <p>Calculate the mean (average) value of an <i>array of numbers</i>. Should be faster than a:mean&variance if all you need is the arithmetic mean</p>
mean&variance	a -- a a'
	<p><i>needs math/mean</i></p> <p>Uses the Knuth variance algorithm to calculate mean and variance in one pass. Returns an array containing [sample variance, mean, count, population variance, stddev]</p>
merge	a1 a2 w -- a3 (as of 21.06)
	<p>Combines the two arrays as in a 'merge sort', using the given word as a comparator. If the arrays are not already sorted, garbage will result.</p>

word	sed/description
new	-- a (as of 17.05)
	Create a new, empty array.
op!	a ix w -- a (as of 17.01)
	Invokes <i>w</i> on the array contents at the specified index, replacing the current contents. Any operands to w should appear in their normal stack order under w .
open	a -- x1 x2 x3 ... xN
	<p>"Open up" the contents of the array a, spilling it onto the stack. Faster and more clear than manually unpacking the array. <i>Be careful</i> not to overflow the stack! Useful for example after using G:unpack so you can access the data easily. Pair with a:close (you need to add back the length in TOS for that to work).</p> <p>See also: a:close</p>
pigeon	a1 a2 x w -- x' (as of 21.09)
	<p>Performs a 'pigeonhole operation', meaning that x is compared using w to successive (ascending, ordered) values in a2. The index of the item which is "greater" than x is used to look-up a value in a1. Ex:</p> <pre>["red", "blue", "green"] [10,20] 15 ' n:cmp a:pigeon</pre> <p>returns "blue", since 15 is between 10 and 20.</p>
pivot	a -- a' (as of 22.08)
	<p>"Pivots" the array. If it consists of arrays (all the same length) like [[x1,x2,...], [y1,y2,...]] then the pivot creates a new array [[x1,y1,...], [x2,y2,...]]. The length of the largest array element is taken to be the length of <i>all</i> elements (and null will be in the pivot for items not in smaller arrays). If any of the elements of a is not itself an array, returns null.</p>
pop	a -- a x
	<p>Pop the item x from the array; that is, from the highest index in the array. If the array was empty, returns null.</p> <p><i>Note:</i> Since it is possible that null was actually stored in the array, use a:exists? to remove ambiguity.</p> <p>See also: a:push a:@ a:!</p>
push	a x -- a x a -- a

word	sed/description
	<p>Push the item x onto the array. The array will expand as needed. After the push, x will occupy the highest index in the array.</p> <p>The order may be either a x or x a . If x is itself an array, then you <i>must</i> use the order a x in order to achieve the desired result.</p> <p>See also: a:pop a:@ a:!</p>
push'	a x -- (as of 24.03)
	Same effect as a:push drop but more efficient.
qsort	a w -- a (as of 18.08)
	<p>Sorts the array using the "quicksort" algorithm and the comparison word w .</p> <p>The SED of w is: x1 x2 -- n , where n is 0 if the items are equal, positive if x1 ≥ x2 , negative otherwise.</p> <p><i>Note:</i> If the comparison function modifies the elements it is given, they will be modified in the original array! It is typically slower than a:sort ; as well as not being a stable sort, but it uses less memory.</p> <p>See also: a:sort</p>
randeach	a w -- a (as of 18.08)
	<p>Same as a:each , but iterates the array in a <i>random</i> order. Because of that, it does <i>not</i> obey step .</p> <p><i>NOTE:</i> You <i>must not</i> modify the array while it is being iterated!</p>
reduce	a w x -- x'
	<p>Iterate over the array, invoking w for each element in a with the accumulated value.</p> <p>The SED of w is x elem -- x' , where elem is an element in a , and x is the accumulated value so far, starting with x .</p> <p>While running, the original array is not on the stack.</p> <p>Ex: [1,2,3] ' n:+ 0 a:reduce sums the array giving 6.</p> <p>See also: a:map a:filter</p>
reduce+	a w x n1 n2 -- x' (as of 18.08)

word	sed/description
	<p>Same as a:reduce , but takes additional parameters:</p> <ul style="list-style-type: none"> • n1 : the number of items of a to pass in each time • n2 : the number of items to advance before creating the next slice <p>As with a:map+ , invoking step (inside w) will make it advance by step items. The SED of w is x a -- x' , where a is an array with n1 items from the original, and x is the accumulated value so-far.</p> <p>Ex: [1,2,3,4,5,6] (a:open n:* n:+) 0 2 1 a:reduce+ gives '70', which is 1*2 + 2*3 + 3*4 + 4*5 + 5*6 .</p> <p>[1,2,3,4,5,6] (a:open n:* n:+) 0 2 2 a:reduce+ gives '50', which is 1*3 + 2*4 + 3*5 + 4*6 .</p> <p>See also: a:reduce</p>
remove	a x -- a n (as of 20.01)
	Removes all instances of the item x from the array. Returns the number of instances found and removed.
rev	a -- a
	Given an array, modifies it so the elements are in reverse order.
rindexof	a x w -- a ix (as of 21.09)
	Same as a:indexof , but searches from the <i>end</i> of the array.
shift	a -- a x
	<p>Removes the item x from the first position in the array (e.g. a[0]) and puts it on TOS. This makes the array one element shorter, and shifts all remaining elements one lower. Like a:pop but from the <i>start</i> of the array.</p> <p>See also: a:slide a:push a:pop</p>
shuffle	a -- a
	Randomly shuffles the entries in the array Uses rand-pcg , so it is not <i>cryptographically</i> random.
slice	a ix n -- a'

word	sed/description
	Returns a slice of the array a beginning at index ix for n items. A negative value of ix means "take from the end of the array". If ix+n is greater than the number of items in a (starting from ix), then the slice will be only as large as the number of items available in a . A negative value of n means "take the rest of the array".
slice+	a ix n step -- a' (as of 18.04)
	<p>Same was a:slice but also takes a "step" amount, which is how many items to advance before creating next slice.</p> <p>Ex: [0,1,2,3,4] 1 -1 2 a:slice+ returns [1,3]</p> <p>See also: a:slice</p>
slide	a x -- a
	<p>Puts x in the first position in the array (e.g. a[0]), moving all remaining items up one index. Like a:push but at the beginning of the array.</p> <p>See also: a:shift a:push a:pop</p>
smear	a n -- a (as of 22.04)
	Smears the last item in the array, duplicating it to fill the array up to n items. Ex: [1,2,3] 5 a:smear results in [1,2,3,3,3] .
sort	a w -- a
	<p>Sorts the array using the "timsort" algorithm and the comparison word w. Exactly the same as a:qsort, but uses a stable sort instead.</p> <p>The SED of w is: x1 x2 -- n, where n is 0 if the items are equal, positive if x1 ≥ x2, negative otherwise.</p> <p><i>Note:</i> If the comparison function modifies the elements it is given, they will be modified in the original array! This sort is generally faster than a:qsort, but uses more memory.</p> <p>See also: a:qsort</p>
split	a n -- a' (as of 23.09)
	Splits the array into one containing n equal sub-arrays. So [1,2,3,4] 2 a:split results in [[1,2],[3,4]] . If the number of items in a doesn't split evenly, the last entry in a' will have all the rest. If n is greater than the number of items in the array (or is less than 0) it is clamped to the number of items in the array.

word	sed/description
squash	a -- a' (as of 22.07)
	Reduces an "array of arrays" to a array whose contents are the contents of the arrays. Ex: <code>[[1],[20]]</code> a:squash returns <code>[1,20]</code> . If an item in the array is not itself an array, it is simply placed as-is.
switch	a ix1 ix2 -- a (as of 22.05)
	Swaps the items at the given array indices.
union	a1 a1 w -- a3 (as of 22.02)
	Creates an array a3 which is the union of two arrays over the comparator word. That means that the result contains all values in both inputs, excluding duplicates.
uniq	a w -- a' (as of 22.02)
	Removes consecutive equal elements of the array. w should return true if the items are equal. Assumes the input array is sorted.
unzip	a -- a' (as of 22.05)
	Inverse of a:zip . Takes an array like <code>[[1,2], [3,4]]</code> and produces <code>[1,3,2,4]</code> . Iterates over all the sub-arrays in the original, and creates a new array containing the corresponding elements of each sub-array, in the order given. See also: a:zip
when	a --
	Given an array containing pairs of words, it invokes the first word in each pair. If that word evaluates to true , then the second word is invoked. If none of the elements evaluates to true , then if there is a last (unpaired) item it is invoked. See also: a:when!
when!	a --
	Same as a:when , but invokes <i>all</i> words whose matching first part evaluates to true . See also: a:when
x	a1 a2 w -- a' (as of 18.08)

word	sed/description
	<p>Same as a:x-each , but returns an array with the results from the processing. So w 's SED is n1 n2 x1 x2 -- x3 , where x3 is put in the resultant array. Modifying the arrays while it being iterated may throw an exception.</p> <p>See also: a:x-each</p>
x-each	a1 a2 w -- a1 a2 (as of 18.08)
	<p>Iterate over two arrays, invoking w for all elements in them. The effect is to invoke w on every pair of items in the two inputs. w 's SED is: n1 n2 x1 x2 -- ; that is, it is passed the indices of both items, and the items themselves. The arrays are not available on the stack while it is running, to avoid consistency problems. w must consume its inputs. Will stop iterating on break or when all items have been iterated.</p> <p><i>NOTE:</i> You <i>must not</i> modify the array being iterated, while it is being iterated! Modifying the array while it is being iterated may throw an exception.</p> <p>See also: a:each a:each! a:2each a:x</p>
xchg	a ix x -- a x' (as of 18.04)
	<p>Stores a new value in the array at index ix , placing the current value on TOS. A negative index means "from the end of the array".</p> <p>See also: G:xchg m:xchg</p>
y	a w -- a'
	<p>Given an array and a word whose SED is x1 x2 -- x3 , produces another array which is the result of applying w to consecutive elements of it. This will result in an array <i>one element shorter</i> than the input. If the input has fewer than two elements, a clone will be returned and w will not be invoked.</p>
zip	a1 a2 -- a3 a4 ...
	<p>Takes two arrays, and "zips" them together (think like a "zipper" on clothing), producing a new array for each corresponding item in the first two. Ex: [1,2,3] [3,4,5] a:zip produce [1,3] [2,4] [3,5] . If the arrays are not of equal size, stops after processing the shorter one. Thus the number of resultant arrays will be the minimum of the lengths of the two inputs.</p> <p><i>Note:</i> be careful not to use this on very long arrays or you may overflow the stack!</p>

Astronomical

Namespace: **astro**

Description: Various astronomical calculations

word	sed/description
dawn	-- v
	<i>needs astro/sunrise</i> var with dawn angle of sun, v. Default is -7.
do-dawn	-- rise set
	<i>needs astro/sunrise</i> Do dawn calculation.
do-dusk	-- rise set
	<i>needs astro/sunrise</i> Do dusk calculation.
do-rise	-- rise set
	<i>needs astro/sunrise</i> Do sunrise/sunset calculation (this is the default).
dusk	-- v
	<i>needs astro/sunrise</i> var with dusk angle of sun, degrees below horizon. Default is -7.
latitude	-- v
	<i>needs astro/sunrise</i> var with latitude coordinate of location (default is Jerusalem: 31.778). North is positive, south is negative.
location!	loc -- lock
	<i>needs astro/sunrise</i> Takes a location <i>map</i> from the geo/location and sets current latitude and longitude.
longitude	-- v

word	sed/description
	<i>needs astro/sunrise</i> var with longitude coordinate of location (default is Jerusalem: 35.235). East is positive, west is negative.
sunrise	d -- rise set
	<i>needs astro/sunrise</i> For the given date d, return rise, set times. Assumes the location has been set already.

Authentication

Namespace: **auth**

Description: Authentication for communication channels

word	sed/description
genkeys	-- priv pub (as of 22.06)
	<i>needs auth/keys</i> Return a pair of keys to be used for session authentication
secret	priv pub -- b (as of 22.06)
	<i>needs auth/keys</i> Return a buffer which is the DH shared-secret of the keys
session-id	-- s (as of 22.06)
	<i>needs auth/sessionkey</i> Return a random 16 character string (12 bytes of random data). This string is intended to identify the session from amongst a multitude of sessions over the channel. Save
session-key	-- b (as of 22.06)
	<i>needs auth/sessionkey</i> Return a random session-key. A key is a cryptographically random buffer which can be used for encrypting the communications channel. The key should not be stored permanently.
validate	priv pub key -- T (as of 22.06)

word	sed/description
	<i>needs auth/keys</i>
	With one party's public key and the other party's private key, validate that the 'key' is the result of 'auth:secret'

AWS

Namespace: **AWS**

Description: Amazon AWS CLI utility

word	sed/description
cb	w -- (as of 23.03)
	<i>needs amazon/aws</i>
	Set the given word as the callback for AWS:cmd to use (if the cb was null ; otherwise, use the cb given). The default callback sets the value returned by AWS:rc , which will contain the last return code from any AWS command. If you do multi-task AWS operations, this is probably not sufficient.
cli	s -- (as of 23.03)
	<i>needs amazon/aws</i>
	Set the aws CLI command string. The default is just aws , but you may need to provide a full path name.
cmd	cmd [opts] cb -- (as of 23.03)
	<i>needs amazon/aws</i>
	Execute the given command and options, asynchronously shelling the aws CLI utility (which you must have installed on your system)
cp	args cb -- (as of 23.03)
	<i>needs amazon/aws</i>
	Issue the AWS CLI cp command to copy files to and from your AWS bucket.
rc	-- n (as of 23.03)
	<i>needs amazon/aws</i>
	Returns the <i>last</i> return-code returned from an AWS operation (unless you used AWS:cb , in which case it's on you).

Blockchain

Namespace: **bc**

Description: Blockchain Framework

word	sed/description
+block	chain x -- chain
	<i>needs blockchain/framework</i> Create a new block containing the <i>string</i> or <i>buffer</i> data item x and add it to the given chain
.blocks	chain -- chain
	<i>needs blockchain/framework</i> Prints the blocks in the given blockchain
add-block	chain block -- chain
	<i>needs blockchain/framework</i> Internal <i>word</i> which adds the given block to the chain. You will use bc:+block instead
block-hash	block -- block hash
	<i>needs blockchain/framework</i> Calculates the hash for the given block, assuming the values it contains are correct
block@	chain ix -- chain block
	<i>needs blockchain/framework</i> Given a chain and a <i>number</i> block index, return the block. Return null if there is no such block. In future will accept the hash
first-block	-- block
	<i>needs blockchain/framework</i> Returns the first, or "genesis" block for the chain. NOTE: currently, all genesis blocks are identical
hash	-- v

word	sed/description
	<p><i>needs blockchain/framework</i></p> <p>A global <i>var</i> which contains the name of the hash to be used. By default, this is 'blake'. NOTE: if you are using blockchains and you also need to use other hash algos, you should either make certain to save and restore the hash in your non-blockchain code, or ensure the blockchain code runs on a separate task (e.g. thread)</p>
last-block	chain -- chain block
	<p><i>needs blockchain/framework</i></p> <p>Returns the last block in the chain</p>
load	chain file -- chain flag
	<p><i>needs blockchain/framework</i></p> <p>Loads a BC_MEM blockchain from a file, where it was saved previously with bc:save</p>
new	-- chain
	<p><i>needs blockchain/framework</i></p> <p>Sets the hash to use and initializes the blockchain. Returns a new and empty blockchain to use. The blockchain is set initially to be "in-memory". NOTE: a blockchain should not be shared between tasks, since the hash algo it sets is task-specific</p>
save	chain file -- chain flag
	<p><i>needs blockchain/framework</i></p> <p>For a BC_MEM blockchain, saves it to a file, so it may be restored with bc:load</p>
set-sql	chain file-name -- chain flag
	<p><i>needs blockchain/framework</i></p> <p>Sets the chain to be a BC_SQL chain. If file-name doesn't exist, a new SQL file is created and the chain initialised. Otherwise, the chain is loaded from the file. If the file exists but is not a valid SQL file or doesn't contain a valid chain, false is returned on TOS; otherwise, true is returned.</p>
validate	chain -- chain flag
	<p><i>needs blockchain/framework</i></p> <p>Validate that the entire chain is valid</p>
validate-block	chain block -- chain block flag

word	sed/description
	<i>needs blockchain/framework</i>
	Validate that the block's hash is valid, and that its prior hash is the previous block's hash

Bloom filters

Namespace: **bloom**

Description: Bloom filters

word	sed/description
add	filter value -- filter
	<i>needs utils/bloomfilter</i>
	Adds the <i>string</i> value to the Bloom filter
filter	size #hash -- filter
	<i>needs utils/bloomfilter</i>
	Create a new Bloom filter , which can have up to size items, using #hash different hashes.
in?	filter value -- filter flag
	<i>needs utils/bloomfilter</i>
	Checks if the <i>string</i> value is in the Bloom filter . If it returns false , the value is definitely not in the filter. Otherwise, depending on the filter parameters, it is probably in it.

Bluetooth

Namespace: **bt**

Description: Bluetooth discovery and I/O

word	sed/description
accept	bt -- bt'

word	sed/description
	<p><i>Professional version</i></p> <p>Given a bt created with bt:listen , waits for a connection to be created. When it is, returns bt2 which can be used with bt:read and bt:write . Returns null if there was an error.</p> <p>See also: bt:listen bt:read bt:write</p>
ch!	bt svcuuid uuid b -- bt T (as of 16.11)
	<p><i>Professional version</i></p> <p>Writes a BLE characteristic identified by the string uuid in the service identified by the string svcuuid on the BT connection given in bt . The buffer will be written; you must ensure that it contains data appropriate for the service and characteristic. Returns true if it succeeded initiating the write, false otherwise.</p> <p>NOTE: Currently unimplemented as of 20.01</p> <p>See also: bt:ch@</p>
ch@	bt svcuuid uuid -- bt b (as of 16.11)
	<p><i>Professional version</i></p> <p>Reads a BLE characteristic identified by the string uuid in the service identified by the string svcuuid on the BT connection given in bt . A buffer will be returned (or null if it was unable to read).</p> <p>NOTE: Currently unimplemented as of 20.01</p> <p>See also: bt:ch!</p>
connect	m -- bt
	<p><i>Professional version</i></p> <p>Given a map representing information about a BT device, as returned from bt:scan , returns a new bt which may be used with bt:read and bt:write . If it fails to connect, returns null .</p> <p>Additional keys in m may be:</p> <ul style="list-style-type: none"> • "l2cap": optional; if true use L2CAP protocol, otherwise use RFCOMM (the default) • "port" - a number defining the port to use • "uuid" - a string containing the UUID of the service to connect to <p>See also: bt:scan bt:read bt:write</p>
disconnect	bt -- bt (as of 16.12)
	<p><i>Professional version</i></p> <p>Given a bt returned from bt:connect or bt:leconnect , disconnect the connection (if the bt is currently connected).</p>

word	sed/description
init	-- T (as of 19.09)
	Ensures the Bluetooth system is running; returns false if not.
leconnect	m -- bt (as of 16.11)
	<p><i>Professional version</i></p> <p>Given a map returned from bt:lescan , returns a new bt which may be used with bt:read and bt:write . If it fails to connect, returns null .</p> <p>See also: bt:lescan bt:read bt:write</p>
lescan	w n -- (as of 16.11)
	<p><i>Professional version</i></p> <p>Scans for nearby Bluetooth Low Energy (BLE) devices. w is invoked for any devices found. Its SED is: m -- T , where m has information about the device found, and the return value is true to continue scanning or false to stop scanning.</p> <p>The scan will continue as long as any devices were found within n seconds. If no devices were found within n seconds, the scan will cease and w will be invoked with a parameter of null . w is invoked on a separate task (thread) and should not do more than save the information and set a flag or similar.</p>
listen	m -- bt
	<p><i>Professional version</i></p> <p>Given a map defining what service to bind to and listen on etc., returns a new bt item which may be used with bt:accept . If it fails to connect, returns null .</p> <p>The keys in m are:</p> <ul style="list-style-type: none"> • "l2cap" - optional; if true use L2CAP protocol, otherwise use RFCOMM (default RFCOMM) • "port" - a number defining the port to use • "uuid" - a string containing the UUID of the service to publish <p>See also: bt:accept</p>
on?	-- T (as of 16.11)
	<p><i>Professional version</i></p> <p>Returns true if the Bluetooth system of the machine is on, false otherwise.</p>
read	bt s n -- bt s n

word	sed/description
	<p><i>Professional version</i></p> <p>Same as f:read but for a bt connection.</p> <p>See also: bt:write bt:connect</p>
scan	w n --
	<p><i>Professional version</i></p> <p>Scans for nearby Bluetooth (BT) devices for n seconds. The word w is invoked for each device found, and has the SED: m -- T, where m is a map with the device information. It is passed null when the scan is done. It must return true to continue scanning, or false to terminate the scan.</p> <p>The map passed to w for each device will contain the keys "id" and "name", where "id" is the 48-bit identifier and "name" is the friendly name given the device by its owner.</p>
service?	s uuid -- a null
	<p><i>Professional version</i></p> <p>Scans the BT device with string identifier s (as returned from bt:scan) for services using the string UUID (or null for all services). Returns an array of service descriptors or null if none are available.</p>
services?	bt w n -- bt (as of 16.11)
	<p><i>Professional version</i></p> <p>Takes a bt returned from bt:leconnect representing a BLE peripheral, and invokes w for each service the peripheral advertises. n is a timeout value in seconds for the service scan to complete.</p> <p>Implemented on Android, iOS, and macOS.</p>
write	bt s -- bt n
	<p><i>Professional version</i></p> <p>Same as f:write but for a bt connection.</p> <p>See also: bt:read bt:connect</p>

BSON

Namespace: **bson**

Description: BSON parse

word	sed/description
parse	b w -- (as of 23.06)
	<p>Parse a buffer of BSON data, invoking the callback w after each document (map or array). The SED of w is x -- T where x is a map or array parsed from the BSON (as a "document"), and the return value is true to continue or false to stop. If there is an error in parsing, null is passed to the callback, and t:err? explains the problem.</p>

Buffer

Namespace: **b**

Description: Buffers represent a memory area with a specific length

word	sed/description
!	b ix n -- b'
	<p>Put the byte n at offset ix in the buffer. If you wish to <i>modify</i> the original buffer, you must invoke b:writable on it first.</p> <p>See also: b:@ b:writable</p>
+	b1 b2 -- b3
	<p>Appends the buffer b2 to the buffer b1 yielding a new buffer.</p>
/	b n -- a b a1 -- a2
	<p>Split the buffer b at n , returning a two-element array [head, tail] . If n is negative, splits b from the end rather than at the start.</p> <p>If an array of numbers a1 is given, then it returns an array containing buffers which represent the original b split at the sizes (not offsets) indicated. Ex: splitting a buffer with 20 bytes using [1,2] will give a three-element array with the first byte, then the next two bytes, with the remainder of 7 bytes in the last element.</p> <p>See also: s:/</p>
1+	b ix -- b (as of 19.09)
	<p>Add 1 to the byte at offset ix in the buffer. Modifies b , so it must be writable.</p>
1-	b ix -- b (as of 19.09)

word	sed/description
	Subtract 1 from the byte at offset ix in the buffer. Modifies b , so it must be writable.
<>	b ix1 ix2 -- b (as of 22.07)
	Swaps the bytes at those indices in the buffer. <i>Does</i> modify the original buffer! If you don't want to modify the original, take care to clone it first.
=	b1 b2 -- T
	Compare the two buffers, returning true if they are byte-wise equal.
>base16	sb -- s (as of 21.02)
	<i>Professional version</i> Takes a string or buffer and converts it to a RFC-4648 base16 encoded string
>base32	sb -- s (as of 21.02)
	<i>Professional version</i> Takes a string or buffer and converts it to a RFC-4648 base32 encoded string
>base64	b -- s
	Encode buffer (or string) in base64 encoding. See also: b:base64>
>base85	sb -- s (as of 21.02)
	<i>Professional version</i> Takes a string or buffer and converts it to a RFC-4648 base85 encoded string
>hex	s -- b
	Convert the 'hex dump' string into the bytes it represents. See also: b:hex>
>mpack	x -- b T (as of 18.08)

word	sed/description
	<p><i>Professional version</i></p> <p>Converts any 8th item to a buffer containing the equivalent MessagePack binary format. TOS is true if the conversion succeeded, false otherwise.</p>
@	b ix -- b n
	<p>Returns the byte at offset ix in the buffer. Returns null if the offset is beyond the bounds of the buffer.</p> <p>See also: b:!</p>
append	b1 b2 -- b1 (as of 17.10)
	<p>Appends the buffer or string b2 to the buffer b1 , modifying the original buffer.</p>
base16>	s -- b (as of 21.02)
	<p><i>Professional version</i></p> <p>Takes a string encoded in RFC-4648 base16 and returns a decoded buffer.</p>
base32>	s -- b (as of 21.02)
	<p><i>Professional version</i></p> <p>Takes a string encoded in RFC-4648 base32 and returns a decoded buffer.</p>
base64>	s -- b
	<p>Decode string s from base64 encoding into a buffer.</p> <p>See also: b:>base64</p>
base85>	s -- b (as of 21.02)
	<p><i>Professional version</i></p> <p>Takes a string encoded in RFC-4648 base85 and returns a decoded buffer.</p>
bit!	b ix n -- b (as of 18.04)
	<p>Sets the value of the bit at the given index to n , where position 0 is the low bit of the first byte in the buffer. ix must be within the bounds of b . n is evaluated as 1 if true , or 0 otherwise.</p> <p>Modifies b , so it must be writable!</p>
bit@	b ix -- b n (as of 18.04)

word	sed/description
	<p>Gets the value of the bit at the given index in the buffer, treating it as an array of bits. Position 0 is the low bit of the first byte. Returns null if n is beyond the bounds of b .</p> <p>See also: b:bit!</p>
clear	b -- b
	<p>Overwrites the contents of the b with 0.</p> <p><i>Note:</i> this modifies the original buffer itself!</p> <p>See also: s:clear</p>
compress	b -- b'
	<p>Exactly analogous to s:compress .</p> <p>See also: s:compress b:expand</p>
conv	b from to -- b' b from to -- n null (as of 16.05)
	<p>Converts the buffer from the character encoding from to to , returning a converted buffer. If it was unable to perform the conversion, returns an error code and null .</p> <p><i>Note:</i> Linux and RPI users must have installed the "libiconv" library for this to work. The errcode (if null was returned) is one of:</p> <ul style="list-style-type: none"> • 1 : no libiconv library • 2 : unknown charset • 3 : error converting text
each	b w -- (as of 16.02)
	<p>Invokes w for each byte of the buffer. The SED of w is: ix n -- , where ix is the index of the byte in the buffer and n is the value of the byte at that index. The original buffer is <i>not</i> present on the stack while this word is running.</p>
each!	b w -- (as of 19.04)
	<p>Same as b:each , but only passes the byte value, omitting the index item, to the word. The SED of w is: n -- .</p> <p>See also: b:each</p>
each-slice	b n w -- (as of 17.04)

word	sed/description
	<p>Invokes w for each slice n bytes wide of the buffer. It is w's responsibility to consume the buffer. The SED of w is: ix b --, where the ix is the index of the slice in the b, and b is the slice of b at ix times n bytes in the original. Each slice is n wide, with the possible exception of the last one.</p>
expand	<p>b n -- b'</p> <p>Exactly analogous to s:expand.</p> <p>See also: s:expand b:compress</p>
fill	<p>b n -- b' b ix sb -- b'</p> <p>Fills the contents of the b with the Unicode character n, or fill it with the contents of the string or buffer sb starting at offset ix in b.</p> <p><i>Note:</i> if you want to modify the original buffer, you <i>must</i> use b:writeable to permit that; otherwise, a new buffer is created.</p> <p>See also: s:fill</p>
getb	<p>buf -- buf b</p> <p><i>needs buf/getb</i></p> <p>Read a single byte from the given <i>buffer</i>. If no byte is available, null is returned.</p>
hex>	<p>b -- s (as of 16.11)</p> <p>Converts a buffer into its "hex dump" representation.</p> <p>See also: b:>hex</p>
len	<p>b -- b n</p> <p>Returns the length of the buffer in bytes.</p>
mem>	<p>n size -- b (as of 17.01)</p> <p>DANGEROUS! Returns a buffer containing a copy of the contents of the memory at address n for size bytes.</p> <p><i>Note:</i> if you give an invalid memory address and/or size, you may cause 8th to crash. Also, if you allow arbitrary code to access this word, you open a security hole, so use "w:forget" or "G:only" if you allow arbitrary code to be evaluated.</p>
move	<p>b dst src n -- b (as of 18.01)</p>

word	sed/description
	Moves n bytes of the buf from src to dst . The original buf is modified!
mpack-compat	T -- (as of 20.08)
	<p><i>Professional version</i></p> <p>By default, the mpack words add a compatibility tag to all future versions of 8th to unpack older versions. However, that means 8th's packing is not decipherable to third-party programs. To make the encoding acceptable to outsiders, pass true to this word.</p>
mpack-date	T -- (as of 18.08)
	<p><i>Professional version</i></p> <p>If true , then the b:>mpack word will convert a date into the MessagePack date extension format. This loses timezone information, but is necessary if you interoperate with 3rd-party software using standard MessagePack. The default is false , which means that date information will round-trip correctly between 8th instances. This only affects serialization; b:mpack> will still deserialize date data sent in 8th format, as well as MessagePack format.</p>
mpack-ignore	T -- (as of 18.08)
	<p><i>Professional version</i></p> <p>If true , the b:mpack> word will ignore any extended types it doesn't know about. The default is false , which causes an error on unknown extended types.</p>
mpack>	b -- x flag (as of 18.08)
	<p><i>Professional version</i></p> <p>Converts a buffer containing MessagePack binary data into the appropriate 8th item. Returns true and the item if successful, or false and null on failure to convert.</p>
n!	b ix n T n' -- b (as of 21.06)
	<p>Inverse of b:n@ . Stores the number n' at the byte-offset ix in the buffer. The bits value n is one of 1,2,4 or 8.</p> <p>Modifies b , so requires the buffer be writable, otherwise does nothing.</p> <p>See also: b:n@</p>
n+	b ix n -- b (as of 19.09)

word	sed/description
	Adds the number to the byte at offset ix in the buffer. Modifies b , so it must be writable.
n@	b ix n T -- b n' (as of 21.06)
	<p>Gets values from the buffer as numbers of byte-size n , a byte-offset ix . If T is true , treat the number as signed, otherwise unsigned. n must be one of '1', '2', '4', or '8'.</p> <p>Ex: 2 2 true b:n@ will get a signed 16-bit number from byte-offset 2 in the buffer.</p> <p>If the index+bittedness is out of range of the buffer, null is returned.</p> <p>See also: b:n!</p>
new	x -- b
	<p>Create a new buffer. If x is a:</p> <ul style="list-style-type: none"> • number: create a buffer of size x bytes which is NOT initialized (use b:clear or b:fill to do that) • string: create a buffer as big as the contents of the string, with a copy of the string's contents • buffer: create a clone of the original one • array of numbers: create a buffer containing the byte-values of the numbers in the array (one byte per number only).
op	source key w -- b
	<p>Takes two buffers and invokes the word w for each byte of source , and a corresponding byte from key (modulo the size of key). The resultant byte goes into the corresponding byte of b .</p> <p>This can be used, for example, to perform an "xor" of a password against a block of data. The result is a buffer the same size as source .</p> <p>The SED of w is: n1 n2 -- n3 , where n1 is a byte from source , and n2 is a corresponding byte from key .</p>
op!	b ix w -- b (as of 22.03)
	Invokes w on the byte at offset ix in the buffer, which must be writable. The SED of w is n -- n , the resultant value is placed back in b at offset ix .
pad	b len -- b' (as of 20.05)

word	sed/description
	<p><i>needs buf/pad</i></p> <p>Pad the given buffer to a multiple of 'len' bytes, using the number of bytes as the filler. Always pads the buffer, even if it is already a multiple of 'len' Can only use a 'len' up to 0xFF</p> <p>See also: b:unpad</p>
rev	b -- b' (as of 16.04)
	Reverses the order of bytes in a buffer.
search	haystack needle -- haystack n haystack ofs needle -- haystack n
	<p>Search the buffer haystack for the first occurrence of the string or regex or number needle , returning the numeric offset in the buffer of the found data, or null . If ofs is given it's a number of bytes to search from the start of the buffer. If needle is a number, the lowest byte of it is searched for (e.g. it's treated as a "uint8_t").</p> <p>See also: s:search</p>
shmem	s n T -- b
	<p>Returns a buffer containing "shared memory", given a string to use as an identifier for an IPC shared-memory object, a number specifying its size, and a boolean indicating if the memory should be read-only (if true). Returns null if this was not possible.</p>
slice	b1 ix n -- b2
	<p>Returns a slice of the buffer b1 , semantically the same as s:slice . b2 is a "slice" of b1 , beginning at offset ix for n bytes (not characters!). A negative value of n means "take the rest of the buffer".</p> <p>See also: s:slice</p>
splice	b1 b2 ix -- b' (as of 19.01)
	Replaces a portion of the buffer b1 starting at offset ix with the contents of the buffer b2 . If b1 is writable, then b1 itself is modified and returned.
ungetb	buf b -- buf
	<p><i>needs buf/getb</i></p> <p>Take the given byte and "unget" it, so that the next b:getb will return it.</p>
unpad	b -- b' (as of 20.05)

word	sed/description
	<p><i>needs buf/pad</i></p> <p>Undoes the padding done by b:pad. Do not use it on a <i>buffer</i> which was not padded!</p> <p>See also: b:pad</p>
writable	b T -- b (as of 17.01)
	<p>Make the buffer writable if T is true . Normally, attempting to modify a buffer will return a modified clone of the original. This allows modifying the original. Use with caution!.</p> <p>See also: b:!</p>
xor	b n -- b' b1 b2 -- b' (as of 16.02)
	<p>Applies a byte-wise XOR between every byte of the buffer and either the Unicode character or the buffer b2 . Repeats until every byte of the input buffer has been XORed.</p> <p><i>Note:</i> if the original buffer is writable, its contents are overwritten; otherwise, a new buffer is created.</p>

calendar

Namespace: **cal**

Description: Calendar utilities

word	sed/description
(.hebrew)	month day year -- S
	<p><i>needs calendar/hebrew</i></p> <p>Returns a <i>string</i> with the given Hebrew date.</p>
(.islamic)	year month day -- s
	<p><i>needs calendar/islamic</i></p> <p>Returns a <i>string</i> corresponding to the Islamic date given.</p>
.hebrew	month day year --
	<p><i>needs calendar/hebrew</i></p> <p>Prints the given Hebrew date.</p>

word	sed/description
.islamic	year month day --
	<i>needs calendar/islamic</i> Prints the Islamic date given.
>hebepoch	gy -- hy
	<i>needs calendar/hebrew</i> Convert Gregorian year gy to Hebrew year hy .
>jdn	d -- n (as of 20.01)
	<i>needs calendar/julian</i> Converts a <i>date</i> to a "Julian day number" as used in astronomy See also: cal:jdn>
Adar	-- 12
	<i>needs calendar/hebrew</i> Constant representing the twelfth Hebrew month.
Adar2	-- 13
	<i>needs calendar/hebrew</i> Constant representing the thirteenth Hebrew month, in a leap year.
Av	-- 5
	<i>needs calendar/hebrew</i> Constant representing the fifth Hebrew month.
Elul	-- 6
	<i>needs calendar/hebrew</i> Constant representing the sixth Hebrew month.
Heshvan	-- 8
	<i>needs calendar/hebrew</i> Constant representing the eighth Hebrew month.
Iyar	-- 2

word	sed/description
	<i>needs calendar/hebrew</i> Constant representing the second Hebrew month.
Kislev	-- 9
	<i>needs calendar/hebrew</i> Constant representing the ninth Hebrew month.
Nissan	-- 1
	<i>needs calendar/hebrew</i> Constant representing the first Hebrew month.
Shevat	-- 11
	<i>needs calendar/hebrew</i> Constant representing the eleventh Hebrew month.
Sivan	-- 3
	<i>needs calendar/hebrew</i> Constant representing the third Hebrew month.
Tammuz	-- 4
	<i>needs calendar/hebrew</i> Constant representing the fourth Hebrew month.
Tevet	-- 10
	<i>needs calendar/hebrew</i> Constant representing the tenth Hebrew month.
Tishrei	-- 7
	<i>needs calendar/hebrew</i> Constant representing the seventh Hebrew month.
days-in-hebrew-year	hyr -- days
	<i>needs calendar/hebrew</i> How many days are in the given year?.

word	sed/description
displaying-hebrew	-- v
	<i>needs calendar/hebrew</i> A <i>var</i> which controls whether the Hebrew date displays in Hebrew or English. Defaults to false .
fixed>hebrew	date -- month day year
	<i>needs calendar/hebrew</i> Converts the fixed-date into the Hebrew date.
fixed>islamic	fixed -- year month day
	<i>needs calendar/islamic</i> Convert the given fixed-date to the Islamic date.
gershayim	s -- s2
	<i>needs calendar/hebrew</i> Inserts the Hebrew "gershayim" character (0x05f4) as appropriate for a Hebrew number.
hanukkah	hy -- fixed
	<i>needs calendar/hebrew</i> Returns the fixed-date of the first day of Hanukkah in the given Hebrew year.
hebrew-epoch	-- heepoch
	<i>needs calendar/hebrew</i> Fixed date corresponding to the beginning of AM (anno mundi, Hebrew year).
hebrew-leap-year?	yr -- flag
	<i>needs calendar/hebrew</i> Return 1 if yr is a Hebrew leap year or 0 otherwise.
hebrew>fixed	month day year -- date
	<i>needs calendar/hebrew</i> Convert the Hebrew month, day and year given into a "fixed date", a <i>number</i> .
hebrewtoday	-- month day year

word	sed/description
	<i>needs calendar/hebrew</i> Returns a the Hebrew date corresponding to today.
hmonth-name	nr -- s
	<i>needs calendar/hebrew</i> Returns a <i>string</i> containing the name of the Hebrew month given as a <i>number</i> . Returns Hebrew if d:displaying-hebrew is true .
islamic.epoch	-- epoch
	<i>needs calendar/islamic</i> Returns the fixed-date of the start of the Islamic epoch.
islamic>fixed	year month day -- fixed
	<i>needs calendar/islamic</i> Convert the given Islamic date to a fixed-date.
islamictoday	-- year month day
	<i>needs calendar/islamic</i> Returns the Islamic date corresponding to today.
jd>	n -- d (as of 20.01)
	<i>needs calendar/julian</i> Converts a "Julian day number" as used in astronomy to a <i>date</i> See also: cal:>jdn
last-day-of-hebrew-month	month year -- day
	<i>needs calendar/hebrew</i> Returns the last day of the given month in a particular year. Either 29 or 30, takes into account the dehiyot etc.
number>hebrew	n -- s
	<i>needs calendar/hebrew</i> Returns a <i>string</i> which is the traditional Hebrew number corresponding to the <i>number</i> given.

word	sed/description
omer	hy -- days
	<i>needs calendar/hebrew</i> Returns the "count of the omer" for the specific fixed-date. If that date is outside the omer period, returns 0.
pesach	hy -- fd
	<i>needs calendar/hebrew</i> Returns the fixed-date of Passover in the given Hebrew year.
purim	hy -- fd
	<i>needs calendar/hebrew</i> Returns the fixed-date of Purim in the given Hebrew year.
rosh-chodesh?	fd -- flag
	<i>needs calendar/hebrew</i> Returns true if the given fixed-date is "Rosh Chodesh".
rosh-hashanah	hy -- fd
	<i>needs calendar/hebrew</i> Returns the fixed-date of Rosh Hashana in the given Hebrew year.
shavuot	hy -- fd
	<i>needs calendar/hebrew</i> Returns the fixed-date of Shavuot in the given Hebrew year.
taanit-esther	hy -- fixed
	<i>needs calendar/hebrew</i> Returns the fixed-date of the Fast of Esther in the given Hebrew year.
tisha-beav	hy -- fixed
	<i>needs calendar/hebrew</i> Returns the fixed-date of the Fast of the Ninth of Av in the given Hebrew year.
yom-haatsmaut	hy -- fixed

word	sed/description
	<i>needs calendar/hebrew</i> Returns the fixed-date of the Israeli Independence Day in the given Hebrew year.
yom-kippur	hy -- fd
	<i>needs calendar/hebrew</i> Returns the fixed-date of Yom Kippur in the given Hebrew year.

Colors

Namespace: **clr**

Description: Colors

word	sed/description
>hsva	x -- a (as of 24.04)
	Returns the color converted from ARGB to HSVA as an array [hue,saturation,value,alpha].
complement	x -- n (as of 24.04)
	Returns a color "complimentary" to the given one.
dist	x1 x2 -- n (as of 24.04)
	Returns the "distance" between two colors, a value between 0 and 1 where 0 means identical.
gradient	x1 x2 n -- a (as of 24.04)
	Returns a gradient of colors from x1 to x2, in n steps.
hsva>	a -- n (as of 24.04)
	Inverse of clr:>hsva . Takes an array of HSVA and returns the RGBA value.
invert	x -- n (as of 24.04)
	"inverts" the given color

word	sed/description
nearest-name	x -- s (as of 24.04)
	<i>needs nk/color</i> Takes a color name or value, and returns the name of the color most nearly matching it from our internal list of colors.
parse	x -- n (as of 24.04)
	Converts a color definition (number, string, or array of values) to an RGBA number value of the color.

Complex

Namespace: **c**

Description: Numbers: complex (native doubles)

word	sed/description
*	c1 c2 -- c3 (as of 18.07)
	Multiplies two complex numbers. See also: c:new c:+ c:abs c:>ri c:conj c:arg c:=
*	c1 c2 -- c3
	<i>needs math/complex</i> Returns the complex product of the two inputs.
+	c1 c2 -- c3
	<i>needs math/complex</i> Adds two complex numbers returning their complex sum.
+	c1 c2 -- c3 (as of 18.07)
	Adds two complex numbers. See also: c:new c:* c:abs c:>ri c:conj c:arg c:=
=	c1 c2 -- T

word	sed/description
	<i>needs math/complex</i> Compares two complex numbers and returns true if they are equal.
=	c1 c2 -- T (as of 18.07)
	Compares two complex numbers, returning true if they are equal. See also: c:new c:+ c:* c:>abs c:>ri c:conj c:arg
>polar	c -- c' (as of 24.04)
	<i>needs math/complex</i> Returns the complex number in "polar form"
>polar	c -- c' (as of 24.04)
	Returns the complex number in "polar form".
>ri	c -- real imag
	<i>needs math/complex</i> Returns the "real" and "imaginary" components of the complex number.
>ri	c1 -- real imag (as of 18.07)
	Returns the "real" and "imaginary" components of the complex number. See also: c:new c:+ c:* c:>abs c:conj c:arg c:=
^	c n -- c' (as of 24.04)
	<i>needs math/complex</i> Raises the complex number c to the exponent n, analogous to n:^ .
^	c n -- c' (as of 24.04)
	Raises the complex number c to the exponent n, analogous to n:^ .
abs	c -- n
	<i>needs math/complex</i> Returns a the absolute-value (e.g. magnitude) of the input.

word	sed/description
abs	c1 -- n (as of 18.07)
	Returns the absolute-value of the complex number. See also: c:new c:+ c:* c:>ri c:conj c:arg c:=
arg	c -- n
	<i>needs math/complex</i> Returns the "argument" (e.g. angle of the radius, in radians) of the input.
arg	c -- n (as of 18.07)
	Returns the "argument" (the angle of the radius in radians) of the complex number. See also: c:new c:+ c:* c:>abs c:>ri c:conj c:=
conj	c -- c'
	<i>needs math/complex</i> Returns the "complex conjugate" of the input. If c is 1+2i, c' is 1-2i.
conj	c -- c' (as of 18.07)
	Returns the complex-conjugate of the complex number. See also: c:new c:+ c:* c:>abs c:>ri c:arg c:=
im	c -- im
	<i>needs math/complex</i> Returns a number which is the "imaginary" component of the complex number c.
im	c -- n (as of 24.04)
	Returns the imaginary component of the complex
log	c -- c' (as of 24.04)
	<i>needs math/complex</i> Returns the "principal value" of the natural logarithm of c .
log	c -- c' (as of 24.04)

word	sed/description
	Returns the "principal value" of the natural logarithm of c .
n>	n -- c (as of 24.04)
	Converts a number to a complex, e.g. creates a new complex number, n + 0i
n>	n -- c
	<i>needs math/complex</i> Converts a number to the complex c, represented as "n+0i".
new	real imag -- c a -- c (as of 18.07)
	Create a new <i>complex number</i> based on native doubles (not regular 8th numbers). This is a faster implementation than the one in the "math/complex" library, but it does not allow unlimited precision. The parameters are either two numbers or an array of two numbers, in the order "real, imag". The numbers given must fit into a native double or they will be truncated, with unpredictable results. May return null if inappropriate values are given. See also: c:+ c:* c:abs c:>ri c:conj c:arg c:=
new	a -- c
	<i>needs math/complex</i> From either a pair of numbers, or an array containing two numbers, returns a new complex representing "real + i * imag".
polar>	n1 n2 -- c (as of 24.04)
	<i>needs math/complex</i> Returns the complex number determined by the "polar coordinates" n1 (abs(c)), and n2 (arg(c)).
polar>	n1 n2 -- c (as of 24.04)
	Returns the complex number determined by the "polar coordinates" n1 (abs(c)), and n2 (arg(c)).
re	c -- re
	<i>needs math/complex</i> Returns a number which is the "real" component of the complex number c.
re	c -- n (as of 24.04)

word	sed/description
	Returns the real component of the complex

Console

Namespace: **con**

Description: Console I/O

word	sed/description
>redir	--
	<i>needs console/redirect</i> Redirects console output (from "." etc) to a string
accept	n a -- s
	Requests a string of maximum length n from the console (a length of 0 implies maximum, which is 4096 characters). If desired, a history array of strings may be provided (or null if no history is needed). The same editing keys as with regular console input are allowed. Returns the entered string or null if Ctrl+C was pressed. See also: meta:console con:key con:accept-pwd
accept-nl	T -- (as of 23.03)
	If false , con:accept will not print an initial newline. The default is true .
accept-pwd	n -- s
	Same as con:accept , except it does not display the characters entered. Instead, it shows '*' for each character. The same editing keys are allowed as with con:accept . See also: con:accept con:key
alert	-- (as of 22.05)
	IMMEDIATE Invoked from the 'con:accept' words if an 'alert' is required. The default does nothing.
ansi?	-- T (as of 20.05)

word	sed/description
	Returns true if the console supports ANSI sequences. Only relevant on Windows.
black	--
	<i>needs console/loaded</i> Set the console foreground color to black. Must be paired with one of the con:on... color commands.
blue	--
	<i>needs console/loaded</i> Set the console foreground color to blue. Must be paired with one of the con:on... color commands.
clreol	--
	Clear the current console line from the current position until the end of the line. See also: meta:console con:cls
cls	--
	Clear the console screen and put the cursor on the first row. See also: meta:console con:clreol
ctrlid-empty	T -- (as of 23.07)
	If T is true , make the REPL only quit on Ctrl+D if the input buffer is empty. The default is false , meaning always quit when Ctrl+D is pressed.
cyan	--
	<i>needs console/loaded</i> Set the console foreground color to cyan. Must be paired with one of the con:on... color commands.
down	--
	Move the console position down one. See also: meta:console con:left con:right con:up
file>history	T s -- (as of 22.03)

word	sed/description
	<p><i>needs console/history</i></p> <p>Restore console history from a file. If T is true , overwrites current history; otherwise, appends to it.</p>
free	--
	<p>Closes the console window.</p> <p><i>Note:</i> Windows OS only.</p>
getxy	-- row col
	Returns the position of the console cursor.
gotoxy	row col --
	<p>Set the console cursor position.</p> <p>See also: meta:console con:getxy</p>
green	--
	<p><i>needs console/loaded</i></p> <p>Set the console foreground color to green. Must be paired with one of the con:on... color commands.</p>
history-handler	w -- (as of 22.03)
	<p><i>needs console/history</i></p> <p>Set the history handling word. Its SED is s -- . The default handler saves to the file given in con:history>file , and runs in the REPL task. You can do other things if you like</p>
history>file	s -- (as of 22.03)
	<p><i>needs console/history</i></p> <p>Save console history to the named file, one line per item. If null is given, stop saving history. If the file name cannot be created or opened, history will not be saved.</p>
init	-- T (as of 24.01)
	Returns true if the console subsystem could be initialized; otherwise returns false and sets t:err? .
key	-- n

word	sed/description
	<p>Returns the key-code of a pressed key, waiting for a key to be pressed.</p> <p>See also: meta:console con:key? con:accept con:accept-pwd</p>
key?	-- T
	<p>Returns true if con:key would return a value without blocking, false otherwise.</p> <p>See also: con:key meta:console</p>
left	--
	<p>Move the console position left one.</p> <p>See also: meta:console con:right con:up con:down</p>
load-history	a T --
	<p>Load the console's history from the array of strings. If T is true , overwrite the current history; otherwise, append the array's contents to the current history.</p> <p>See also: meta:console</p>
magenta	--
	<p><i>needs console/loaded</i></p> <p>Set the console foreground color to magenta. Must be paired with one of the con:on... color commands.</p>
max-history	n -- (as of 23.03)
	<p>Changes the default console history size from 100 to n . The maximum is 100000.</p> <p>See also: meta:console</p>
onBlack	--
	<p><i>needs console/loaded</i></p> <p>Set the console background color to black.</p>
onBlue	--
	<p><i>needs console/loaded</i></p> <p>Set the console background color to blue.</p>

word	sed/description
onCyan	--
	<i>needs console/loaded</i> Set the console background color to cyan.
onGreen	--
	<i>needs console/loaded</i> Set the console background color to green.
onMagenta	--
	<i>needs console/loaded</i> Set the console background color to magenta.
onRed	--
	<i>needs console/loaded</i> Set the console background color to red.
onWhite	--
	<i>needs console/loaded</i> Set the console background color to white.
onYellow	--
	<i>needs console/loaded</i> Set the console background color to yellow.
print	s --
	Print the string to the console, taking into account console text attributes and positioning. See also: meta:console
red	--
	<i>needs console/loaded</i> Set the console foreground color to red. Must be paired with one of the con:on... color commands.
redir>	-- s

word	sed/description
	<p><i>needs console/redirect</i></p> <p>Stops console redirection and returns the captured string</p>
redir?	-- T
	<p>Returns true if the input is redirected (from a pipe for instance).</p> <p>See also: meta:console</p>
right	--
	<p>Move the console position right one.</p> <p>See also: meta:console con:left con:up con:down</p>
save-history	w -- null --
	<p>If given word, it is a callback for saving history whose SED is s -- . If null , stop saving history.</p> <p>History is not saved by default. This word causes the history to be saved from the point it is invoked with a word, until the point it is invoked with null .</p> <p>The library console/history implements history saving in a more easy-to-use manner.</p> <p>See also: meta:console</p>
size?	-- col row (as of 16.10)
	Returns the current size of the console in columns and rows.
up	--
	<p>Move the console position up one.</p> <p>See also: meta:console con:left con:right con:down</p>
white	--
	<p><i>needs console/loaded</i></p> <p>Set the console foreground color to white. Must be paired with one of the con:on... color commands.</p>
yellow	--

word	sed/description
	<p><i>needs console/loaded</i></p> <p>Set the console foreground color to yellow. Must be paired with one of the con:on... color commands.</p>

Crypto

Namespace: **cr**

Description: Encryption and decryption

word	sed/description
>aes128gcm	<p>item key -- buf</p> <p><i>needs crypto/aes128gcm</i></p> <p>Encrypt the <i>buffer</i> or <i>string</i> item with the <i>buffer</i> or <i>string</i> key using AES-128-GCM. The key will be converted into an appropriate buffer using cr:ensurekey. The encrypted <i>buffer</i> buf is returned.</p>
>aes256gcm	<p>sb sb2 -- b (as of 17.04)</p> <p>Encrypts sb using AES-256-GCM using the key sb2 returning a buffer which can be decrypted with cr:aes256gcm> . If sb2 is not a 32-byte buffer, it will be converted to one using cr:ensurekey .</p> <p>See also: cr:aes256gcm> cr:ensurekey</p>
>cp	<p>sb sb2 -- b (as of 17.04)</p> <p>Takes a string or buffer sb , and a string or buffer key sb2 , encrypts using Chacha20Poly1305, returns a buffer which can be decrypted with cr:cp> . If sb2 is not a 32-byte buffer, it will be converted to one using cr:ensurekey .</p> <p>See also: cr:cp> cr:ensurekey</p>
>cpe	<p>sb b b2 -- b3 (as of 17.04)</p> <p>Encrypts the string or buffer using the key b with Chacha20Poly1305, signs with the private Ed25519 key b2 , then boxes it up in a buffer b3 . That can be given to cr:cpe> to verify and decrypt.</p> <p>See also: cr:cpe></p>
>decrypt	<p>sb key -- cr sb key iv aad tag -- cr m -- cr</p>

word	sed/description																					
	<p>Given a buffer key containing an encryption key (see cr:genkey or cr:randkey) and a string or buffer with encrypted data, begins decryption and returns a crypt which must be passed to further crypt words, or null if there was an error. In GCM or "chacha1305" modes, the iv, add, and tag are also required.</p> <p>If a map is given, it has the same keys as for cr:>encrypt .</p> <p>See also: cr:decrypt> cr:decrypt+</p>																					
>edbox	sb b -- b2 (as of 17.04)																					
	<p>Creates a "box" from a string or buffer s and an Ed25519 private key b (created using cr:ed25519), containing a header and the signature for the item and key.</p> <p>See also: cr:edbox> cr:ed25519</p>																					
>encrypt	sb key -- cr sb key iv aad -- cr m -- cr																					
	<p>Given a buffer key containing an encryption key (see cr:genkey or cr:randkey) and a string or buffer sb with data to encrypt, starts encryption and returns a crypt which must be passed to further crypt words, or null if there was an error.</p> <p>In modes other than ECB, and for the "chacha" and "chacha1305" ciphers, a buffer iv must be given. In GCM mode or with the "chacha1305" cipher, a buffer aad may be given (or null must be passed, if no additional data is to be given). With the "chacha" cipher, the aad must be a number, used as a counter.</p> <p>If a map is given, then its keys are as follows:</p> <table><tr><th>key</th><th>kind</th><th>description</th></tr><tr><td>key</td><td>sb</td><td>encryption/decryption key</td></tr><tr><td>tag</td><td>b</td><td>GCM etc tag</td></tr><tr><td>iv</td><td>b</td><td>IV</td></tr><tr><td>aad</td><td>sb</td><td>additional authentication data</td></tr><tr><td>read</td><td>w</td><td>callback word for getting input</td></tr><tr><td>write</td><td>w</td><td>callback word for writing output</td></tr></table> <p>The callback words, if present, have the following SEDs:</p> <ul style="list-style-type: none">• read: -- m b	key	kind	description	key	sb	encryption/decryption key	tag	b	GCM etc tag	iv	b	IV	aad	sb	additional authentication data	read	w	callback word for getting input	write	w	callback word for writing output
key	kind	description																				
key	sb	encryption/decryption key																				
tag	b	GCM etc tag																				
iv	b	IV																				
aad	sb	additional authentication data																				
read	w	callback word for getting input																				
write	w	callback word for writing output																				
>nbuf	num -- buf																					
	<p><i>needs crypto/totp</i></p> <p>Convert a number into a (64-bit) buffer of the binary digits.</p>																					

word	sed/description
>rsabox	sb b -- b' (as of 17.04)
	<p>Creates a box containing a head and signature from the string or buffer sb and an RSA private key b (created using 64 cr:rsa_genkey).</p> <p>See also: cr:rsabox> cr:rsa_genkey</p>
>uuid	b -- s
	Creates a UUID string from a 16-byte buffer.
aad?	-- T (as of 17.04)
	Returns true if the current crypto settings accept "additional authentication data".
aes128box-sig	item key -- buf
	<p><i>needs crypto/aes128gcm</i></p> <p>This is the encryption header, which identifies the 'box' as being AES128GCM. The last three numbers are "id", "IV size", and "tag length" (in AEAD suites).</p>
aes128gcm>	buf key -- item flag
	<p><i>needs crypto/aes128gcm</i></p> <p>Inverse of cr:>aes128gcm. Decrypt the <i>buffer</i> buf with the <i>buffer</i> or <i>string</i> key using AES-128-GCM. The key will be converted into an appropriate buffer using cr:ensurekey. The decrypted <i>buffer</i> item is returned and flag is true, if the decryption was successful. Otherwise, the original buffer is returned and false is on TOS.</p>
aes256box-sig	-- b (as of 17.04)
	Returns a buffer containing the correct 8-byte signature for an AES-256-GCM box.
aes256gcm>	b b2 -- b' (as of 17.04)
	<p>Decrypts a buffer which was encrypted with cr:>aes256gcm , using the key buffer b2 . Returns null if the decryption failed.</p> <p>See also: cr:>aes256gcm</p>
aesgcm	-- (as of 17.04)
	Sets the encryption mode to GCM, the cipher to AES, and the GCM tag size to 16.

word	sed/description
blakehash	-- (as of 17.04)
	Sets the current hash to "blake".
chacha20box-sig	-- b (as of 17.04)
	Returns a buffer containing the correct 8-byte signature for a Chacha20Poly1305 box.
chachapoly	-- (as of 17.04)
	Sets the current encryption cipher to "Chacha20Poly1305".
cipher!	s -- (as of 16.01)
	<p>Set the cipher to use in this task. The string may be any of the ones returned by cr:ciphers . Throws an exception if given an unknown cipher name. The default value is "aes256-gcm".</p> <p>See also: cr:hash! cr:encrypt cr:decrypt</p>
cipher@	-- s (as of 16.04)
	<p>Returns the current cipher algorithm name, suitable for passing to cr:cipher! .</p> <p>See also: cr:cipher!</p>
ciphers	-- a (as of 22.04)
	Returns an array of the names of all supported ciphers.
cp>	b b2 -- b' (as of 17.04)
	<p>Decrypts a buffer which was encrypted with cr:>cp , and a buffer key b2 which was used to encrypt it. Returns a buffer with the clear-text, or null if the decryption failed.</p> <p>See also: cr:>cp cr:ensurekey</p>
cpe>	b b2 b3 -- b' (as of 17.04)
	Decrypts the buffer b created by cr:>cpe , using the key b2 and verifying with the public Ed25519 key b3 . If the decryption or verification fail, null is returned; otherwise, a buffer of plain-text is returned.
decrypt	-- (as of 17.04)

word	sed/description
	Shorthand for cr:>decrypt cr:decrypt> .
decrypt+	sb cr -- cr
	<p>Given a crypt returned by cr:>decrypt , add the contents of the string or buffer and return the same crypt for further processing, or null if there was an error.</p> <p>See also: cr:decrypt> cr:>decrypt</p>
decrypt>	cr -- b cr -- tag b
	<p>Given a crypt returned by cr:decrypt or cr:decrypt+ , finalize the decryption and return a buffer with the decrypted data or null if there was an error.</p> <p>In GCM mode or chacha1305 cipher, returns the tag as well.</p> <p>If callbacks were used, then the buffer returned on TOS will be an empty one (since the results will have been written using the callback).</p> <p>See also: cr:decrypt+ cr:>decrypt</p>
ebox-sig	b -- b' (as of 17.04)
	Creates an encryption box signature buffer from a three-byte buffer.
ecc-curves	-- a (as of 22.04)
	Returns an array of maps containing "id" and "name" of all supported ECC curves.
ecc-genkey	n -- priv pub (as of 16.01)
	<p>Returns a binary encoded ECC key-pair corresponding to n , or null if there was an error.</p> <p><i>NOTE:</i> valid values of n are the id values returned from cr:ecc-curves . Values outside of them are not accepted.</p> <p>See also: cr:ecc-secret cr:ecc-sign cr:ecc-verify</p>
ecc-secret	priv pub -- b (as of 16.01)
	<p>Generates an ECC shared secret using one party's private ECC key, and the other party's public ECC key. The result is a buffer with the shared secret, or null if there was a problem.</p> <p>See also: cr:ecc-genkey cr:ecc-sign cr:ecc-verify</p>

word	sed/description
ecc-sign	priv hash -- sig (as of 16.01)
	<p>Signs the hash using ECDSA algorithms with the private key and returns an signature buffer in sig . Returns null on error.</p> <p>See also: cr:ecc-genkey cr:ecc-secret cr:ecc-verify</p>
ecc-verify	pub hash sig -- T (as of 16.01)
	<p>Verifies the hash signature versus the public key of the sender, using ECDSA algorithms and returns true on successful verification or false otherwise.</p> <p>See also: cr:ecc-genkey cr:ecc-secret cr:ecc-sign</p>
ed25519	-- priv pub
	<p>Generates an "ed25519" key-pair in two buffers, corresponding to the new private-key and public-key.</p> <p>See also: cr:ed25519-secret cr:ed25519-sign cr:ed25519-verify</p>
ed25519-secret	priv pub -- b
	<p>Generates an ed25519 <i>shared secret</i> using one party's private key priv , and the other party's public key pub . The result is a buffer which is the shared secret, or null if there was an error.</p> <p>See also: cr:ed25519 cr:ed25519-sign cr:ed25519-verify</p>
ed25519-sign	priv hash -- sig
	<p>Given a buffer containing an ed25519 Diffie-Hellman private key of the sender, and another buffer containing a hash (could be a full message, but typically it's just a hash) of a message, <i>signs</i> the hash using DH algorithms and returns an signature buffer in sig or null if there was an error.</p> <p>See also: cr:ed25519 cr:ed25519-secret cr:ed25519-verify</p>
ed25519-verify	pub hash sig -- T
	<p>Given a buffer containing an ed25519 Diffie-Hellman public key (of the sender), another buffer containing a hash which was presumably signed by the sender, and another buffer containing the DH signature, verifies the hash signature using DH algorithms and returns true on successful verification or false otherwise.</p> <p>See also: cr:ed25519-genkey cr:ed25519-secret cr:ed25519-sign</p>

word	sed/description
edbox-sig	-- b (as of 17.04)
	Returns a buffer containing the correct 8-byte signature for a ChaCha20Poly1305Ed25519 box.
edbox>	b sb b2 -- T (as of 17.04)
	<p>Given an Ed25519 public key b , a string or buffer sb , and a "box" b2 which was created using cr:>edbox , verify that the signature is correct; returning true if it is.</p> <p>See also: cr:>edbox cr:ed25519</p>
encrypt	-- (as of 17.04)
	Shorthand for cr:>encrypt cr:encrypt> .
encrypt+	sb cr -- cr
	<p>Given the crypt cr returned by cr:encrypt , add the contents of the string or buffer and return the same crypt for further processing, or null if there was an error.</p> <p>See also: cr:>encrypt cr:encrypt></p>
encrypt>	cr -- b cr -- tag b
	<p>Given the crypt returned by cr:>encrypt or cr:encrypt+ , finalize the encryption and return a buffer b with the encrypted results or null if there was an error.</p> <p>In GCM mode or if using "chacha1305", a buffer tag will also be returned, which is used to verify the decryption succeeded.</p> <p>See also: cr:>encrypt cr:encrypt+</p>
ensurekey	sb n -- b (as of 17.04)
	<p>Ensures that a buffer of n bytes is created from the string (or non n-sized buffer). If it needs to create b , it will use the salt value "salty8thtears" and 10000 as parameters to cr:genkey . Otherwise, if sb is already a properly sized buffer, it will just past it on.</p> <p>See also: cr:genkey</p>
genkey	pwd salt iter -- key

word	sed/description
	<p>Takes the password string or buffer pwd , the string salt , the number of iterations iter and returns a PBKDF2 compliant key in a 32 byte buffer key , or null if there was an error.</p> <p>See also: cr:randkey cr:rsagenkey</p>
hash	sb -- cr
	<p>Returns the hash of a string or buffer as a crypt, or null if there was an error. The specific hash algorithm to use is determined by cr:hash! , with "blake3" being the default hash. The hash being used is task-specific, so separate tasks may use different hash functions simultaneously without concern. See also: cr:hmac .</p> <p>See also: cr:hash! cr:hash+ cr:hash>s cr:hash>b cr:hmac</p>
hash!	s -- (as of 16.01)
	<p>Set the hash to use in this task. The string may be any of the values returned by cr:hashes . The default value is "blake3".</p> <p>Throws an exception if given an unknown hash name.</p> <p>See also: cr:hash cr:cipher!</p>
hash+	sb cr -- cr
	<p>Take the string or buffer and add its data to the current hash in the crypt (returned from cr:hash or cr:hmac). Returns the crypt it was given, or null if there was an error.</p> <p>See also: cr:hash! cr:hash cr:hash>s cr:hash>b cr:hmac</p>
hash>b	cr -- b
	<p>Finalize the hash calculation of crypt cr , and return the hash value as a binary buffer, or null if there was an error.</p> <p>See also: cr:hash! cr:hash cr:hash+ cr:hash>s cr:hmac</p>
hash>s	cr -- s
	<p>Finalize the hash calculation of crypt cr , returning the hash value as a readable string, or null if there was an error.</p> <p>See also: cr:hash! cr:hash cr:hash+ cr:hash>b cr:hmac</p>

word	sed/description
hash@	-- s (as of 16.04)
	Returns the current hash algorithm name, suitable for passing to cr:hash! . See also: cr:hash!
hashes	-- a (as of 22.04)
	Returns an array of the names of all supported hashes.
hmac	sb key -- cr
	Take the string or buffer and return a crypt cr which is its HMAC-hash using the key key (which may be a string or a buffer), or null if there was an error. The rest of its semantics are exactly the same as cr:hash . See also: cr:hash! cr:hash cr:hash+ cr:hash>s cr:hash>b
hotp	key time #digits -- totp
	<i>needs crypto/totp</i> Same as cr:totp , but uses an incrementing counter rather than the time.
iv?	-- T (as of 17.04)
	Returns true if the current crypto settings require an "IV" (initialization vector).
pem-read	s -- m b -- m (as of 22.04)
	Reads a PEM-encoded item. If given a string, reads from that named file. If given a buffer, the PEM data are in-memory. Returns a map with "name", "header", and "data"; or null if the read failed.
pem-write	name header b filename -- T (as of 22.04)
	Takes strings 'name' and 'header' and 'filename', and a buffer; writes them in PEM format to the file. Returns true if successful, otherwise false .
pwd-valid?	s m -- T (as of 22.03)
	<i>needs crypto/password</i> Determines whether or not the given password meets the minimum requirements given in the map, which has numbers corresponding to cr:pwd/

word	sed/description
pwd/	s -- m (as of 22.03)
	<p><i>needs crypto/password</i></p> <p>Splits the given password string into components for analysis as to its fitness. The returned map has "uc", "lc", and "len", which are counts of number of uppercase, lowercase, and total string length. It will also have the count of characters by script, e.g. "number", "symbol", "latin", "cyrillic" etc (as per s:script?).</p>
pwd>hash	s n -- salt hash (as of 22.03)
	<p><i>needs crypto/password</i></p> <p>Given a password string and a number indicating how many random bytes of salt to create, returns a random salt string as well as the salted hash of the password</p>
rand	-- n
	<p>Generate a 64-bit cryptographically-secure pseudo-random number using the "ChaCha20" generator. Much slower than the other PRNGs, but suitable for crypto work. Each task has its own PRNG.</p> <p>See also: G:rand-pcg G:rand-jsf G:rand-jit</p>
randbuf	n -- b
	<p>Generate a buffer of size n filled with pseudo-random bytes from the cryptographically strong random source.</p> <p>See also: G:randbuf-pcg</p>
randkey	-- b
	<p>Returns a buffer containing a pseudo-random key for encryption, using a cryptographically strong random number generator, suitable for the currently chosen cipher.</p> <p>See also: G:randbuf cr:rsagenkey cr:genkey</p>
restore	X -- (as of 17.04)
	<p>Restores the crypto settings to what they were when cr:save was invoked.</p> <p>See also: cr:save</p>
root-certs	-- cert
	<p><i>needs crypto/root-certs</i></p> <p>A <i>var</i> which contains "root certificates" which you can give as the "rootcert" key of a net transaction</p>

word	sed/description
rsa_decrypt	b sb -- b'
	<p>Decrypt the string or buffer sb using the RSA private-key b , putting the results in b , or null on error.</p> <p>See also: cr:rsa_encrypt cr:rsagenkey</p>
rsa_encrypt	b sb -- b'
	<p>Encrypt the string or buffer sb with the RSA public key b , returning the encrypted data b . The RSA corresponding <i>private</i> key is required to decrypt.</p> <p>See also: cr:rsa_decrypt cr:rsagenkey</p>
rsa_sign	hash key -- b
	<p>Sign the hash of a message with the RSA private key , returning the signed buffer or null on error.</p> <p>See also: cr:rsa_verify</p>
rsa_verify	hash key sig -- T
	<p>Verify the hash of a message with the RSA public key , against the RSA signature sig given by cr:rsa_sign . Returns true if the hash was verified, false otherwise.</p> <p>See also: cr:rsa_sign</p>
rsabox-sig	-- b (as of 17.04)
	Returns a buffer containing the correct 8-byte signature for a AES-256-GCM-RSA box.
rsabox>	b sb b2 -- T (as of 17.04)
	<p>Verifies that the signature box b2 created using cr:>rsabox is correct given the RSA public key b and a string or buffer sb , returning true if it is.</p> <p>See also: cr:>rsabox cr:rsa_genkey</p>
rsagenkey	n -- priv pub
	<p>Generate an RSA key-pair of size n . Valid values of n are 1024, 2048 and 4096. Returns the DER encoded key-pair as two buffers priv and pub , or null if there was an error.</p> <p>See also: G:randbuf cr:randkey cr:rsa_encrypt cr:rsa_decrypt cr:genkey</p>

word	sed/description
save	-- X (as of 17.04)
	Returns an item encoding the current crypto settings. Use cr:restore to restore those settings. See also: cr:restore
sbox-sig	b -- b' (as of 17.04)
	Creates an "signature box signature" buffer from a three-byte buffer.
sha1-hmac	msg key -- hash
	<i>needs crypto/totp</i> Returns the SHA1-HMAC hash of the <i>buffer</i> or <i>string</i> msg given the <i>buffer</i> key .
shard	sb n m -- a (as of 17.04)
	Uses Shamir Secret Sharing to split a string or buffer into n shards, of which any m may be used to recreate the original secret. Using fewer than m shards is <i>guaranteed</i> to provide insufficient information to recover any of the secret. Returns an array of buffers, each of which is a shard of the original secret. Returns null if there was an error. Use cr:unshard to recover the original s . See also: cr:unshard
tag?	-- T (as of 17.04)
	Returns true if the current crypto settings return a "tag" value under the encrypted data, or require a tag value for decryption. Currently, this is the same as cr:aad? , but may change as new cipher suites and modes are added.
totp	key time #digits -- totp
	<i>needs crypto/totp</i> Given a key *buffer, a <i>number</i> time , and a <i>number</i> #digits to return, returns a "Time-based One-Time Password".
totp-epoch	-- var
	<i>needs crypto/totp</i> A <i>var</i> containing the epoch to use for the TOTP. Defaults to 0.
totp-time-step	-- var

word	sed/description
	<i>needs crypto/totp</i> A <i>var</i> containing the time-step to use for TOTP. Defaults to 30 seconds.
unshard	a -- b (as of 17.04)
	Recombines an array containing buffers of shards produced by cr:shard to form the original (as a buffer). Returns null if there was an error. <i>Note:</i> un-sharding might not fail to produce a result, even if the number of shards given was incorrect, or if the shards were incorrect; the <i>result</i> will simply be incorrect (and have nothing to do with the secret which was processed by cr:shard). See also: cr:shard
uuid	-- s
	Returns a cryptographically strong random UUID (RFC 4122 compliant).
uuid>	s -- b
	Returns a buffer 16 bytes long representing the given UUID, or null if the string is not in the correct format.
validate-gpg-sig	fname -- ok (as of 18.06)
	<i>needs crypto/pgp</i> Takes a <i>string</i> fname which is a file whose PGP signature(s) are to be validated, and runs gpg --verify synchronously. It returns a <i>number</i> ok which is 0 if the signature is valid; otherwise, the gpg return code (-1 if gpg couldn't be launched). Requires 'gpg' be installed and in your PATH. See also: G:-----BEGIN
validate-pwd	pwd salt hash -- T (as of 22.03)
	<i>needs crypto/password</i> Validates the hash matches the password and salt as returned by cr:pwd>hash .

Date

Namespace: **d**

Description: Date and time object

word	sed/description
(.time)	n -- s
	<i>needs date/utls</i> Factor of d:time which prints the time to a string
+	d n -- d'
	Add n whole days to the date d . It may be a negative number, in which case the days are subtracted from d . See also: d:- d:=
+day	d1 hr -- d2
	<i>needs date/utls</i> Returns a <i>date</i> a certain number of days from the given <i>date</i> .
+hour	d1 hr -- d2
	<i>needs date/utls</i> Returns a <i>date</i> a certain number of hours from the given <i>date</i> .
+min	d1 min -- d2
	<i>needs date/utls</i> Returns a <i>date</i> a certain number of minutes from the given <i>date</i> .
+msec	d n -- d'
	Advances the date by n milliseconds. n may be negative, in which case the date is decreased. See also: d:new d:msec
-	d1 d2 -- n
	Returns the difference n , in days (and fractions of a day), between the dates. See also: d:+ d:=
.time	n --
	<i>needs date/utls</i> Print the <i>number</i> as HH:MM

word	sed/description
/	d -- a
	<p>Split the date into an array containing, in order: year, month, day, hour, minute, second, tzHH, tzMM (time-zone offset in hours and minutes from GMT), msec, and "approx" (if not zero, how uncertain the date is in days).</p> <p>See also: d:join</p>
=	d1 d2 -- T
	<p>Returns true if the dates are equal. If the dates are "approximate", then they will be considered equal if their approximate regions overlap.</p> <p>See also: d:- d:+ d:?=</p>
>fixed	d -- n
	<p>Returns the "fixed" time value corresponding to the date d .</p> <p>See also: d:fixed></p>
>hmds	x -- s (as of 23.08)
	<p><i>needs date/format</i></p> <p>Convert a millisecond value to a string like "1h 23m 5.678s". If x is a date, converts to msec first. Leaves off leading 0 values, e.g. if less than an hour, or minute.</p> <p>See also: d:>hmds:</p>
>hmds:	x -- s (as of 23.08)
	<p><i>needs date/format</i></p> <p>Convert a millisecond value to a string like "1:23:05.678". If x is a date, converts to msec first. Leaves off leading 0 values, e.g. if less than an hour, or minute.</p> <p>See also: d:>hmds</p>
>msec	d -- n
	<p>Returns the number of milliseconds since Jan 01 1970 corresponding to the date. It takes the TZ of the date into account, normalizing to UTC.</p> <p>See also: d:msec></p>
>unix	d -- n

word	sed/description
	<p>Returns the Unix time-stamp n corresponding to the date d .</p> <p>See also: d:unix></p>
>ymd	d1 -- y m d
	<p><i>needs date/utils</i></p> <p>Returns the year, month and day for a given <i>date</i>.</p>
?=	d1 d2 n -- T (as of 20.07)
	<p><i>needs date/approx</i></p> <p>Return true if the <i>dates</i> d1 and d2 are approximately equal, based on a maximum day-difference of n. Takes into account if the date is approximate as well.</p>
Fri	-- 5
	<p><i>needs date/utils</i></p> <p>Returns the number corresponding to Friday.</p>
Mon	-- 1
	<p><i>needs date/utils</i></p> <p>Returns the number corresponding to Monday.</p>
Sat	-- 6
	<p><i>needs date/utils</i></p> <p>Returns the number corresponding to Saturday.</p>
Sun	-- 0
	<p><i>needs date/utils</i></p> <p>Returns the number corresponding to Sunday.</p>
Thu	-- 4
	<p><i>needs date/utils</i></p> <p>Returns the number corresponding to Thursday.</p>
Tue	-- 2

word	sed/description
	<i>needs date/utls</i> Returns the number corresponding to Tuesday.
Wed	-- 3
	<i>needs date/utls</i> Returns the number corresponding to Wednesday.
adjust-dst	d n -- d
	<i>needs date/dst</i> Adjusts the <i>date</i> d according to the number 0-6 of the dst-zone, n . See d:dst for the meaning of that number.
alarm	d m -- d --
	<i>needs date/alarm</i> Set (if a map on TOS) or remove (if a date on TOS) an "alarm". The alarm will go off at or very close to the time requested (within a second). The map has the key word which is a word, or task which is a task to signal. A word will be invoked in the context of the alarm task, and must do its job quickly to avoid missing other alarms. Its SED is m -- , which is the map passed initially (so other data can be held inside) A task will have that map pushed to its queue, and then be signalled. In the even
approx!	d n -- d (as of 20.07)
	Sets the uncertainty, in days, of the given date, modifying it.
approx?	d -- d n (as of 20.07)
	Returns the uncertainty, in days, of the given date.
approximates!	a -- (as of 20.07)
	<i>needs date/approx</i> Set the <i>regex</i> to use for approximate date matches. Pass an <i>array</i> of <i>strings</i> each of which is indicates "approximate". Be careful to <i>not</i> use text which matches your localized month names.
between	d1 d2 d3 -- T (as of 22.07)
	Like n:between , but for dates. true if d2 ≤ d1 ≤ d3 .
cmp	d1 d2 -- n (as of 22.07)

word	sed/description
	Compare the two dates, similar to n:cmp . If d1 is chronologically before d2 , returns -1; 0 if the same, 1 if after. If either date is "approximate", then the comparison takes that into account and will consider them equal if their approximate regions overlap.
d.	d -- <i>needs date/utils</i> Format and print the given <i>date</i> in "YYYY-MM-DD" format.
default-now	T -- (as of 22.05) If true (the default), makes d:parse and d:join assume "now" as current values for all missing fields; if false , 0 is assumed. See also: d:parse d:join
doy	d -- day-in-year <i>needs date/utils</i> For a given <i>date</i> , return the day in the year (counting Jan 1 as '1').
dst-ofs	x d -- n (as of 23.04) <i>needs date/daylight</i> Given a time-zone name and a date, returns the correct GMT offset in minutes for that date and tz
dst?	d n -- d f <i>needs date/dst</i> Determines if the <i>date</i> d is in DST (Daylight Savings Time) or not, in accordance with the <i>number</i> n of the dst-zone. The values are 0=OFF, 1=ON, 2=USA, 3=EU, 4=Israel, 5=Mexico, 6=Australia.
dstinfo	zone -- arr (as of 23.04) <i>needs date/daylight</i> Queries the 'libs/date/dst.db' database for the DST information for the given <i>string</i> zone (e.g. "America/New_York") or its numeric code (e.g. zones.id) Returns an <i>array</i> consisting of <i>arrays</i> with the "code", "name", "abbrev", "time_start" (in unix timestamp), "gmt_offset", and "dst" indicator.
dstquery	zone start end -- arr (as of 18.06)

word	sed/description
	<p><i>needs date/daylight</i></p> <p>Queries the 'libs/date/dst.db' database for the DST information for the given <i>string zone</i> (e.g. "America/New_York") between the <i>dates</i> start and end. Returns an <i>array</i> consisting of <i>arrays</i> with the "code", "name", "abbrev", "time_start" (in unix timestamp), "gmt_offset", and "dst" indicator.</p>
dstzones?	-- arr
	<p><i>needs date/daylight</i></p> <p>Returns an <i>array</i> of known country-time-zones for DST queries.</p>
elapsed-timer	n -- elapsed
	<p><i>needs utils/elapsed</i></p> <p>Returns the number of ticks since timer n was started.</p>
elapsed-timer-hmds	n T -- s (as of 23.08)
	<p><i>needs utils/elapsed</i></p> <p>Returns a string like "10h 23m 15.234s" if T is true , otherwise "10:23:15.234".</p>
elapsed-timer-msec	n -- msec (as of 23.08)
	<p><i>needs utils/elapsed</i></p> <p>Returns the number of milliseconds since timer n was started.</p>
elapsed-timer-seconds	n -- secs
	<p><i>needs utils/elapsed</i></p> <p>Returns the number of seconds since timer n was started.</p>
first-dow	d1 dow -- d2
	<p><i>needs date/utils</i></p> <p>Returns a <i>date</i> for the first day-of-week in the month of the given <i>date</i>.</p>
fixed>	n -- d

word	sed/description
	<p>Returns a new date corresponding to the number which is a "fixed" time value. The range of valid n is 0..3652059, corresponding to 1-1-1 to 9999-12-31.</p> <p>See also: d:>fixed</p>
fixed>dow	n -- n
	<p><i>needs date/utls</i></p> <p>Returns the day-of-week (Sunday is 0) for a given <i>date</i>.</p>
format	s d -- s' d s -- s'

word	sed/description																																												
	<p>Format a date according to the format string s . If a "%" character appears, the following character is a format-specifier; otherwise, it is output directly.</p> <p>Valid format specifiers are:</p> <table><tr><th>fmt</th><th>description</th></tr><tr><td>d</td><td>day</td></tr><tr><td>D</td><td>two-digit day</td></tr><tr><td>H</td><td>2-digit hour</td></tr><tr><td>h</td><td>hour</td></tr><tr><td>m</td><td>month</td></tr><tr><td>M</td><td>two-digit month</td></tr><tr><td>N</td><td>long month name</td></tr><tr><td>n</td><td>short month name</td></tr><tr><td>P</td><td>2-digit hour, am/pm</td></tr><tr><td>p</td><td>hour, am/pm</td></tr><tr><td>S</td><td>2-digit second</td></tr><tr><td>s</td><td>second</td></tr><tr><td>T</td><td>2-digit minute</td></tr><tr><td>t</td><td>minute</td></tr><tr><td>W</td><td>long day name</td></tr><tr><td>w</td><td>short day name</td></tr><tr><td>x</td><td>3 digit milliseconds</td></tr><tr><td>Y</td><td>four digit year</td></tr><tr><td>y</td><td>two digit year</td></tr><tr><td>z</td><td>time-zone offset (HHMM)</td></tr><tr><td>Z</td><td>time-zone offset (HH:MM)</td></tr></table>	fmt	description	d	day	D	two-digit day	H	2-digit hour	h	hour	m	month	M	two-digit month	N	long month name	n	short month name	P	2-digit hour, am/pm	p	hour, am/pm	S	2-digit second	s	second	T	2-digit minute	t	minute	W	long day name	w	short day name	x	3 digit milliseconds	Y	four digit year	y	two digit year	z	time-zone offset (HHMM)	Z	time-zone offset (HH:MM)
fmt	description																																												
d	day																																												
D	two-digit day																																												
H	2-digit hour																																												
h	hour																																												
m	month																																												
M	two-digit month																																												
N	long month name																																												
n	short month name																																												
P	2-digit hour, am/pm																																												
p	hour, am/pm																																												
S	2-digit second																																												
s	second																																												
T	2-digit minute																																												
t	minute																																												
W	long day name																																												
w	short day name																																												
x	3 digit milliseconds																																												
Y	four digit year																																												
y	two digit year																																												
z	time-zone offset (HHMM)																																												
Z	time-zone offset (HH:MM)																																												
join	a -- d																																												
	<p>Inverse of d:/ : takes an array of numbers with year, month, day, hour, minute, second, tzHH, tzMM, msec, and approx, and returns a date.</p> <p>If the array is shorter than 10 items, the missing items will be given default values from the current date and time.</p> <p>See also: d:/ d:parse</p>																																												
last-dow	d1 dow -- d2																																												

word	sed/description
	<i>needs date/utls</i> Returns a <i>date</i> for the last day-of-week in the month of the given <i>date</i> .
last-month	d1 -- d2
	<i>needs date/utls</i> Returns a <i>date</i> one month before the given <i>date</i> .
last-week	d1 -- d2
	<i>needs date/utls</i> Returns a <i>date</i> one week before the given <i>date</i> .
last-year	d1 -- d2
	<i>needs date/utls</i> Returns a <i>date</i> one year before the given <i>date</i> .
leap?	d -- d T (as of 24.03)
	Returns true if the date is in a leap-year.
mdays	d -- d n (as of 24.03)
	Returns the number of days in the date's month.
msec	-- n
	Returns the current time as number of milliseconds since Jan 01, 1970. See also: d:ticks d:new
msec>	n -- d
	Returns a new date corresponding to the number of milliseconds since Jan 01 1970. It takes the current notion of timezone into account (that can be set using d:updatetz). See also: d:>msec
new	-- d

word	sed/description
	<p>Create a new date d , initialized with the current time.</p> <p>See also: d:ticks d:msec</p>
next-dow	d1 dow -- d2
	<p><i>needs date/utls</i></p> <p>Returns a <i>date</i> for the corresponding day-of-week on or after the given <i>date</i>.</p>
next-month	d1 -- d2
	<p><i>needs date/utls</i></p> <p>Returns a <i>date</i> one month after the given <i>date</i>.</p>
next-week	d1 -- d2
	<p><i>needs date/utls</i></p> <p>Returns a <i>date</i> one week after the given <i>date</i>.</p>
next-year	d1 -- d2
	<p><i>needs date/utls</i></p> <p>Returns a <i>date</i> one year after the given <i>date</i>.</p>
parse	s -- d
	<p>Attempt to parse a valid date and/or time from a string, and return the date appropriately set if successful, or null otherwise.</p> <p>The string should be in one of the ISO-8601 formats or the RFC5322 format.</p> <p>It may also be "UNK" or "UNKNOWN", which returns a date which is "unknown"; "NOW", "TODAY", or "CURRENT", which return the current date; or something like "Nov 1896", in which case it will return an approximate date with the month of November, the year of 1896, and whatever the current day of the month is.</p> <p>The year must be less than 100 or greater than 1000; to create a date with a year of 150 (for example), use d:join .</p> <p>See also: d:join</p>
parse-approx	s -- d null (as of 20.07)

word	sed/description
	<p><i>needs date/approx</i></p> <p>Same as d:parse, but understands meaning of various "approximate" indicators like "ABT" or "CA". The list of words which is understood can be set using d::approximates!</p>
parse-range	s -- a null (as of 20.07)
	<p><i>needs date/range</i></p> <p>Parse a "date range", e.g. "BET 2001 AND 2005"</p>
prev-dow	d1 dow -- d2
	<p><i>needs date/utls</i></p> <p>Returns a <i>date</i> for the corresponding day-of-week on or before the given <i>date</i>.</p>
rfc5322	d -- s (as of 23.04)
	<p><i>needs date/rfc</i></p> <p>Format a date as per RFC-5322</p>
start-timer	n --
	<p><i>needs utls/elapsed</i></p> <p>Starts a "high-resolution timer" accessed by the <i>number</i> n. Use d:elapsed-timer or d:elapsed-timer-seconds to read</p>
ticks	-- n (as of 18.05)
	<p>Returns a number of "ticks", which is a "high-resolution" timing counter. The specific resolution can be retrieved using d:ticks/sec .</p> <p>See also: d:ticks/sec d:msec</p>
ticks/sec	-- n (as of 18.05)
	<p>Returns the resolution in "ticks per second" of the "high-resolution" timer.</p> <p>See also: d:ticks d:msec</p>
timer	m -- X (as of 18.06)
	Establishes a timer running on a separate task, in accordance with the options in the map. Keys are:
timer-ctrl	X m -- (as of 20.03)

word	sed/description
	<p>Controls a timer returned from <code>d:timer</code> . The map keys may be "repeat", the number of seconds to repeat; or "stop" which stops the timer if <code>true</code> .</p> <p>See also: d:timer</p>
tzadjust	d n -- d' (as of 18.05)
	<p>Applies a "time zone adjustment" of <code>n</code> minutes offset from GMT to the date, returning a new date with the given time-zone offset. Ex: convert current time to GMT: <code>d:new 0 d:tzadjust</code></p> <p>Or perhaps: <code>"2019-06-23T12:00:00+02:00" d:parse 0 d:tzadjust .</code> resulting in <code>2019-06-23T10:00:00+00:00</code> .</p>
unix>	n -- d
	<p>Returns a new date corresponding to the Unix time-stamp <code>n</code> .</p> <p>See also: d:>unix</p>
unknown	-- d (as of 20.07)
	Returns the singleton "unknown" date.
unknown?	d -- d T (as of 20.07)
	Returns <code>true</code> if the date is "unknown".
updatetz	n -- (as of 18.06)
	Sets the current time-zone offset in minutes from GMT. If <code>n</code> is <code>null</code> , refreshes the notion of the system's time-zone programmatically.
year@	d -- d y
	<p><i>needs date/utls</i></p> <p>Returns the year from the given <i>date</i> <code>d</code>.</p>
ymd	d -- d s
	<p><i>needs date/utls</i></p> <p>Format a given date as YYYY-MM-DD, returning a string.</p>
ymd>	y m d -- d1

word	sed/description
	<i>needs date/utils</i> Returns a <i>date</i> (at midnight) for a given year, month and day.

DB

Namespace: **db**

Description: SQLite database

word	sed/description
SQL!	param ! -- (as of 23.02)
	IMMEDIATE <i>needs db/sql</i> Invokes the SQL query without returning values.
SQL[param] -- a (as of 23.02)
	IMMEDIATE <i>needs db/sql</i> Returns an array (possibly empty) of the results of the SQL query. The rows are returned as an array of the column values.
SQL{	param } -- a (as of 23.02)
	IMMEDIATE <i>needs db/sql</i> Returns an array (possibly empty) of the results of the SQL query. The rows are returned as a map of named column, value pairs.
add-func	db m -- db T (as of 19.01)

word	sed/description																												
	<p>Adds a new SQLite function implemented in 8th to the open database. The map may contain the following keys:</p> <table><tr><th>key</th><th>type</th><th>description</th><th>default</th></tr><tr><td>func</td><td>w</td><td>an word, required if the function is a "scalar"</td><td></td></tr><tr><td>inverse, value</td><td>w</td><td>words, required if the function is an "window"</td><td></td></tr><tr><td>name</td><td>s</td><td>required: the name of the function to be added</td><td></td></tr><tr><td>nparams</td><td>n</td><td>the number of parameters the function will take;</td><td>-1</td></tr><tr><td>step, final</td><td>w</td><td>words, required if the function is an "aggregate"</td><td></td></tr><tr><td>window</td><td>T</td><td>if true, declaring a "window function"</td><td>false</td></tr></table> <p>Consult the SQLite documentation for definitions and semantics of "window", "aggregate", and "scalar".</p> <p>Returns true if the addition was successful. The "func" word receives an array of parameters from the SQL call. It is its responsibility to ensure the parameters are the correct type for whatever it's going to do. Whatever it leaves on TOS is the function result, but only number, string, buffer, and null are valid SQLite types. Anything else should be converted to a string first. More details in the manual.</p>	key	type	description	default	func	w	an word, required if the function is a "scalar"		inverse, value	w	words, required if the function is an "window"		name	s	required: the name of the function to be added		nparams	n	the number of parameters the function will take;	-1	step, final	w	words, required if the function is an "aggregate"		window	T	if true, declaring a "window function"	false
key	type	description	default																										
func	w	an word, required if the function is a "scalar"																											
inverse, value	w	words, required if the function is an "window"																											
name	s	required: the name of the function to be added																											
nparams	n	the number of parameters the function will take;	-1																										
step, final	w	words, required if the function is an "aggregate"																											
window	T	if true, declaring a "window function"	false																										
aes!	T -- (as of 20.04)																												
	<p>If true , sets the SQLite cipher used for encrypted databases to AES+256+GCM. The default is Chacha20Poly1305, which is set if T is false .</p>																												
again?	db -- db T (as of 23.05)																												
	<p>(SQLite only) Returns true if the last action on the database returned a "busy" or "locked" status.</p>																												
begin	db -- db (as of 20.04)																												
	<p>Equivalent of issuing the SQL "BEGIN TRANSACTION" for the database.</p>																												
bind	sql x n -- sql sql a -- sql sql m -- sql																												
	<p>Bind parameters to the prepared SQL query in preparation for db:exec or db:exec-cb . The alternatives are:</p> <ul style="list-style-type: none">• x is an item to bind to parameter number n• a is an array of <i>items</i> to bind, in order the appear in the SQL statement• m is a map whose <i>keys</i> correspond to <i>named</i> parameters																												
bind-exec	db s x w -- db (as of 19.06)																												

word	sed/description
	<p>Combines bind and exec for a named sql. The s parameter is the name of a sql created previously with db:prep-name. x is an array or map of the parameters to bind to the sql. w is either null, or a word to invoke with each SQL result row as for db:exec-cb.</p>
bind-exec[]	<p>db s a -- db a' (as of 19.07)</p>
	<p>Like db:bind-exec, but returns an array of results from the SELECT statement. The results contain one array for each row of the result, as db:col[] is invoked on each row. An empty array is returned if no results were available, null is returned if there was an error.</p> <p>See also: db:bind-exec</p>
bind-exec{}	<p>db s a -- db a' (as of 23.04)</p>
	<p>Same as db:bind-exec[], but returns one map for each row as db:col{} is invoked for each.</p> <p>See also: db:bind-exec</p>
close	<p>db --</p>
	<p>Close the database. It may not be read from or written to after this.</p> <p>See also: db:open</p>
col	<p>sql n -- sql x</p>
	<p>Used <i>only</i> inside a db:exec-cb callback! Returns the value of the query column n from the sql.</p> <p>See also: db:exec-cb</p>
col[]	<p>sql -- sql a</p>
	<p>May be used <i>only</i> inside a db:exec-cb callback; retrieves all the columns of the query sql into an array, in the order they appear in the query.</p> <p>See also: db:exec-cb db:col{}</p>
col{}	<p>sql -- sql m</p>

word	sed/description
	<p>May be used <i>only</i> inside a db:exec-cb callback; retrieves all the columns of the query into a map, where the keys are the column names.</p> <p>See also: db:exec-cb db:col[]</p>
commit	db -- db T (as of 20.04)
	Commits the transaction opened with db:begin , and returns true if the commit succeeded.
db	-- db (as of 23.02)
	<p><i>needs db/sql</i></p> <p>Returns the current database.</p>
dbpush	s -- n (as of 23.02)
	<p><i>needs db/sql</i></p> <p>Opens the database file based on the string or map it's given. The opened database is pushed onto the list of open databases, and that database is accessible by number with db:use . The first database pushed is the default one, until db:use is invoked.</p>
disuse	n -- (as of 23.02)
	<p><i>needs db/sql</i></p> <p>Closes the database #n and makes it unavailable.</p>
each	db s key fwd w -- db (as of 20.04)
	<p>Iterates over the keys in a KV database. The starting point is the key key (if null , starts from the beginning). s is the name of the database, as in the get/set words. fwd is true if the iteration order should be "forward", otherwise it's "backward". The word w is invoked with the SED: k v -- , and it can stop iteration by invoking break .</p> <p>Note: the callback w must not change the database in any way!</p>
err-handler	w -- (as of 23.02)
	<p><i>needs db/sql</i></p> <p>Sets the error handler for the SQL words. If a SQL error occurs, it will be invoked with a SED of s -- . The default handler throws the string it's passed.</p>
exec	db sql -- db db s -- db

word	sed/description
	<p>Execute the prepared statement sql , or the SQL string s on the database, running until finished.</p> <p>See also: db:open db:prepare db:exec-cb</p>
exec-cb	db w sql -- db db w s -- db
	<p>Like db:exec , but invokes the word w as a callback for each row retrieved. The SED for w is sql -- . Use db:col within the callback to retrieve the column data. Note: you <i>must</i> adhere to the callback SED!</p> <p>See also: db:open db:prepare db:exec db:col</p>
exec-name	db w s -- db (as of 19.06)
	<p>Same as db:exec-cb , but uses the sql stored under the name s in the db . If w is null , acts the same as db:exec , using the index or named sql. Note: you must adhere to the callback SED as per db:exec-cb !</p> <p>See also: db:exec-cb db:prep-name db:bind-exec</p>
exec[]	db s -- db a (as of 23.04)
	<p>Like db:bind-exec[] , but takes a SQL SELECT string (or prepared statement) without parameters.</p> <p>See also: db:exec{}</p>
exec{}	db s -- db a (as of 23.04)
	<p>Like db:bind-exec{} , but takes a SQL SELECT string (or prepared statement) without parameters.</p> <p>See also: db:exec[]</p>
get	db key -- db x (as of 20.04)
	<p>Get the value associated with the give key in the "kv" database. The key may be anything, but a string is preferred. It must have the same value as what was used with db:set . The value is converted back from mpack format into whatever it originally was.</p>
get-sub	db s key -- db x (as of 20.04)
	<p>Same as db:get , but gets from the named sub-database.</p>
key	db b -- db

word	sed/description
	<p>Tells the SQLite engine it should use that key as the encryption key for the database. The database <i>must</i> have been created encrypted in order for this to work, and the key must <i>exactly</i> match the one used originally. The buffer given must be 32-bytes long, most likely created with cr:genkey from a password string entered by the user. Alternatively, you may pass the key in the 'key' key of the map given to db:open .</p> <p>The key used can be changed using db:rekey .</p> <p>See also: db:rekey</p>
kind?	db -- db s (as of 19.05)
	Returns one of "SQLITE", "MYSQL", "ODBC", "SQLITE ENC", or "KV" for the open db.
last-rowid	db -- db n (as of 19.05)
	Gets the last row-id inserted (specific to SQLite databases).
mysql?	-- T
	<p>Returns true if MySQL support is currently available. If true , you may use the db words to access MySQL databases. This support is provided by a third-party library which <i>must</i> be installed separately from 8th, from here: https://dev.mysql.com/downloads/connector/c/ .</p> <p>MySQL support is <i>only</i> available for Pro+, so this word will always report false on less capable SKUs.</p> <p>See also: db:odbc?</p>
odbc?	-- T
	<p>Returns true if ODBC support is currently available, false otherwise. If true , you may use the db words to access an ODBC database.</p> <p>On Linux/RPI you need to have installed unixodbc or iodbc. On macOS you need to have configured an ODBC source. On mobile devices there is no ODBC support at present.</p> <p>ODBC support is <i>only</i> available on Pro+, so this word will always report false on less capable SKUs.</p> <p>See also: db:mysql?</p>
open	s -- db b -- db m -- db null -- db

Open the SQLite (Pro+: MySQL/ODBC/KV) database indicated by the string, buffer, or map. If passed a:

type	description
string	the name of a file to open (or create, if it doesn't exist) as a SQLite database
buffer	assumed to be an entire SQLite database as a buffer. In that case, it is treated as a read-only database
null	creates an in-memory SQLite database

If a map is passed, uses these keys to create a SQLite database (possibly encrypted):

key	description
create	Create regardless of existence
file	Required: name of db file to create
key	Optional: the encryption key (as a buffer)
kind	"enc" (encrypted SQLite) or "sqlite" (plain SQLite)
ro	if true, the database is read-only
limits	For "enc" or "sqlite", set limits according the map below

Limits are some of:

key	value	default	max
length	SQLITE_MAX_LENGTH	1G	1G
sql_length	SQLITE_MAX_SQL_LENGTH	10M	1G
column	SQLITE_MAX_COLUMN	64	2000
expr_depth	SQLITE_MAX_EXPR_DEPTH	256	1000
compound_select	SQLITE_MAX_COMPOUND_SELECT	16	500
vdbe_op	SQLITE_MAX_VDBE_OP	250M	250M
func_arg	SQLITE_MAX_FUNCTION_ARG	127	127
attached	SQLITE_MAX_ATTACHED	10	10
pattern_length	SQLITE_MAX_LIKE_PATTERN_LENGTH	50K	50K
variable_number	SQLITE_MAX_VARIABLE_NUMBER	32766	32766
trigger_depth	SQLITE_MAX_TRIGGER_DEPTH	1000	1000
worker_threads	SQLITE_MAX_WORKER_THREADS	8	1000

Pro+: creates a MySQL, ODBC, or KV database using the map keys below:

key	value	DB	default
charset	the character set to use	MySQL	utf8mb4
create	if true, create the file or dir if it doesn't exist	KV	false
db	database to access	MySQL	
dir	if false, the 'file' option is a file name not a directory	KV	true

word	sed/description			
	key	value	DB	default
	dsn	The normal DSN value for the ODBC connection	ODBC	
	fetch-local	if false, don't fetch all the result locally	MySQL	true
	file	the file or directory where the KV database is	KV	
	host	IP address or DNS name of server	MySQL	127.0.0.1
	kind	"mysql", "odbc", or "kv"	all	
	lock	if false, don't do any locking	KV	true
	map	if true, write using mmap	KV	false
	mapsize	if not -1, set the max db size (10485760 default)	KV	-1
	max-buf	maximum size of buffer (BLOB) to retrieve	MySQL	10000000
	max-str	maximum size of string to retrieve	MySQL	100000
	maxdbs	if not 0, allow that many sub databases	KV	0
	meminit	if false, don't ensure memory is zeroed	KV	true
	mode	the permissions for the created database	KV	0664
	msync	if false, don't fsync metapage after commit	KV	true
	port	port on server to access	MySQL	3305
	pwd	password for user to access server	MySQL	
	rdahead	if false, don't do readahead	KV	true
	ro	if true, the database is read-only	KV	false
	sock	string with a UNIX socket or pipe	MySQL	0
	sync	if false, don't flush to disk for each transaction	KV	true
	user	user name to access server	MySQL	
<p>Returns null if it was unable to open the database for some reason; otherwise returns a valid db.</p> <p>See also: db:close</p>				
open?	filename -- db flag			
	<p><i>needs db/open</i></p> <p>Open a presumed SQLite file. Returns 0 if the file is invalid, 1 if it is empty, and 2 if it has been initialized</p>			
prep-name	db sql s -- db (as of 19.06)			
	<p>Same effect as db:prepare , but stores the query under the given name s in the db item. The stored sql will be null if there was an error creating it.</p> <p>See also: db:prepare db:exec-name db:sql@ db:bind-exec</p>			

word	sed/description
prepare	db s -- db sql
	<p>Prepare a SQL query from the string s , for execution against that database. Returns a sql, or null if there was an error.</p> <p>See also: db:open db:exec db:exec-cb db:bind</p>
query	db qry -- result
	<p><i>needs db/query</i></p> <p>Execute a query on the <i>database</i> db, and return the first row, as an <i>array</i> of items.</p>
query-all	db qry -- result
	<p><i>needs db/query</i></p> <p>Execute a query on the <i>database</i> db, and return all rows, as an <i>array of arrays</i> of items.</p>
rekey	db b -- db
	<p>Tell the SQLite engine it should use the new key as the encryption key for the database. If the database was not encrypted, it will be from now on; otherwise, it will be re-encrypted with the new key.</p> <p><i>Note:</i> a system failure (power outage etc) during the rekeying will likely corrupt the database and render it unusable. So operate on a <i>copy</i> of the database when rekeying!</p> <p>See also: db:key</p>
rollback	db -- db (as of 20.04)
	Rolls back the transaction started with db:begin .
set	db key x -- db (as of 20.04)
	Puts a key-value pair into the database opened as type "kv". The key may be anything, but a string is preferred; the value can be anything. It is converted to a buffer using G:mpack before storing. Use db:get to retrieve the stored value. If null is stored, it deletes the (k,v) pair from the database. Thus, null cannot be stored (though the string "null" can).
set-sub	db s key x -- db (as of 20.04)
	Same as db:set , but sets to the named sub-database.
sql@	db s -- db sql (as of 19.06)

word	sed/description
	Retrieve the sql indicated by the string. The return value may be null , if db:prep-name failed or if no sql exists by the given name or index.
sql[]	params s -- a (as of 23.02)
	<i>needs db/sql</i> Same as SQL[] , but not "immediate-mode", so the SQL can be built at runtime.
sql[np]	s -- a (as of 23.02)
	<i>needs db/sql</i> Same as sql[] but takes no parameters for the SQL
sql{np}	s -- a (as of 23.02)
	<i>needs db/sql</i> Same as sql{ } but takes no parameters for the SQL
sql{ }	params s -- a (as of 23.02)
	<i>needs db/sql</i> Same as SQL{ } , but not "immediate-mode", so the SQL can be built at runtime.
use	n -- (as of 23.02)
	<i>needs db/sql</i> Makes the database # n the current one to access. Only necessary if more than one db:push has been invoked.
zip	db -- db (as of 23.04)
	<i>needs db/zip</i> Add compress() and expand() SQL functions to the open database

DBUS

Namespace: **DBUS**

Description: D-BUS Interface

word	sed/description																					
call	DBUS m -- DBUS s (as of 21.09)																					
	<p><i>Professional version</i></p> <p>Linux/RPI only: call the D-Bus interface as specified in the map, returns a string result The map keys are:</p> <table><tr><th>key</th><th>type</th><th>description</th></tr><tr><td>name</td><td>s</td><td>DBUS bus name</td></tr><tr><td>path</td><td>s</td><td>DBUS path</td></tr><tr><td>iface</td><td>s</td><td>DBUS interface</td></tr><tr><td>method</td><td>s</td><td>DBUS method</td></tr><tr><td>to</td><td>n</td><td>timeout in msec (default: -1)</td></tr><tr><td>args</td><td>a</td><td>array of arguments</td></tr></table> <p>Note: the args array must have as its first element a string containing the D-Bus signature of each following argument. Follow the D-Bus documentation for details on what that means.</p>	key	type	description	name	s	DBUS bus name	path	s	DBUS path	iface	s	DBUS interface	method	s	DBUS method	to	n	timeout in msec (default: -1)	args	a	array of arguments
key	type	description																				
name	s	DBUS bus name																				
path	s	DBUS path																				
iface	s	DBUS interface																				
method	s	DBUS method																				
to	n	timeout in msec (default: -1)																				
args	a	array of arguments																				
init	n -- DBUS (as of 21.09)																					
	<p><i>Professional version</i></p> <p>Linux/RPI only: connect to the D-Bus system. n is 0 for the login-session bus, 1 for the system bus, and 2 for the bus that started us, if any.</p>																					

Debug

Namespace: **dbg**

Description: Debugging words

word	sed/description
.state	-- (as of 24.06)
	Prints the current task's state (for debugging only).
bp	w --
	<p>Set a breakpoint at the word w. This is a "one-shot" breakpoint, and gets reset once you do dbg:go.</p> <p>See also: dbg:go dbg:stop</p>
bt	-- (as of 23.01)

word	sed/description
	<p><i>needs debug/nicer</i></p> <p>Once dbg:stop puts you into the debug mode, invoking dbg:bt will show the backtrace of how you got where you are.</p>
except-task@	-- t (as of 18.04)
	<p>Returns the task from which the last exception was thrown, or null if there wasn't one. Resets the last-exception task to null .</p> <p>See also: G:throw</p>
go	--
	<p>Continue operation after dbg:stop or a breakpoint from dbg:bp .</p> <p>See also: dbg:stop dbg:bp</p>
prompt	s -- s'
	<p>DEFERRED</p> <p>Same as G:prompt but in "debug" mode.</p> <p>See also: G:prompt</p>
see	w -- (as of 24.06)
	Display the contents of an 8th word. Useful for debugging compilation issues.
stop	--
	<p>Stops and enters an interpreter waiting for commands. Use dbg:go to continue.</p> <p>See also: dbg:go dbg:bp</p>
trace	T --
	<p>Turns "tracing" on or off. If on, new words will include invocations of the words set by dbg:trace-enter and dbg:trace-leave .</p> <p>See also: dbg:trace-leave dbg:trace-enter</p>
trace-enter	w --

word	sed/description
	<p><i>needs debug/common</i></p> <p>If dbg:trace is on, invoke w upon entry to any compiled <i>word</i>. If w is null, turns off the trace. The <i>word</i> must consume TOS. The SED of w is w -- .</p> <p>See also: dbg:trace dbg:trace-leave</p>
trace-leave	w --
	<p><i>needs debug/common</i></p> <p>If dbg:trace is on, invoke w upon exit from any compiled <i>word</i>. If null, turns off the trace. The <i>word</i> must consume TOS. The SED of w is w -- .</p> <p>See also: dbg:trace dbg:trace-enter</p>

DOM

Namespace: **DOM**

Description: DOM (Document Object Model)

word	sed/description
+	DOM1 DOM2 -- DOM1 (as of 21.08)
	Adds DOM2 as a child of DOM1 , and assigns DOM1 as the parent of DOM2 .
-	DOM1 DOM2 -- DOM1 (as of 21.08)
	Removes DOM2 from the children of DOM1 , making DOM2 parentless.
attr!	DOM s x -- DOM DOM m -- DOM (as of 21.08)
	Sets the attribute with that name in the DOM. If a map is given, it is added to the DOM attributes (per key,val).
attr@	DOM s -- DOM x (as of 21.08)
	Gets the attribute with that name from the DOM. Returns null if the attribute doesn't exist.
attrs	DOM -- DOM m (as of 21.09)

word	sed/description
	Returns the "attributes" of the DOM node, which may be null.
children	DOM -- DOM a (as of 21.08)
	Gets an array (perhaps empty) containing the child nodes of the DOM.
css-parse	sb -- m (as of 21.09)
	Takes a string or buffer containing CSS, and parses it into a map which can be used by DOM:css-apply . Only understands a subset of CSS3 at the moment. Returns null on error.
each	DOM w n -- DOM (as of 21.08)
	<p>Iterates the DOM, passing the word w each element in turn. Iteration is stopped with G:break, and will not go farther down the DOM than 'n' nodes.</p> <p>The SED of w is DOM n --, where 'n' is the current depth.</p>
find	dom w depth -- dom a (as of 21.08)
	<p><i>needs dom/find</i></p> <p>Similar to DOM:each, but the SED of w is dom n -- T, where true means keep this item, and 'n' is the depth of the item from the source DOM. The DOM nodes found are returned in an array.</p>
new	s DOM -- DOM' (as of 21.08)
	Create a new DOM node, with the string "type", and DOM "parent" -- either or both may be null.
type	DOM -- DOM s (as of 21.09)
	Returns the "type" of the DOM node, which may be null.

File

Namespace: **f**

Description: File operations

word	sed/description
/	s -- a (as of 21.02)
	<p>Separates the "path" s into an array [drive,path,filename,extension,separator] . Inverse of f:join .</p> <p>See also: f:join f:relpath f:abspath</p>
>posix	s -- s' (as of 23.03)
	<p>Converts any Windows backslash \ characters in the string to POSIX forward / slashes.</p>
abspath	f -- s s -- s'
	<p>Given a file or string containing a file name, returns a string containing the absolute path of the original file name.</p> <p>See also: f:relpath</p>
absrel	x rel -- abs (as of 21.02)
	<p><i>needs file/absrel</i></p> <p>Like f:abspath, but assumes x is relative to the path given by rel</p>
append	f -- f (as of 17.02)
	<p>Seek to the end of the file in preparation for appending to it.</p> <p>See also: f:write</p>
associate	ext desc longdesc app --
	<p>"Associate" the application named app as the default for opening the file extension ext , with, a short description of desc and a long description of longdesc . The short description <i>must not</i> have spaces in it.</p> <p><i>Note: only on Windows currently</i></p>
atime	f -- f d (as of 16.05)
	<p>Returns a time-stamp of when the file was last accessed.</p>
autodel	f -- f (as of 22.03)
	<p>Makes the file automatically delete itself when the program ends, or when the its refcount goes to 0.</p>

word	sed/description
canwrite?	s -- T
	Returns true if it is possible to write to the named file, or false otherwise. If the file <i>does not exist</i> , it determines whether or not the parent directory is writeable.
chmod	f n -- f s -- s n -- s s' --
	<p>Change the access "mode" of the file to that indicated in the string or number. If the mode is a string, it can contain "r" = "readable", "w" = "writeable", "x" = "executable". By default, it only affects "owner-modes". To affect "group" modes, use 'g'; to affect 'other' modes, use 'o'; to affect all, use 'a'.</p> <p>Ex: "o+x" will add execute permissions for 'others'. In order to apply to more than one you issue two separate stanzas, e.g. "o+xg+x". If a "+" is included then the modes are added to the existing one, if "-" they are removed, otherwise they are set exactly.</p> <p>The file given may be an open file or a string with the name of a file. If the mode is a number, then that OS-specific mode will be set exactly without interpretation by 8th.</p> <p>See also: f:getmod</p>
close	f --
	<p>Close the file. I/O to that file will not be possible if, for example, a var stored f .</p> <p>See also: f:open-ro f:create f:open</p>
copy	src dest -- T
	<p>Try to copy the file named src to a new file named dest , which will be <i>removed</i> if it already exists. If dest is a directory, an appropriately named file will be created there. If the copy succeeds, true will be on TOS, otherwise false .</p> <p>See also: f:copydir</p>
copydir	src dest -- T
	<p>Try to recursively copy the directory named src to a new one named, dest . If the copy succeeds, true will be on TOS, otherwise false .</p>
create	s -- f
	<p>Create the file named by the string s , returning the file f .</p> <p><i>Note:</i> an existing file by the same name <i>will be destroyed</i>. If that is not your intention, use f:open instead.</p>

word	sed/description
ctime	f -- f d
	Returns the 'creation' time of the file or string naming a file, as a date. See also: f:mtime f:atime
dir?	s -- T
	Returns true if the s represents a directory.
dname	s -- s'
	Returns the path of the filename, excluding the file name as returned by f:fname . See also: f:fname f:abspath f:relpath
eachbuf	f w n -- f s w n --
	Similar to f:eachline , but passes a buffer of up to n bytes to the word w , whose SED is b -- . Responds to break . <i>NOTE:</i> the same actual buffer is passed to w each time. See also: f:getline s:eachline f:eachline
eachline	f w -- f s w --
	Similar to s:eachline , but leaves the file on TOS when done. The SED of w is s -- . Responds to break . If a string is passed rather than a file, it is assumed to be the name of a file, and the string is not left on TOS after running. If the file cannot be read (or opened, if a string), t:err? will report an error. <i>NOTE:</i> the <i>same</i> actual string is passed to w each time, so if you want to store the data you must clone it (or use const). See also: f:getline s:eachline f:eachbuf
enssep	s -- s'
	Ensure that the file name given by the string ends with the OS-appropriate directory-separator. Use this instead of hard-coding "\" or "/".
eof?	f -- f T (as of 18.03)

word	sed/description
	Returns true if the file is at "end of file".
exec	s -- (as of 23.01)
	Attempts to execute the file as 8th code, unpacking and decrypting it if necessary. Throws on failure.
exists?	s -- T
	Returns true if the s represents a file or directory which exists.
flush	f -- f
	Make sure any buffers containing unwritten data in the file are written.
fname	s -- s'
	Returns the last part (exclusive of path) of the filename. See also: f:dname f:abspath f:relpath
getb	f -- f b
	<i>needs file/getc</i> Read a single byte from the given <i>file</i> . If no byte is available, null is returned.
getc	f -- f c
	<i>needs file/getc</i> Read a single Unicode character from the given <i>file</i> . If no character is available, null is returned.
getline	f -- f s
	Read one line from the file f . "line" means up to but not including a newline or carriage-return character ("\n" or "\r"). Returns null if no data are available (or trying to read past end-of-file). See also: f:eachline
getmod	f -- n s -- n

word	sed/description
	<p>Returns the OS-specific numeric "mode" of the file. See your OS documentation for the specific interpretation of the "mode". It may be given an open file or a string representing the name of a file.</p> <p>See also: f:chmod</p>
glob	s -- a
	<p>Returns an array of file names matching the string pattern s . The "glob" pattern is <i>not</i> a regex but rather the usual "*" and "?" file matching.</p> <p>See also: f:rglob</p>
glob-links	T -- (as of 22.01)
	<p>If true , then f:rglob will descend into directories which are symbolic links. The default is to not do so.</p>
glob-nocase	T --
	<p>If T is true , then make the f:glob word case-insensitive; otherwise it is case-sensitive.</p> <p>The default for this setting is true on Windows and macOS, and false on all other systems.</p> <p>See also: f:glob f:rglob</p>
gunz	readcb writecb -- n (as of 22.04)
	<p>"gzip -d" equivalent. Takes a read callback word to get buffers of data to decompress, and a write callback to write the decompressed data. Returns 0 on success, or a 'miniz' error code.</p> <p>The SED for readcb is -- b ; if no more data is available, returns null . The SED for writecb is b -- .</p>
homedir	-- s (as of 19.08)
	<p>Returns the home-directory for the current user</p>
homedir!	s -- (as of 21.04)
	<p>Sets the value returned by f:homedir . Useful if running as 'root' but not installed in the root directory.</p> <p>See also: f:homedir</p>
include	s --

word	sed/description
	<p>Read the named file and interpret it as 8th code. Looks for the file in this order:</p> <ol style="list-style-type: none"> 1. the file name as given, then 2. that name in the "incs" <i>asset</i>, then 3. in the app:datadir , then 4. in the directory pointed to by the EIGHTHLIB environment variable (if any), then finally 5. in app:8thdir . <p>Throws an exception if it cannot load the file.</p> <p>See also: G:needs f:slurp app:asset G:requires</p>
ioctl	f req x -- f x' n f null (as of 20.06)
	<p><i>Hobbyist version</i></p> <p>(NOT Windows) Issues an ioctl system call on the given file or net (socket). The parameter req is a number indicating which ioctl call is to be performed, and x is either an integer number, a string, or a buffer, depending on the IOCTL call.</p> <p>Returns a number and the value x' is of the same type as x .</p> <p><i>Note:</i> This is OS and ioctl-call specific. Use with caution!</p> <p>If f isn't a file or a net, or if x is an invalid type, returns null. On ioctl error, sets t:err? return value.</p>
join	a -- s (as of 21.02)
	<p>Joins the array containing strings [drive,path,filename,extension,separator] into a file name. Inverse of f:/ . Returns null if any of the components is not a string or is missing.</p> <p>See also: f:/ f:relpath f:abspath</p>
launch	s params --
	<p>Launches the named file. If it is executable, executes it; if it is a document, starts the appropriate application to open it; if it is a folder, displays it in the OS-specific folder viewer.</p> <p>Give the parameter string params (may be null) if needed.</p> <p>On mobile devices this is the same as passing s to net:launch.</p>
link	orig link -- T
	<p>Makes a symbolic link named link , to the file named orig . Returns true on success, or false on failure.</p> <p><i>Note:</i> Only on macOS, Linux, and RPI.</p>

word	sed/description
link>	s -- s'
	If the named file or folder named is a link or alias, returns what it points to; otherwise returns the original name.
link?	s -- T
	Returns true if the named file is a <i>link</i> to a file.
lock	f ofs len x -- f T (as of 22.03)
	<p>Locks or unlocks the file.</p> <p>When locking: creates an "shared" lock (e.g. "read-only" access) if x is true , and an "exclusive" (no access) if false . ofs and len are numbers which are the offset in the file to start locking, and the number of bytes to lock. If len is 0, it means "lock the rest of the file". Therefore, 0 0 false f:lock will lock the entire file, forbidding any other process access.</p> <p>Returns true if the lock was acquired, or false (and t:err? set) if not.</p> <p>To unlock, pass null as x .</p>
mkdir	s -- T
	Create the named directory. Returns true on success, else false .
mmap	s T -- b
	<p>Opens the named file as "memory-mapped". If T is true if the file should be read-only, otherwise it should be writeable. Note that resizing the file is <i>not possible</i>, it must be whatever size is required before opening it. Returns a buffer containing the contents of the mapped file.</p> <p>See also: f:mmap-range f:mmap-range? f:open</p>
mmap-range	s T start end -- b
	<p>Same as f:mmap , but only maps the file from start to end , which must be numbers within the range of the file size. The start value might be rounded-down to an OS page boundary.</p> <p>See also: f:mmap f:mmap-range? f:open</p>
mmap-range?	b -- b start end

word	sed/description
	<p>Returns start and end offsets of a buffer returned from f:mmap or f:mmap-range , or null if it was not created by f:mmap or the equivalent.</p> <p>See also: f:mmap f:mmap-range f:open</p>
mtime	f -- f d
	<p>Returns the 'last-modified' time of the file or string naming a file, as a date.</p> <p>See also: f:ctime f:atime</p>
mv	oldname newname -- T
	<p>Rename ("move") the file named oldname to the newname . Returns true on success, false otherwise.</p> <p>See also: f:rm f:rmdir</p>
name@	f -- f s (as of 21.02)
	Gets the name of the file.
open	s -- f n -- f
	<p>Opens the file named by the string for read/write access. Returns null if there was an error (no access, or file doesn't exist). If a number is given instead, it is presumed to refer to an <i>already open</i> file-descriptor, and the file returned will wrap it.</p> <p>See also: f:open-ro f:create f:close</p>
open!	s -- f (as of 22.03)
	Same as f:open , but creates the file if (and only if) it doesn't already exist.
open-ro	s -- f
	<p>Same as f:open , but the file is opened in "read-only" mode.</p> <p>See also: f:open</p>
popen	s T -- f

word	sed/description
	<p>Returns a new file which is a "pipe" from which the results of the command s can be read. If it was not successful, returns null . If T is true , the returned pipe is read-only; otherwise it is write-only.</p> <p>See also: f:open</p>
popen3	s -- a
	<p>Takes a string with the name and perhaps parameters of an external command to invoke and returns an array [in,out,err,pid] . The first three are files which can be used to write to the command and read its output. The last is the process-id of the command. Returns null if anything failed.</p> <p>Not on mobile platforms.</p>
print	s -- (as of 16.02)
	<p><i>needs file/print</i></p> <p>Attempts to print the file named by the <i>string</i> using the default OS-specific method. Does <i>not</i> notify if there is an error. Uses CUPS on macOS, Linux and RPI; uses "ShellExecute" on Windows.</p> <p>Currently does nothing on mobile platforms. As of 20.01 moved to file/print library</p>
read	f sb n -- f sb n f n -- f b n
	<p>Read n bytes from the open file, into the string or buffer sb . The number of bytes actually read is left on TOS and sb contains that many bytes. If there was a problem, null is on TOS and sb may contain partial data,</p> <p>If no string or buffer is given then a buffer is created to accommodate n bytes of data.</p> <p>See also: f:open f:write f:read-buf</p>
read-buf	f n -- f b (as of 22.04)
	<p>Read n bytes from the file f , creating a new buffer with the data, or null on error (setting t:err?).</p> <p>This should be used in preference to f:read because the SED is simpler, and the number of bytes read is implicit in the length of the returned buffer.</p> <p>See also: f:open f:write f:read</p>
read?	f s n -- f s n (as of 19.06)
	<p>Same as f:read , but will read fewer than n bytes if that's all there is. In particular, if reading from a device rather than a disk-file it may read fewer bytes.</p> <p>See also: f:read</p>

word	sed/description
relpath	f s -- s' name s -- s'
	<p>Returns a file name, which is the original file's name adjusted relative to the folder name s. It may be given a file or a string naming a file. If a relative path is impossible to produce, the returned path will be an absolute one.</p> <p>See also: f:abspath</p>
rglob	s -- a
	<p>Same as glob, but recursively finds all files in subdirectories as well.</p> <p>See also: f:glob</p>
rm	s -- T
	<p>Remove the named file from the system. Returns true if it succeeded, false otherwise.</p> <p>See also: f:rmdir f:mv</p>
rmdir	s -- T
	<p>Remove the named directory from the system, as well as all its contents. Returns true if it succeeded, false otherwise.</p> <p>See also: f:rm f:mv</p>
seek	f n -- f
	<p>Seek to position n in the file. If n is negative, it seeks from the end of the file, otherwise it seeks from the beginning.</p> <p>See also: f:open f:create f:read f:tell</p>
sep	-- n
	<p>Returns the ASCII code of the character used for separating path components. Use this instead of hard-coding "\" or "/".</p>
size	f -- f n
	<p>Returns the byte size of a regular file or string naming a file f. If f is a zip-file, returns the number of entries in it.</p>
slurp	s -- b f -- b

word	sed/description
	Open the file named by the string (or an open file) and return its contents as a buffer. The name may begin with "~" to represent the user's HOME directory. If the file does not exist or if there is not enough memory to hold its contents, null is returned. The buffer returned is read-only, and is memory-mapped onto the file's contents.
sparse?	f -- T s -- T (as of 19.02)
	If the file (or string containing the name of a file) is a "sparse file", returns true . Otherwise, return false .
spit	x s -- (as of 21.03)
	Inverse of f:slurp . Creates a file named s and writes the buffer or string x to it. Check G:error? for failure.
stderr	-- f
	Open a file corresponding to the "standard error output". Desktop systems only: returns null on mobile systems. See also: f:stdin f:stdout
stdin	-- f
	Open a file corresponding to the "standard input". Desktop systems only: returns null on mobile systems. See also: f:stdout f:stderr
stdout	-- f
	Open a file corresponding to the "standard output". Desktop systems only: returns null on mobile systems. See also: f:stdin f:stderr
tell	f -- f n
	Returns the current read or write position in the file. See also: f:open f:create f:read f:seek
tempfile	-- f (as of 22.03)
	Creates a temporary file which will be automatically deleted as per f:autoDel .
times	s d1 d2 -- s n1 n2 -- (as of 16.04)

word	sed/description
	Sets the access time d1 and modification time d2 of the named file. If either is null , do not change that time; otherwise either may be a number (Unix timestamp) or a date.
tmpspit	x -- s (as of 21.03)
	Uses f:spit to write the item x to a temporary file. Returns the name of the temporary file.
trash	s -- T
	Try to move the named file or folder named to the OS-specific "trash". On success, returns true , otherwise false .
truncate	f n -- f s n -- s (as of 19.02)
	Changes the length of the file or string naming a file to be n bytes. Sets t:err? , which will indicate what failed if there was a failure.
ungetb	f b -- f
	<i>needs file/getc</i> Take the given byte and "unget" it, so that the next f:getb will return it.
ungetc	f c -- f
	<i>needs file/getc</i> Take the given Unicode character and "unget" it, so that the next f:getc will return it.
unzip	f s T -- f (as of 16.04)
	Unpacks the entire ZIP file to the named destination folder. If T is true , overwrites any existing entry by that name. Otherwise, leaves them alone. See also: f:unzip-entry
unzip-entry	f s ix T -- f (as of 16.04)
	Unpacks one item from the ZIP file designated by the index into the named destination folder. If T is true , overwrites an existing entry by that name. Otherwise, leaves it alone. See also: f:unzip
watch	m -- X (as of 18.06)

word	sed/description
	<p>Watches files or sockets listed in the key "files" of the map, for events listed in the "events" key. Returns an item which when refcounted to 0 (or when G:free is invoked on it) will release the watches on the items listed. Keys in opts may be:</p>
write	f sb -- f n
	<p>Write the string or buffer sb to the open file. The number of bytes written is left on TOS, or null if there was a problem.</p> <p>See also: f:open f:read f:create f:writen</p>
writen	f sb n -- f n
	<p>Write n bytes to the file from the string or buffer sb . The number of bytes written is left on TOS, or null if there was a problem.</p> <p>See also: f:write</p>
zip+	f s x -- f
	<p>Add item x to the new zip-file f , giving it the name s in the zip directory. If x is a string, it is considered to be a file-name. If it is a buffer, it is considered to be data to be put in the zip-file.</p> <p>See also: f:zipnew f:zipsave f:zipopen f:zip@</p>
zip@	f x -- f b
	<p>Read item x from the zip file f , returning a buffer b or null . If x is a number, then the item with that index is returned. If x is a string, the first item with that name is returned.</p> <p>See also: f:zipnew f:zipsave f:zip+ f:zipopen</p>
zipentry	f ix -- f d n s
	<p>Returns information about the entry at index ix in the zip file. Returns the string name of the entry s , its size n in bytes, and the date of the item.</p>
zipnew	-- f
	<p>Create a new zip-file in-memory. The file can be saved to disk using f:zipsave .</p> <p>See also: f:zipsave f:zip+ f:zipopen f:zip@</p>

word	sed/description
zipopen	sb -- f
	<p>Open the zip-file sb , which may be either a buffer of zip data, or a string denoting a file-name, creating the zip file f . Returns null if there was an error.</p> <p>See also: f:zipnew f:zipsave f:zip+ f:zip@</p>
zipsave	f s --
	<p>Save the in-memory zip-file f created with f:zipnew or opened with f:zipopen to the file named by the string s . After saving the zip, no further write operations are possible on it.</p> <p>See also: f:zipnew f:zip+ f:zipopen f:zip@</p>

Font

Namespace: **font**

Description: Font utilities

word	sed/description
atlas!	font s -- n (as of 21.06)
	<p>Adds the font to the global font-atlas under the given name. Returns a number which can be used as an alternative to the name to access this specific font. The name might also be null , in which case the font can only be accessed by its number. Adding a font to the atlas with the same name as a previously added font will make the previously added font accessible solely by the number it was assigned.</p>
atlas@	n -- font s -- font (as of 21.06)
	<p>Retrieves a font from the font-atlas. The parameter is either a number (as returned by font:atlas!) or a string given to font:atlas! . If the font does not exist, null is returned.</p>
default-size	n -- (as of 21.06)
	<p>Sets the default font size (if not otherwise specified) to n pixels, for fonts declared or created after it is invoked.. The default value is 12.</p>
default-size@	-- n (as of 21.09)

word	sed/description																						
	Gets the default font size (in pixels)																						
info	font -- font m (as of 17.04)																						
	<p>Returns a map containing information about the given font (if null , the current nk font). The keys returned include:</p> <table><tr><th>key</th><th>description</th></tr><tr><td>ascent</td><td>Maximum height of font above baseline</td></tr><tr><td>desc</td><td>A textual description of the font as ".s" would display it</td></tr><tr><td>descent</td><td>Maximum depth of font below baseline</td></tr><tr><td>height</td><td>Pixel height the font was created at</td></tr><tr><td>lineGap</td><td>Spacing between one row's descent and next row's ascent</td></tr><tr><td>over_h</td><td>Horizontal oversampling (1-8)</td></tr><tr><td>over_v</td><td>Vertical oversampling (1-8)</td></tr></table> <p>If any codepages have been loaded, the key "pages" is an array of maps:</p> <table><tr><th>key</th><th>description</th></tr><tr><td>cp</td><td>The page id (the codepage/256)</td></tr><tr><td>nchars</td><td>Number of valid characters (codepoints) in the page</td></tr></table>	key	description	ascent	Maximum height of font above baseline	desc	A textual description of the font as ".s" would display it	descent	Maximum depth of font below baseline	height	Pixel height the font was created at	lineGap	Spacing between one row's descent and next row's ascent	over_h	Horizontal oversampling (1-8)	over_v	Vertical oversampling (1-8)	key	description	cp	The page id (the codepage/256)	nchars	Number of valid characters (codepoints) in the page
key	description																						
ascent	Maximum height of font above baseline																						
desc	A textual description of the font as ".s" would display it																						
descent	Maximum depth of font below baseline																						
height	Pixel height the font was created at																						
lineGap	Spacing between one row's descent and next row's ascent																						
over_h	Horizontal oversampling (1-8)																						
over_v	Vertical oversampling (1-8)																						
key	description																						
cp	The page id (the codepage/256)																						
nchars	Number of valid characters (codepoints) in the page																						
ls	-- a																						
	Returns an array of strings containing the names of all fonts known to the system.																						
measure	font s -- font n																						
	Returns the width in pixels of the string using that font.																						
new	null -- font s -- font b -- font n -- font a -- font font -- font m -- font																						
	<p>Creates a new font. If it is passed a:</p> <ul style="list-style-type: none">• null: the default system font will be produced• string: following the same rules as for any gui item (see the manual)• buffer: containing a TTF or OTF font file (e.g. from an asset)• number: the default font in that pixel size• array: a list of any of these possible items, the first one to yield a font wins• font: creates a clone of the font• map: with the keys "size" (in pixels), "font" (any of the above), "name" by which it is known in the atlas																						

word	sed/description
oversample	hor ver -- (as of 21.06)
	Sets the "oversampling" values (horizontal, vertical) for succeeding font declarations. These oversampling values are used when rendering the font into bitmaps for display. The values may be between 1 and 8, inclusive. Large values increase the size of the font bitmaps, but may make the image more legible. The default values are 3 and 1, meaning horizontal oversampling only, which produces reasonably sized bitmaps with nice quality.
pixels	font n -- font
	Changes the size of the font to n pixels.
pixels?	font -- font n
	Returns the font's height in pixels.
system	n -- s
	DEFERRED <i>needs font/loaded</i> Returns a string for a 'system font' in the given pixel size. The default fonts are searched; if none of them are found, the first font returned by font:ls is used.
system	-- a
	<i>needs font/loaded</i> Returns an array containing all the fonts present in the system's usual font locations.

Global

Namespace: **G**
 Description: Catch-all class

word	sed/description
!	x v -- v x --

word	sed/description
	<p>Put the item x in the variable v . x may be from <i>any</i> namespace, but if it itself is a variable then the first SED x v -- is required.</p> <p>See also: G:@ G:var G:var,</p>
!if	-- (as of 22.06)
	<p>IMMEDIATE</p> <p>Same as not if but more efficient.</p> <p>See also: if else then not</p>
#!	--
	<p>IMMEDIATE</p> <p>Like \ , but intended for use on Linux or macOS where this tells the system which interpreter to use. If you do use it this way, make sure to put a space after it so 8th recognizes it properly and ignores it.</p> <p>See also: G:SED: G:-- G:\ G:(*</p>
##	n --
	<p>Set the precision, e.g. the number n of digits to the right of the decimal (for floating point; integers will be printed only to the left of the decimal). The default is 5. Setting it to 0 will print as many places as are necessary.</p> <p>See also: G:n#</p>
#if	f --
	<p>IMMEDIATE</p> <p>Analogous to if but intended for "pre-processing". It parses the input stream for "#then", and then parses that for "#else". It then evaluates the #else or #then text depending on the value held in TOS. Used mainly to select different code to compile based on platform or other load-time environmental factors.</p> <p>See also: G:if</p>
'	-- w
	<p>IMMEDIATE</p> <p>Returns the word whose name is <name> , and put it on TOS. Returns null if it cannot find <name> .</p> <p>See also: w:find</p>
(--

word	sed/description
	<p>IMMEDIATE</p> <p>Begin compiling a new anonymous word. The <code>)</code> word finishes the process; if you forget it, you'll stay in compile mode forever...</p> <p>See also: <code>G:) G:: G;; G;;;</code></p>
<code>(*)</code>	<code><*></code> -- (as of 17.06)
	<p>IMMEDIATE</p> <p>Multi-line comment. This word parses until it finds a <code>"*)"</code> sequence in the input, and will drop all that text. Handles nested comments of its own type, e.g. <code>(* foo (* bar *) *)</code>.</p> <p>See also: <code>\ --</code></p>
<code>(:)</code>	<code>s --</code>
	Begin defining the word named <code>s</code> .
<code>(code)</code>	<code>-- used allocated</code>
	Returns the number of bytes of code space used as well as the total amount allocated . These are global over all tasks.
<code>(defer)</code>	<code>s --</code> (as of 20.04)
	<p>Same as <code>G:defer</code>, but instead of parsing the name to create it is given a string.</p> <p>See also: <code>w:undo w:is G:defer G:deferred:</code></p>
<code>(dump)</code>	<code>x -- s</code> (as of 20.04)
	Creates a string containing a hex-dump of the contents of <code>x</code> .
<code>(getc)</code>	<code>-- n</code>
	<p>Returns one character from stdin or the console. Reads as an ASCII character from stdin, or a Unicode character from the console.</p> <p>See also: <code>G:(gets) con:accept con:accept-pwd</code></p>
<code>(gets)</code>	<code>-- s</code>

word	sed/description
	<p>Read a string from stdin or the console.</p> <p>See also: G:(getc) con:accept con:accept-pwd</p>
(interp)	--
	<p>Start the "REPL loop". Used internally.</p>
(log)	x --
	<p>Write the item x to the 8th logging facility, converting it to a string if necessary. Only the first 511 bytes of the string will be logged.</p>
(needs)	s -- (as of 17.05)
	<p>Load the named 8th library. This is a factor of G:needs , but is useful if you need to load a library which has a space in the file or folder name (8th does not have such libraries), or if you want to load a library dynamically.</p>
(parseIn)	-- s
	<p>Same as G:parseIn , but does not consume the CR/LF at the end of the line if the current character is at the end of the line</p> <p>See also: G:parse G:parsews G:parsech</p>
(putc)	n --
	<p>Print the Unicode character to stdout.</p> <p>See also: G:putc G:. G:(puts)</p>
(puts)	s --
	<p>Print the string to stdout.</p> <p>See also: G:putc G:. G:(putc)</p>
(stat)	n -- a

word	sed/description
	<p>For the namespace identifier n , return an array containing in order:</p> <ol style="list-style-type: none"> 1. the number of items on the free list 2. the total number of items allocated 3. the number of words in the namespace 4. the namespace short name 5. the namespace long name 6. the number of abandoned items <p>If an invalid namespace number is given, returns null .</p> <p><i>Note:</i> this only returns valid information if you've turned on allocation counting with G:counting-allocations !</p>
(with)	s --
	Internal factor of G:with
)	-- w
	<p>IMMEDIATE</p> <p>End compilation of a new anonymous word, leaving it on TOS. If there is no matching (, an exception will be thrown.</p> <p>See also: G:(G:: G;; G;;;</p>
+hook	w -- n (as of 19.04)
	<p>Adds a "last-gasp handler". It gets the name of any item the interpreter cannot handle. It returns the index of this gasp-handler, which may be passed to G:-hook in order to remove it.</p> <p>The SED for w is: s -- T . If it recognizes the item, it processes it and returns true ; otherwise it returns false .</p> <p>See also: G:-hook G:.hook</p>
+ref	x -- x
	<p>Increase the "reference-count" of the item x . Usually you should not need to use this.</p> <p>See also: G:-ref G:ref@</p>
,#	n --

word	sed/description
	<p>Sets the thousands separator for printing numbers to the ASCII character given by the number n . If zero, no separator will be used (the default).</p> <p>See also: G:.# G:n# G:c# s:strfmt s:fmt</p>
--	--
	<p>IMMEDIATE</p> <p>Same as \% , this is a SQL-style comment.</p> <p>See also: G:SED: G:#! G:\ G:(*</p>
-----BEGIN	<...> -- (as of 18.06)
	<p><i>needs crypto/pgp</i></p> <p>Ignores everything from the "-----BEGIN " until and including "-----END PGP SIGNATURE-----". This allows you to embed PGP signatures in your scripts, allowing your users to be confident the script has not been modified and that it is from you.</p> <p>See also: cr:validate-pgp-sig</p>
-Inf	-- n
	<p>Put the number with value -Inf on TOS, "negative infinity".</p> <p>See also: G:Inf G:NaN</p>
-Inf?	o -- o f
	<p><i>needs math/isa</i></p> <p>Return f which is true if the item o is -Inf, or false otherwise.</p>
-hook	n -- (as of 19.04)
	<p>Removes a "last-gasp handler" added with G: +hook . If the value n is not one which was returned by G: +hook , or if the handler has already been removed, it will be ignored.</p> <p>See also: G:+hook G:.hook</p>
-ref	x -- x

word	sed/description
	<p>Decrease the "reference-count" of the item x . Usually you should not need to use this.</p> <p>See also: G:+ref G:ref@</p>
-rot	a b c -- c a b
	<p>Move the item in TOS to the third position on the stack.</p> <p>See also: G:rot G:swap G:over G:drop G:nip G:tuck</p>
.	x --
	<p>Print the item x , converting it to a string first if necessary.</p> <p>See also: G:putc G:(putc) G:(puts)</p>
.#	n --
	<p>Sets the decimal separator for printing numbers to the ASCII character given by the number n ; if zero, a period will be used (the default).</p> <p>See also: G:;# G:n# G:c# s:strfmt s:fmt</p>
.hook	-- (as of 19.04)
	<p>Displays active hooks installed via G:+hook along with their index numbers.</p> <p>See also: G:+hook G:-hook</p>
.needs	-- (as of 16.04)
	<p>Prints the names of all libraries loaded by G:needs , one per line.</p> <p>See also: G:needs</p>
.r	--
	<p>Print the top 10 items in the "r-stack", in the same manner that G:.s does for the data stack.</p>
.s	--

word	sed/description
	<p>Display up to st:dot-depth items on top of the data stack (default is 10 items). The display shows the index on the stack, the namespace name, the address of the item, its reference-count and finally its value. If there is nothing on the stack, prints "Stack empty". If the item is currently locked (see G:lock), an asterisk ("*") appears after the stack index and before the namespace name. TOS is item '0'.</p> <p>The first item (TOS) also shows the depth of the stack, e.g. 0/11 if there are 11 items on the stack.</p> <p>See also: G:r G:depth</p>
.s-truncate	T -- (as of 20.08)
	Controls whether or not G:.s truncates output. By default it does; to disable truncation, pass false .
.stats	--
	<p><i>needs utils/stats</i></p> <p>Display system statistics. The first line shows how much code space has been used vs. how much is allocated (in bytes). Then, for each <i>namespace</i> it shows its name followed by how many <i>words</i> are currently defined in it, how many data items of that type are in use, how large the pool is for that <i>namespace</i> and finally, how many pools have been allocated.</p>
.ver	--
	<p>Print the version of 8th, including build number, OS and bit-size, and customer id. Does nothing, in mobile builds.</p> <p>See also: G:8thver? G:buildver? G:build?</p>
.with	--
	<p>Displays the current "with list".</p> <p>See also: G:with: G;;with</p>
0;	x --
	If x evaluates false , drop it and exit the word; otherwise, continue execution. Used within a word instead of the phrase dup !if drop ;; then .
2dip	a b c w -- w(a) b c
	<p><i>needs combinators/2dip</i></p> <p>Implementation of Factor's "2dip" combinator. Invokes w on the items in position 3 (etc.) on the stack, moving the items in positions 1 and 2 out of the way and restoring them afterwards to TOS.</p>

word	sed/description
2drop	a b c d -- a b
	Drop the two top items from the stack. See also: G:dup G:2dup G:2over G:2swap
2dup	a b -- a b a b
	Duplicate the two top items on the stack. See also: G:dup G:2drop G:2over G:2swap
2nip	a b c -- c (as of 22.05)
	Drops the two items under TOS. See also: G:tuck G:drop G:dup G:swap G:over G:nip
2over	a b c d -- a b c d a b
	Copy items in positions 3 and 4 to TOS. See also: G:dup G:2dup G:2drop G:2swap
2swap	a b c d -- c d a b
	Like G:swap but for the two sets of two items on TOS. See also: G:dup G:2dup G:2drop G:2over
2tuck	a b c d -- c d a b c d (as of 23.06)
	Same as tuck but copies the top two items on the stack under the two items below them.
3drop	a b c d -- a (as of 22.05)
	Drops the three top items from the stack. See also: G:dup G:2dup G:2over G:2swap
3drop	x y z --

word	sed/description
	<i>needs stack/3drop</i> Drops three items off the data stack
3dup	a b c -- a b c a b c (as of 22.08)
	Duplicates the three top items on the stack. See also: G:dup G:2dup G:2drop G:2over G:2swap
3rev	a b c -- c b a (as of 21.03)
	Same effect as -rot swap but faster and more efficient.
4drop	x y z a --
	<i>needs stack/3drop</i> Drops four items off the data stack
8thdt?	-- s (as of 18.01)
	Returns a readable string containing the build date of 8th.
8thsku	-- n (as of 21.03)
	Returns the running 8th's SKU: 0 = Free, 1 = Hobbyist, 2 = Professional, 3 = Enterprise
8thver?	-- s
	Returns a string of the 8th version. e.g.: "23.08" or the like. See also: G:buildver? G:build? G:.ver
8thvernum?	-- n (as of 21.02)
	Returns the 8th version as a number in the format major, minor, patch, e.g. '210100'.
:	--

word	sed/description
	<p>IMMEDIATE</p> <p>Begin defining the word named <name> . May only be used in interpret mode, and may not be used inside another word definition (e.g. nested words are <i>not</i> allowed).</p> <p>See also: G;; G:(G:) G;;</p>
;	--
	<p>IMMEDIATE</p> <p>Terminate a word definition which was started with G: : . May only be used in compile mode. Must be used after an initial : .</p> <p>See also: G:: G:(G:)</p>
;;	--
	<p>IMMEDIATE</p> <p>Exits the current word immediately, prior to the usual terminating ; .</p> <p>See also: G;; G;;then G:break</p>
;;;	--
	<p>IMMEDIATE</p> <p>Immediately exit the word which invoked the current one (as opposed to G:;; , which exits only the current word).</p> <p>See also: G;;; G;; G:break</p>
;then	-- (as of 17.08)
	<p>IMMEDIATE</p> <p>The equivalent of ;; then to make code a bit more compact.</p>
;with	--
	<p>IMMEDIATE</p> <p>Drops the last with: namespace from the with list.</p> <p>See also: G:with:</p>
>clip	x --

word	sed/description
	Take an item x and put it on the system clipboard. The item's <i>string representation</i> will be put on the clipboard as if G:>s was performed on it first.
>json	x -- s
	<p>Take an item x and convert it to its JSON representation. An equivalent item may be retrieved by giving the resultant JSON to json> .</p> <p><i>Note:</i> this does not work with all namespaces. It only works with the JSON standard types of number, boolean, string, array, map, and null. If you've put e.g. a word in the array or map, it will be converted to the name of the word, which standard JSON parsers will not accept.</p> <p>See also: G:json></p>
>kind	x -- n
	<p>Returns the numeric namespace identifier of the item x . Each namespace identifying number is obtained by using its name in the ns: namespace. Ex: the identifier of strings is ns:s .</p> <p>See also: G:ns</p>
>n	sb -- n n -- n T -- n
	<p>Convert the string or buffer to the number n , obeying the current numeric base . If a number is passed, it is simply returned. If a boolean is passed, returns 1 for true and 0 for false . If none of the above is passed, or the string cannot be converted to a number in the current base , returns null .</p> <p>See also: G:>s</p>
>r	x --
	<p>Pop the item x from the data-stack and push it onto the r-stack.</p> <p>See also: G:r> G:r@ G:rswap G:rdrop</p>
>s	x -- s
	<p>Convert the item x to its string representation s . If x is already a string, returns the same string.</p> <p>See also: G:>n</p>
?:	x1 x2 -- x' (as of 18.08)

word	sed/description
	Returns 'x2' if x1 is null , otherwise x1 . Same effect as swap null? if drop else nip then , but <i>much</i> more efficient.
?@	x1 x2 -- x3 (as of 23.04)
	Similar to ?: , but if x1 is null , then if x2 is a var gets the value it holds, while if x2 is a word, invokes it.
@	v -- x
	Returns the contents of the variable v , which may contain an item from <i>any</i> namespace. See also: G:!
BITMAP:	bits , bits2 , ... ; -- (as of 20.06)
	<i>needs utils/bitmap</i> Create a set of words for each 'bits' value, assuming 'little-endian' order. That is, the first values will be at the least-significant end. For each 'name', creates 'name_bits', 'name_mask', 'name_shift', 'name@', and 'name!'
ENUM:	startfrom ... ; -- (as of 20.01)
	<i>needs utils/enums</i> Same as G:enum: but stops at ";" instead of "enum;"
FLAG:	startfrom ... ; -- (as of 20.01)
	<i>needs utils/enums</i> Like G:ENUM: , but creates a value of 1 shifted left the number of the enum.
I	-- n (as of 22.04)
	Inside loop , loop- , times return the current loop index. <i>Note:</i> the loop index is a singleton, so if you want to store it, clone it first. See also: G:J G:K G:X
Inf	-- n
	Put the number with value Inf on TOS, "positive infinity". See also: G:-Inf G:NaN

word	sed/description
Inf?	o -- o t
	<i>needs math/isa</i> Return true if the item o is Inf , or false otherwise.
J	-- n (as of 22.04)
	Inside loop , loop- , times return the loop index of the next outer loop. <i>Note:</i> the loop index is a singleton, so if you want to store it, clone it first. See also: G:I G:K G:X
K	-- n (as of 22.04)
	Inside loop , loop- , times return the loop index of the second outer loop. <i>Note:</i> the loop index is a singleton, so if you want to store it, clone it first. See also: G:I G:J G:X
NaN	-- n
	Put the number with value NaN on TOS. This is generally the result of dividing zero by zero, for example. Any numeric operation on NaN propagates, so e.g.: NaN 0 + gives NaN . See also: G:Inf G:-Inf
NaN?	o -- o t
	<i>needs math/isa</i> Return true if the item o is the same as NaN , or false otherwise.
SED-CHECK	f --
	<i>needs debug/sed</i> Enable (if f is true) or disable SED checking. The check depends on proper use of SED:
SED:	--

word	sed/description
	<p><i>needs debug/sed</i></p> <p>Overwrites the built-in SED: (which is simply a comment). If true SED-CHECK has been invoked, then any <i>word</i> using this to declare a stack-effect-diagram will have it checked on entry and exit. The SED must use namespace identifiers (e.g. "s" or "n" etc) in order for SED: to properly check data types. If the SED doesn't match the stack at runtime, an exception will be thrown. This can also be enabled by using the '-g' CLI option to 8th.</p> <p>NOTE: use of this <i>word</i> will slow performance, so it should not be enabled in production code.</p> <p>NOTE: does not check the r-stack, and the SED declaration must be simple. e.g.: SED: n n -- n or something like that.</p>
SED:	-- (as of 18.08)
	The builtin version is just a "comment to end of line". It is intended to be overwritten by the version in the "debug/sed" library, which can check the stack-effect diagram against runtime conditions.
TODO:	-- (as of 20.08)
	A comment word, which makes it easier to see items needing to be taken care of.
X	n -- n (as of 22.04)
	<p>Inside loop , loop- , times return the loop index of the n th outer loop, counting from 0 for current loop. Returns 0 if an invalid loop index is given (e.g. if there is no such nested loop).</p> <p><i>Note</i>: the loop index is a singleton, so if you want to store it, clone it first.</p> <p>See also: G:I G:J G:K</p>
**	--
	<p>IMMEDIATE</p> <p>Comment to end-of-line: everything after the "\" is ignored.</p> <p>See also: G:SED: G:-- G:#! G:(*</p>
_dup	a b -- a a b (as of 21.03)
	Same effect as over swap but faster and more efficient.
_swap	a b c -- b a c (as of 21.03)
	Same effect as rot swap but faster and more efficient.
actor:	x w --

word	sed/description
	<p>IMMEDIATE <i>needs utils/actor</i></p> <p>Similar to a <i>var</i>, but an "actor" takes an item x as its value, and a <i>word</i> w to invoke. As with var, the is taken from the input stream and is the name by which the actor is invoked. As of 18.04, 'actor:' is just an immediate wrapper around 'curry:'</p>
again	--
	<p>IMMEDIATE</p> <p>Unconditional return to the most recent unpaired repeat . May only be used in compile-mode, throws an exception otherwise. May be stopped with ;; (which immediately exits the word), or break (which causes the loop to not repeat).</p> <p>If no repeat was given, then it branches to the start of the most recently compiled word (started with with : or (). This is different from recurse in that it does not issue a call/ret, rather just jumps to the start, same as if there was a repeat as the first word compiled in.</p> <p>See also: G:while G:repeat G:while! G:break G:break?</p>
ahead	-- (as of 17.10)
	<p>IMMEDIATE</p> <p>Compiles an unconditional "jump" instruction at the current PC, to be resolved with G:then .</p>
and	T T -- T
	<p>Returns the boolean "AND" of the top two items on the stack.</p> <p>See also: G:or G:not G:xor</p>
apropos	--
	<p>DEFERRED</p> <p>Looks for any word whose description or name contains <text> and prints out the matching words' names.</p> <p>See also: G:help G:words G:words/ app:opts</p>
argc	-- n
	<p>Returns the number of command-line arguments to the running program. Always 0 on mobile.</p> <p>See also: G:args</p>
args	n -- s null -- a

word	sed/description
	<p>Returns the command-line argument number n to the running program. Numbering starts at 0 for the first argument, and accesses any items on the command-line which were not treated as options (by G:process-args). A negative value of n counts from the end (e.g. -1 args returns the last argument).</p> <p>A value of null returns an array of all the arguments.</p> <p><i>Note:</i> On macOS, Linux, and RPI, a 'backslash' on the command line is ignored unless the argument is enclosed in quotes. Forgetting that may lead to unexpected consequences.</p> <p>See also: G:argc</p>
array?	x -- x T
	Returns true if x is an array, false otherwise.
assert	x --
	<p>Assert that the value x in TOS is "true", and quits with a message if it is not. If the value is a string, it is evaluated using eval and then TOS is checked. This allows you to assert that a condition at is valid at runtime, for example:</p> <pre>"100 over n:=" assert</pre>
base	n -- n'
	<p>Set the numeric base to the n and returns the previously active base. The base is set on a per-task basis, the default base is 10.</p> <p>See also: G:decimal G:hex</p>
base>n	s n -- n' (as of 22.08)
	<p>Same as G:>n, but sets the numeric base to n first, and restores it after. So "FF" 16 base>n returns 255.</p> <p>See also: G:>s G:>n</p>
bi	x w1 w2 -- w1(x) w2(x)
	<p><i>needs combinators/bi</i></p> <p>Implementation of Factor's "bi" combinator. Invokes w1 and w2 on x, leaving the results.</p>
bits	-- n
	Returns 32 or 64: the "bit-size" of the current build of 8th (not the OS native bit-size!)

word	sed/description
break	--
	<p>Flags the current loop or iterator to stop at the next iteration point. Does <i>not</i> leave the loop or iterator at the point it is invoked! In addition, if invoked in interpret-mode, will cause the script being evaluated to stop evaluation at that point.</p> <p>Affected loops: while , again , times , loop , loop- , times .</p> <p>Affected iterators: m:each , a:each , s:each , b:each , b:each-slice , xml:each</p> <p>Other affected words: a:map , a:filter , a:reduce , a:y , b:op , db:exec-cb , f:eachbuf , m:map , s:eachline f:eachline .</p> <p>See also: G:break?</p>
break?	-- T
	<p>true if G:break was invoked.</p> <p>See also: G:break</p>
breakif	x -- x (as of 20.08)
	<p>Same effect as dup if break then .</p> <p>See also: G:break? G:break</p>
build?	-- s
	<p>Returns a string containing the OS/bits. e.g.: "LIN64" or the like.</p> <p>See also: G:8thver? G:buildver? G:.ver</p>
buildver?	-- s
	<p>Returns a string of the 8th build version. e.g.: "6973156a" or the like.</p> <p>See also: G:8thver? G:build? G:.ver</p>
bye	--
	<p>Quit 8th in the normal manner. The words in the onexit chain will be invoked in reverse order of their addition.</p> <p>See also: G:die G:onexit</p>
c/does	x dw cw --

word	sed/description
	<p><i>needs utils/create-does</i></p> <p>Similar to ANS Forth "CREATE... DOES>". Takes an item x which is the default value, a <i>word</i> dw which is the "run-time" behavior, a <i>word</i> cw which is the "create-time" behavior and a <i>string</i> which is read from the input stream. It creates a new <i>word</i> called , which is itself used to create new <i>words</i>. Those new words it creates will invoke the cw <i>word</i> when they are created, and the dw <i>word</i> when they are run. In the case of dw, it is passed a clone of the item x. In the caes of cw, it is passed an <i>array</i> which contains the item x as well as the dw <i>word</i>. Thus, the cw <i>word</i> may modify the value or the action taken at run-time.</p>
case	s table -- n table -- table s -- table n -- (as of 16.12)
	<p>Similar to G:caseof , but takes a number or string, and an array or map table . The table consists of (key,value) pairs where the value is a word (or [value,value...] if an array). If the key exists in table , its value is invoked. If not, nothing happens. As with G:caseof , the parameters may be in reverse order as well.</p> <p>See also: G:caseof</p>
case:	-- (as of 19.03)
	<p><i>needs utils/case</i></p> <p>Begins a "case:... case;" block, similar to the C "switch" statement. Use with of: ... of; and default: ... of; to define the cases. Uses caseof internally, but provides a more "friendly" syntax.</p>
caseof	a n -- x n a -- x m s -- x s m -- x
	<p>A "case" construct which takes an array or map, and a number or string, as appropriate. It looks up the value in the container, and then if x is:</p> <ul style="list-style-type: none"> • a word: invokes it and leaves the result (if any) on TOS • not found: puts null on TOS • otherwise, puts x itself on TOS. <p>The order of the container vs scalar isn't important, thus eliminating the need to invoke swap .</p> <p>See also: G:case</p>
catch	w -- x true w --false (as of 20.05)
	<p>Invokes the word. If an exception was thrown, returns the exception and true ; otherwise just returns false . Use with caution, since exceptions are generally intended to be fatal errors. G:catch will install its own temporary t:handler (and thus it is per-task). A typical usage might be</p> <pre>' possible-thrower catch if \ caught an exception... handle-exception then</pre>
chdir	s --

word	sed/description
	<p>Change the current directory.</p> <p>See also: G:getcwd</p>
clip>	-- s
	Returns the current contents of the system clipboard.
clone	x1 -- x1 x2
	<p>Create a "clone" x2 of the item x1 . Like dup , a "clone" has the same value as the original, but it is not the same actual item.</p> <p>Modifying the clone does not change the value of the original item x . If x1 is a container type, all the contained items are also cloned.</p> <p><i>Note:</i> some item types cannot be cloned: "db", "sql", "hw", and "X".</p> <p>See also: G:clone-shallow G:same? G:dup</p>
clone-shallow	x1 -- x1 x2
	<p>The same as G:clone , except that if x1 is a container type then the items <i>contained</i> are not cloned. That is, a shallow-clone of a container creates a new container with the same actual items as the original.</p> <p>See also: G:clone G:dup G:same?</p>
cold	--
	The default system startup, begins the interpreter loop.
compile	w --
	<p>IMMEDIATE</p> <p>Compiles an invocation of the word into the word currently being compiled. When that word is subsequently invoked, the word w will be invoked at that point.</p> <p>See also: G:literal G:compile?</p>
compile?	w --

word	sed/description
	<p>IMMEDIATE</p> <p>Same as G:compile , except it only compiles if 8th is in compile mode; otherwise, it invokes the word w .</p> <p>See also: G:literal G:compile</p>
compiling?	-- n (as of 19.03)
	Returns non-zero if 8th is currently in "compile mode", 0 otherwise.
conflict	s -- T
	<p>DEFERRED</p> <p>Determine how 8th responds if a new word or other named item already exists in the current namespace. It is passed the name of the conflicting item, and returns true to permit it to be created, or false to disallow it. The default 8th behavior is to <i>permit</i> overwriting of existing items, but to <i>warn</i> that it is occurring.</p>
const	x1 -- x2
	<p>Equivalent of clone nip , used to ensure a <i>copy</i> of a container is used rather than the original. Generally used in the case where a container item is compiled into a word, and the container itself is not to be modified.</p> <p>See also: G:clone</p>
constant	x -- (as of 19.01)
	Creates a "constant" value. This is essentially syntactic sugar for the prior method of creating a new word containing the value x . Upon invocation of the name, x is on TOS.
container?	x -- x T (as of 19.01)
	Returns true if the item x is a "container", e.g. a type which contains other items.
counting-allocations	T -- (as of 19.04)
	<p>If T is true , causes allocations of new items to be counted. This makes the G:.stats word return proper information. The default is false , meaning items are not counted when allocated.</p> <p>See also: G:.stats</p>
cr	--

word	sed/description
	<p>Print an OS-specific 'newline' sequence.</p> <p>See also: G:., G:puts G:putc s:strfmt G:space</p>
critical:	-- (as of 23.09)
	<p>IMMEDIATE</p> <p>Begins a "critical section" within a word. Locks that section of code so that another task cannot execute it simultaneously. May not cross word boundaries and must be paired with G:critical;</p> <p>See also: G:critical: G:lock G:unlock</p>
critical;	-- (as of 23.09)
	<p>IMMEDIATE</p> <p>Ends a "critical section" started with G:critical: .</p> <p>See also: G:critical: G:lock G:unlock</p>
curlang	-- v
	<p>A var containing a string determining the current language used by s:intl etc. The default value is "en", for English. It should be set by the appropriate "lang/xx" asset.</p> <p>See also: s:intl s:lang</p>
curry	x w -- w' (as of 18.04)
	<p>Returns a new anonymous word which takes a parameter x . Encapsulates w(x) as w' .</p> <p>See also: G:curry:</p>
curry:	x w -- w' (as of 18.04)
	<p>Same as G:curry , but creates a named word.</p> <p>See also: G:curry</p>
decimal	--
	<p>Set the numeric base to 10. Equivalent to 10 base drop .</p> <p>See also: G:base G:hex</p>

word	sed/description
default:	-- (as of 19.03)
	<i>needs utils/case</i> Used with G:case: to enumerate the default case.
defer:	--
	Create a deferred word <name> . This new word initially does nothing (its action is G:noop). Assign it an action using w:is , remove an action with w:undo . See also: w:undo w:is
deferred:	w -- (as of 20.02)
	Just like G:defer: except that it assigns the word w as the action of the newly created deferred <name> . See also: w:undo w:is G:defer:
deg>rad	n -- n'
	Converts from degrees to radians. See also: G:rad>deg
depth	-- n
	Current depth of the data-stack, e.g. the number of items it contains (before pushing the depth to TOS).
die	n --
	Stop 8th abnormally, with n as the program's return code. The words in the onexit chain will be invoked in reverse order of their addition. See also: G:bye
dip	a b w -- w(a) b
	<i>needs combinators/dip</i> Implementation of Factor's "dip" combinator. Invokes w on the items in position 2 (etc.) on the stack, moving the item in position 1 out of the way and restoring it afterwards to TOS.
drop	a b -- a

word	sed/description
	<p>Drop the item on TOS.</p> <p>See also: G:dup G:2drop</p>
dstack	-- st
	<p>Returns a reference to this task's "data-stack", similar to G:rstack .</p> <p>See also: G:rstack</p>
dump	x --
	<p>Do a hex-dump of the contents of the item x . Exactly what information is dumped depends on the type of x .</p>
dup	x -- x x
	<p>"duplicate" the item on TOS. This makes an <i>additional reference</i> to that item, it does <i>not</i> create a separate copy. If you wish to make a separate copy, use G:clone .</p> <p>See also: G:dup? G:clone G:drop</p>
dup>r	x -- x
	<p>Duplicates TOS and pushes it onto the r-stack. Same effect as dup >r but faster.</p> <p>See also: G:r> G:r@ G:rswap G:rdrop</p>
dup?	x -- x x null -- null
	<p>Duplicate the item on TOS if and only if it is not null .</p> <p>See also: G:dup</p>
e#	T -- (as of 20.04)
	<p>If true , numbers will be printed using scientific notation. The default is false . As with the other formatting words, this only affects the task in which it is invoked.</p>
else	--

word	sed/description
	<p>IMMEDIATE</p> <p>Begin alternate branch in a if... else... then block. May only be used in compile-mode, throws an exception otherwise. Also throws an exception if there was no accompanying if .</p> <p>See also: G:if G:then</p>
enum:	startfrom ... enum; --
	<p><i>needs utils/enums</i></p> <p>Create an "enumeration", e.g. words which return a numeric (increasing) value. Give a value to start from, and each name thereafter becomes a new "enum" -- up until the word "enum;," which ends the sequence.</p>
error?	-- n (as of 19.02)
	Returns the task-specific errno. A non-zero value means something is amiss.
eval	s --
	<p>Interpret the string (or buffer) as if it were typed into the interpreter. If s is null , nothing happens.</p> <p>See also: G:eval!</p>
eval!	s -- T
	<p>Same as G:eval , but will not throw exceptions if evaluating the string would cause one. Instead, returns a true if no exceptions would have been thrown, false otherwise.</p> <p>See also: G:eval</p>
eval0	s -- (as of 17.08)
	<p>Similar to G:eval but ensures the system is in "interpret mode" prior to evaluation. The mode is reset to what it was originally.</p> <p>See also: G:eval</p>
exit	n -- (as of 24.06)
	<p>IMMEDIATE</p> <p>Causes a return of n words from the current word. So 1 exit is the same as ;; (and it's the implementation of it). Note: if there are fewer than n items on the system's return-stack, will throw an exception.</p>
expect	a x -- a flag

word	sed/description
	<p><i>needs utils/expect</i></p> <p>Tests the >kind of a against the <i>number</i> or <i>array</i> of numbers x. If a is not one of the listed types, puts false on TOS otherwise, puts true on TOS. In either case, the original item a is left under TOS.</p>
extra!	<p>x x' -- x (as of 18.03)</p> <p>Sets the item y as "extra" data associated with the item x .</p> <p><i>Note:</i> An item may not be associated with itself.</p> <p>See also: G:extra@</p>
extra@	<p>x -- x' (as of 18.03)</p> <p>Gets any "extra" data associated with a particular item. <i>Any</i> item may have "extra" data associated with it. If no item is currently associated with x , null is returned.</p> <p>See also: G:extra!</p>
false	<p>-- T</p> <p>Returns the boolean value false .</p> <p>See also: G:true</p>
fnv	<p>s -- n</p> <p>Returns the FNV1a hash of the given string.</p>
fourth	<p>a b c d -- a b c d a (as of 17.08)</p> <p>Copies the fourth item on the stack to TOS. Same as 3 pick .</p>
free	<p>x -- (as of 17.08)</p> <p>Frees the item x by repeatedly decreffing it until it has a zero refcount. This should not normally be used, but is useful in some cases.</p>
func:	<p>param s</p>

word	sed/description
	<p>Create a word called <name> which will invoke the external function s from the G:lib created library last invoked when creating the function.</p> <p>At runtime, the parameter string param will be used to convert 8th types from the stack to native and vice-versa. Parameters are in the order expected by the external function (from low to TOS on the stack) and may also be given in an array.</p> <p>The parameter string contains the return value as its first character, followed by one character for each parameter to the function, with a maximum of 12 parameters total.</p> <p>Please view the manual's chapter on FFI for a complete description of the parameters.</p> <p>See also: G:lib w:cb</p>
getc	-- n
	<p>DEFERRED</p> <p>Read one character from the input (standard input or console) and return as a number n. Default behavior is G:(getc).</p> <p>See also: G:gets G:(getc) con:accept con:accept-pwd</p>
getcwd	-- s
	<p>Returns the current working directory.</p> <p>See also: G:chdir</p>
getenv	s -- s'
	<p>Returns the value of the environment variable, or null if the variable doesn't exist.</p> <p>See also: G:setenv</p>
gets	-- s
	<p>DEFERRED</p> <p>Read from standard input or the console, up to a newline and return a string s. Default behavior is G:(gets)</p> <p>See also: G:getc G:(gets) con:accept con:accept-pwd</p>
goto	n -- (as of 24.06)

word	sed/description
	<p>IMMEDIATE</p> <p>Compiles a non-returning jump to the 8th code address n . Used by G:recurse . If n is not in the range of valid 8th addresses, fails.</p>
handler	x -- T
	<p>DEFFERED</p> <p>Handle an exception. Receives item x which describes the exception (usually a string, but you can throw any item type); return true to continue or false to quit. The default is to quit.</p> <p>See also: G:throw</p>
header	s --
	<p>Creates a new dictionary entry named s whose code is left unfilled, and does not change to compile mode.</p> <p>See also: G:: G:(G:(:)</p>
help	--
	<p>DEFFERED</p> <p>Looks for any word <name> and prints out the documentation for it to the console. The name matching is case-sensitive. The default width of the output is 132 (or fewer) characters, or the width of the terminal, whichever is smaller. You can set the maximum by setting the "help.width" key using app:opts .</p> <p>If a fully-qualified name is given, only that word's help is shown. Otherwise, all matching words' help is shown.</p> <p>See also: G:apropos G:words G:words/ app:opts</p>
help_db	-- v
	<p><i>needs utils/help</i></p> <p>Variable containing the name of the help database.</p>
here	-- n (as of 24.06)
	Returns the execution address/token to be compiled next.
hex	--
	<p>Sets the numeric base to 16. Equivalent to 16 base drop .</p> <p>See also: G:base G:decimal</p>

word	sed/description
i:	--
	<p>IMMEDIATE</p> <p>Immediately cause the action of the next word. Ensures the next word is run immediately rather than compiled in. Only use on non-"immediate" words.</p> <p>See also: G:p: G:l: G:i;</p>
i;	--
	<p>IMMEDIATE</p> <p>When used instead of ; , marks the word being compiled as being <i>immediate</i>, e.g. one which is immediately invoked rather than compiled, and terminate the word's compilation just as ; does. Must be paired with : .</p> <p>See also: G:p: G:l: G:i: G;;</p>
if	--
	<p>IMMEDIATE</p> <p>Begins a conditional if... else... then block. At runtime, examines (and consumes) TOS. If it evaluates true , the code immediately following the if is run, until the matching else or then .</p> <p>Otherwise, the code after the else or then is run. May only be used in compile-mode, throws an exception otherwise. Must be accompanied by then .</p> <p>See also: G:else G:then</p>
if;	x --
	<p>If x evaluates true , exit the current word; otherwise, continue execution. Used within a word instead of the phrase if ;; then .</p>
isa?	x n -- x T
	<p>Returns true if x is an item from the given namespace identifier, false otherwise. Ex: to test if TOS contains a heap you could use ns:h isa? .</p>
items-used	-- n
	<p><i>needs utils/stats</i></p> <p>Returns an <i>array of arrays</i>, each containing the name of the namespace and a count of how many items of each namespace have been used.</p>
jcall	X a -- x

word	sed/description
	<p>Takes an X returned from G:jmethod , and an array of parameters to pass to the method. Makes the function call with the given parameters, and returns the type of data specified in the parameters list of the jmethod .</p> <p><i>Note:</i> Android only!</p> <p>See also: G:jclass G:jmethod</p>
jclass	s -- X
	<p>Returns an X given a string containing the fully qualified name of a Java class (such as "java.lang.Thread"), which may be passed to G:jmethod to create a method by which one may call that method using G:jcall .</p> <p><i>Note:</i> Android only!</p> <p>See also: G:jmethod G:jcall</p>
jmethod	X name params -- X'
	<p>Takes a string containing the name name of a Java method in a Java class previously returned from G:jclass . The params string follows the JNI specification and must match the signature of the Java method <i>exactly</i>. The returned item can then be passed to G:jcall . Please refer to the manual for a complete description of how this is used.</p> <p><i>Note:</i> Android only!</p> <p>See also: G:jclass G:jcall</p>
json!	ma s x -- x (as of 20.04)
	<p>Analogous to G:json@ , but sets the item(s) at the JSONPath s to the value x . If that is a word, then it is invoked with the current value of the item at the path, and the result it leaves on TOS replaces the current item. Its SED is x -- x' . See the manual chapter 11.7 for details of the JSONPath syntax.</p> <p>See also: G:json@</p>
json-8th>	sb -- x (as of 20.03)
	<p>Takes a JSON string or buffer and converts it to an equivalent of the original item given to G:>json . null on TOS will result in null . Unlike G:json> , this handles 8th JSON enhancements.</p> <p>See also: G:>json G:json></p>
json-nesting	n -- (as of 17.08)

word	sed/description
	Sets the limit of nesting JSON when converting items to a string. This affects >s and .s among other things. The default value is 100, which should be sufficient for most uses.
json-pretty	n --
	Tell the >json and other JSON output words to pretty-print JSON so it is easier for humans to read. A value of 0 means don't pretty-print; any other value is the number of spaces to indent each nested level. Legal values are between 0 and 16.
json-throw	T -- (as of 18.04)
	If true , then conversion of a self-referencing JSON to a string will throw an exception. Otherwise, the item will show as in the output string. The default is to <i>not</i> throw.
json>	S -- X
	<p>Take a JSON string (or buffer containing JSON) on TOS and convert it to an equivalent of the original item given to >json. null on TOS will result in null.</p> <p><i>Note:</i> 8th-specific enhancements to JSON are not permitted for security reasons, and will appear as null in the returned item. If you have trusted JSON with 8th enhancements which you want to restore, use json-8th> instead, or perhaps eval.</p> <p><i>Note:</i> Using eval does pose a security risk if the string it is given is not from a trusted source.</p> <p>See also: G:>json G:json-8th></p>
json@	ma s -- ma x true ma s -- ma false (as of 20.04)
	<p>Accesses elements from the map or array using a subset of "JSONPath" syntax, returning false if the element does not exist, or true and the element x. See the manual chapter 11.7 for details of 8th's JSONPath syntax.</p> <p>See also: G:json!</p>
k32	-- n
	Windows only: Returns the lib handle of the "kernel32" system library.
keep	a b w -- w(a,b) b
	<p><i>needs combinators/keep</i></p> <p>Implementation of Factor's "keep" combinator.</p>
l:	--

word	sed/description
	<p>IMMEDIATE <i>needs utils/latebind</i></p> <p>Makes the next word "delayed lookup". Compiles the string <name> into the word currently being compiled. At runtime, every time the code is invoked, it looks up the word with that name. So 1: + will add two numbers, or concatenate two strings, using the same code.</p> <p>See also: G:p: G:i:</p>
last	-- w
	Returns the last word compiled.
lib	s -- a --
	<p>Create a word named <name>, which will load the external named library when invoked. At runtime, invoking the created word has the SED -- X returning an item representing the loaded library's handle, or null if it was unable to load it.</p> <p>The library name s may be a <i>complete</i> name which is OS specific, such as "libiconv.so" or "iconv.dll". If just the base name is given, e.g. "iconv", then 8th will try to load "iconv", then "iconv.dll" (or .so or .dylib, depending on OS) and then "libiconv.dll" (etc). This makes writing cross platform library routines much easier, assuming the dynamic library has the same base name across platforms. That is, "iconv.dll", "libiconv.so" and "libiconv.dylib" in this example.</p> <p>If an array is given (instead of a string), it should be an array of strings, which will be tried in the order they appear in the array. The first one to be a valid library on the system is used.</p> <p>Sets t:err? if unable to load the library.</p> <p>Ex: ["glib","glib-2.0"] lib glib .</p> <p>See also: G:func: w:cb</p>
libbin	-- (as of 18.06)
	<p>Similar to G:needs, but creates a new word called <name> which causes, when invoked, that item from the 'libs' to be loaded as a buffer.</p> <p>Ex: libbin fonts/generic.ttf creates a new word fonts/generic.ttf in the current namespace, which when invoked, will load the file "libs/fonts/generic.ttf" into a buffer.</p> <p>It also flags the "build" utility that it needs to copy that library file into the asset folder so that invoking the new word in a packaged application will work properly.</p> <p>Because this word is intended to allow you to incorporate binary data from libraries, it will silently ignore repeated invocations with the same name.</p> <p>See also: G:needs</p>

word	sed/description
libc	-- n
	Linux, RPI and macOS: Returns the libc handle of the "libc" system library.
literal	x --
	<p>IMMEDIATE</p> <p>Compiles the item x into the word currently being compiled. When that word is subsequently invoked, the item x will be put on TOS.</p> <p>See also: G:compile G:compile?</p>
locals:	-- (as of 17.06)
	<p>IMMEDIATE</p> <p>Flags the word about-to-be-defined as using local variables. This is <i>required</i> if the word will use w:@ or w:! .</p> <p>See also: w:@ w:!</p>
lock	x -- x
	<p>Acquire a lock on the item x . That means it will execute a tight loop until it can lock the item. If the item is already locked, it will "spin" forever until it acquires the lock. Locks are "advisory", meaning that they do not prevent other tasks accessing the item unless they cooperate and also use the locking facilities.</p> <p>See also: G:unlock G:locked? G:lock-to</p>
lock-to	x n -- x T
	<p>Same as G:lock , but takes a timeout value as a whole number n of milliseconds. If that many msec pass without being able to acquire the lock, it returns false ; otherwise it acquires the lock and returns true .</p> <p>See also: G:unlock G:locked? G:lock</p>
locked?	x -- x T
	<p>Queries the lock state of the item x , returning true if it is currently locked or false otherwise.</p> <p>See also: G:unlock G:lock-to G:lock</p>
log	s --

word	sed/description
	Prints to the "log". That will be the controlling terminal, in a console app (or one which has a console), and to the system log facility (unless false log-syslog was invoked). Logging is performed on a separate task.
log-syslog	T -- (as of 21.02)
	If false , prevents G:log from also logging to the system log facility.
log-task	T -- (as of 18.07)
	If T is true , makes the G:log word print the task invoking it.
log-time	T -- (as of 17.08)
	If T is true , makes G:log print the time when it was invoked (the actual printing of the log message is asynchronous).
log-time-local	T -- (as of 18.01)
	Makes G:log print the time it was invoked in the local time, if true ; otherwise, it will use GMT. The default is local (e.g. machine) time.
long-days	-- v
	<p>A var containing a map, mapping language names (e.g. "en") to an array of long names of the week-days. The default contains the English names of the week-days. It may be set using the appropriate "lang/xx" asset.</p> <p>See also: s:intl s:lang G:curlang G:long-months G:short-days G:short-months</p>
long-months	-- v
	<p>A var containing a map, mapping language names (e.g. "en") to an array of long names of the months. The default contains the English names of the months. It may be set by the appropriate "lang/xx" asset.</p> <p>See also: s:intl s:lang G:curlang G:long-days G:short-days G:short-months</p>
longjmp	n -- (as of 20.01)
	<p>Performs a non-local jump to the word given to G:setjmp . The number must be one returned by G:setjmp .</p> <p><i>Note:</i> you can <i>only</i> do a non-local jump within the same task!</p> <p>See also: G:setjmp</p>

word	sed/description
lookup	m s -- x a n -- x s m -- x n a -- x (as of 20.08)
	<p>Same as G:caseeof , but does not try to dereference a word from the container.</p> <p>See also: G:case G:caseof</p>
loop	w low hi --
	<p>Invoke the word w , hi - low + 1 times, counting up from low to hi , inclusive. The SED of the invoked w is n -- , where n is the current iteration number. If low is greater than hi , no iteration will occur.</p> <p><i>Note:</i> the loop index is a singleton, so if you want to store it, clone it first.</p> <p>See also: G:times G:loop- G:repeat G:while</p>
loop-	w low hi --
	<p>Same as G:loop , but in the reverse direction, e.g. from hi to low .</p> <p><i>Note:</i> the loop index is a singleton, so if you want to store it, clone it first.</p> <p>See also: G:times G:loop G:repeat G:while</p>
map?	x -- x T
	Returns true if x is a map, false otherwise.
mark	x T -- x (as of 18.05)
	<p>If true , flags the item as "marked", otherwise removes the flag. Useful for keeping track of whether an item has been seen or used.</p> <p>See also: G:mark?</p>
mark?	x -- x T (as of 18.05)
	<p>Returns the "marked" flag of the item.</p> <p>See also: G:mark</p>
mobile?	-- T
	true if running a mobile platform, false otherwise.

word	sed/description
n#	n --
	<p>Set the accuracy, e.g. the number of significant digits kept after certain big floating-point calculations. It affects at least: n:sqrt , n:exp , n:ln , n:sin , n:cos , n:tan , n:asin , n:acos , n:atan , n:atan2 , n:^ . Default is 32.</p> <p>See also: G:##</p>
name>os	s -- n
	<p><i>needs utils/os-names</i></p> <p>Convert an OS name to an os value.</p>
name>sem	s -- x (as of 16.11)
	<p>Returns the semaphore of the name created by G:sem , or null if it doesn't exist.</p> <p>See also: G:sem>name G:sem</p>
ndrop	a1 a2... aN N --
	<p><i>needs stack/utils</i></p> <p>Drop 'n' items from the data stack</p>
needs	--

word	sed/description
	<p>Parses the white-space-delimited string <name> , and evaluates the file it references. The library is searched for in this order:</p> <ol style="list-style-type: none"> 1. in the "libs" asset, then 2. app:datadir , then 3. the directory pointed to by the "EIGHTHLIB" environment variable, then 4. app:8thdir . <p>The file searched for may have no extension or ".8th". The preferred library naming convention is to omit an '.8th' extension (as all 8th's libraries are so named), but an '.8th' extension will be searched for if the no-extension version isn't found.</p> <p>Notes:</p> <ul style="list-style-type: none"> • only includes the library once. • throws an exception if it cannot load the library. • use G:(needs) if you require a library name having spaces in it. • if your library has a private word named lib-init then it will be invoked automatically by G:needs just after the library has been loaded. • any words in a private namespace are not-findable after the library has loaded. • the "with-list" is restored to what it was prior to invoking G:needs . • likewise, the current namespace. <p>See also: f:include app:asset G:.needs f:slurp G:private G:public G:requires</p>
needs-throws	-- v (as of 24.03)
	Variable which, if true , means that needs throws if the needed library can't be loaded. Defaults to true .
needs[] -- (as of 22.02)
	Same as G:needs but operates on several libraries at once, e.g. needs[lib1 lib2 lib3] .
new	n -- x
	<p>Create a new item x from the given namespace number, which may be obtained by invoking the namespace name (for example, ns:n for a number). If n is <i>not</i> a valid identifier (that is, less than 0 or greater than the largest namespace identifier), then null is returned.</p> <p>If the item being created is a "user-defined" item, e.g. from a namespace not built-in to 8th, then the word init is invoked with the new item on TOS. When refcounted to zero, the word deinit will be invoked prior to destruction.</p>
next-arg	-- x

word	sed/description
	<i>needs utils/args-common</i> Gets the next argument after the currently processed one.
nip	a b -- b
	Drop the item under TOS. See also: G:tuck G:drop G:dup G:swap G:over
noop	--
	Literally: "do nothing".
not	T -- T
	Convert true to false and vice versa. Interprets numbers which are non-zero as true for the purpose of this conversion. See also: G:and G:or G:xor
nothrow	w -- x (as of 20.05)
	Invokes the word while forcing exceptions to be ignored. If there were any exceptions during the execution of w , the exception will be on TOS; otherwise null . This differs from G:catch in that exception handling is completely bypassed.
ns	ns --
	Make the namespace identified by the number or string ns the current one. If ns does not represent a valid namespace, nothing happens. See also: G:ns:
ns:	--
	Use <name> as the current namespace. That namespace is the where new words will be created. The default namespace is user . See also: G:ns
ns>ls	n -- s

word	sed/description
	<p>Convert the namespace identifier to its <i>long</i> string representation.</p> <p>See also: G:ns>s G:ns G:ns:</p>
ns>s	n -- s s -- s'
	<p>Convert a namespace identifier, to its <i>short</i> string representation. That is, it converts the number given by ns:n to the string s , or returns null if it does not correspond to a known namespace.</p> <p>It may also be given a string, e.g. ns:n .</p>
ns?	-- s (as of 16.01)
	Returns the name of the currently active namespace.
null	-- null
	Put the value null on TOS. This is an item whose value is indeterminate. It is often returned from words as a way of indicating there is no valid value to return.
null;	x -- x null --
	If x is null , drop and exit the word; otherwise, continue execution. Used within a word instead of the phrase null? if drop ;; then .
null?	x -- x T
	Returns true if the item is null , or false otherwise.
nullvar	-- (as of 22.07)
	<p>IMMEDIATE</p> <p>Same as G:var but initializes the variable to null instead of 0 .</p> <p>See also: G:! G:@ G:var</p>
number?	x -- x T
	Returns true if x is a number, false otherwise.
of:	x -- (as of 19.03)

word	sed/description
	<i>needs utils/case</i> Used with G:case: to enumerate cases to be handled. Parses up to "of," and evaluates that into a <i>word</i> for caseof
off	v -- (as of 17.08)
	Sets the var to the numeric value 0.
on	v -- (as of 17.08)
	Sets the var to the numeric value -1, equivalent to "all bits set".
onexit	w --
	Adds the word to the list of words to be executed, in reverse order, upon program termination.
only	n --
	Makes the namespace designated by the number n the only one which the interpreter will search for words. This should be used inside an application where you wish to give user access to eval under controlled conditions. If null is passed for n , then all namespaces are available again (that is, only is cancelled).
op!	v w -- (as of 17.01)
	Invokes w on the contents of the var, replacing its current contents. Any operands to w should appear in their normal stack order under v . Effectively the same as over @ swap w:exec swap ! .
or	T T -- T
	Returns the boolean "OR" of the top two items on the stack. See also: G:and G:not G:xor
os	-- n

word	sed/description														
	<p>Returns a number indicating the operating system:</p> <table><tr><th>n</th><th>operating system</th></tr><tr><td>0</td><td>Linux</td></tr><tr><td>1</td><td>Windows</td></tr><tr><td>2</td><td>macOS</td></tr><tr><td>3</td><td>Android</td></tr><tr><td>4</td><td>iOS</td></tr><tr><td>5</td><td>Raspberry Pi</td></tr></table> <p>See also: G:os>name</p>	n	operating system	0	Linux	1	Windows	2	macOS	3	Android	4	iOS	5	Raspberry Pi
n	operating system														
0	Linux														
1	Windows														
2	macOS														
3	Android														
4	iOS														
5	Raspberry Pi														
os-names	-- v (as of 17.01)														
	<p>Returns a var containing an array of names of OSes 8th supports, in order of the return value of G:os .</p> <p>See also: G:os</p>														
os>long-name	n -- s														
	<p><i>needs utils/os-names</i></p> <p>Convert an OS value to the OS long name.</p>														
os>name	n -- s (as of 16.12)														
	<p>Returns a string corresponding to value as returned from G:os . Invalid values of will return null .</p> <p>See also: G:os G:os-names</p>														
over	a b -- a b a														
	<p>"Duplicate" the item underneath TOS and make it TOS.</p> <p>See also: G:2over</p>														
p:	--														

word	sed/description
	IMMEDIATE Postpone the action of the next word. Ensures the next word is <i>compiled</i> , rather than run immediately. Only really affects "immediate" words. See also: G:l: G:i: G:i;
pack	a s -- b

word	sed/description																																																		
	<p>Takes the items in the array a and packs them according to the format string s , returning a buffer with the binary representation of the packed array items.</p> <p>If a is null , returns a buffer large enough to hold data as indicated by the format string s .</p> <p>The format string may have any of:</p> <table><tr><th>fmt</th><th>description</th></tr><tr><td>[0-9]*</td><td>repeat count</td></tr><tr><td>b</td><td>byte (as buffer)</td></tr><tr><td>B</td><td>byte (as one-byte number)</td></tr><tr><td>c</td><td>character (string)</td></tr><tr><td>d</td><td>(8-byte IEEE) double</td></tr><tr><td>f</td><td>(4-byte IEEE) float</td></tr><tr><td>h</td><td>hex bytes (reverse hex dump)</td></tr><tr><td>i</td><td>4 byte integer</td></tr><tr><td>I</td><td>BE i</td></tr><tr><td>l</td><td>8 byte integer</td></tr><tr><td>L</td><td>BE l</td></tr><tr><td>p</td><td>pointer to a string or buffer</td></tr><tr><td>P</td><td>pointer (to a 'p' item: see ptr:pack - keep the ptr while its value is being used</td></tr><tr><td>s</td><td>count byte (etc)</td></tr><tr><td>w</td><td>2 byte integer</td></tr><tr><td>W</td><td>BE w</td></tr><tr><td>x</td><td>ignore byte</td></tr></table> <p>Size overrides:</p> <table><tr><th>char</th><th>description</th></tr><tr><td>-</td><td>short int</td></tr><tr><td>=</td><td>int</td></tr><tr><td>+</td><td>long int</td></tr><tr><td>&</td><td>void *</td></tr><tr><td>*</td><td>use the full length of the buffer or string</td></tr><tr><td>:</td><td>if present separates the "repeat count" from the "number of bytes" count</td></tr></table> <p>Ex: 3:2c means repeat a string of two bytes, three times</p> <p>See also: G:unpack</p>	fmt	description	[0-9]*	repeat count	b	byte (as buffer)	B	byte (as one-byte number)	c	character (string)	d	(8-byte IEEE) double	f	(4-byte IEEE) float	h	hex bytes (reverse hex dump)	i	4 byte integer	I	BE i	l	8 byte integer	L	BE l	p	pointer to a string or buffer	P	pointer (to a 'p' item: see ptr:pack - keep the ptr while its value is being used	s	count byte (etc)	w	2 byte integer	W	BE w	x	ignore byte	char	description	-	short int	=	int	+	long int	&	void *	*	use the full length of the buffer or string	:	if present separates the "repeat count" from the "number of bytes" count
fmt	description																																																		
[0-9]*	repeat count																																																		
b	byte (as buffer)																																																		
B	byte (as one-byte number)																																																		
c	character (string)																																																		
d	(8-byte IEEE) double																																																		
f	(4-byte IEEE) float																																																		
h	hex bytes (reverse hex dump)																																																		
i	4 byte integer																																																		
I	BE i																																																		
l	8 byte integer																																																		
L	BE l																																																		
p	pointer to a string or buffer																																																		
P	pointer (to a 'p' item: see ptr:pack - keep the ptr while its value is being used																																																		
s	count byte (etc)																																																		
w	2 byte integer																																																		
W	BE w																																																		
x	ignore byte																																																		
char	description																																																		
-	short int																																																		
=	int																																																		
+	long int																																																		
&	void *																																																		
*	use the full length of the buffer or string																																																		
:	if present separates the "repeat count" from the "number of bytes" count																																																		

word	sed/description																																				
parse	n -- s x -- s																																				
	<p>Parse a string <text> from the input stream. If n is a number, then parse until the character whose ASCII (not Unicode!) value is 'n' is reached. If x is a string or a regex, parse until that string or regex is matched. The matched item is excluded from the resultant string.</p> <p>See also: G:parsews G:parseIn G:parsech</p>																																				
parse-csv	m -- m (as of 20.01)																																				
	<p>Takes a map with information about what kind of parsing to do. Keys are:</p> <table><tr><th>key</th><th>type</th><th>description</th><th>default</th></tr><tr><td>cols</td><td>n</td><td>expected number of columns</td><td>8</td></tr><tr><td>comment</td><td>n,s</td><td>comment character if at start of line</td><td>#</td></tr><tr><td>first</td><td>w</td><td>REQUIRED: opens or initialize the read, returns first line to parse</td><td></td></tr><tr><td>ffld</td><td>T</td><td>If true, first row is field names</td><td>false</td></tr><tr><td>next</td><td>w</td><td>REQUIRED: reads next line to be parsed</td><td></td></tr><tr><td>quote</td><td>n,s</td><td>character used for quoting fields</td><td>"</td></tr><tr><td>row</td><td>w</td><td>REQUIRED: invoked with result array for each row</td><td></td></tr><tr><td>sep</td><td>n,s</td><td>field separator ("," etc)</td><td>,</td></tr></table> <p>The "first" and "next" words have the SED m -- m s , and s may be null to indicate end of data. They may store information in the map which is the same one used to create the parser.</p> <p>The "row" word has the SED m a -- m , where the array has string values. It is up to the application code to interpret the received values.</p> <p>A line beginning with the "comment" character is ignored. "cols" is an optimization so 8th only allocates an array of the expected size (instead of a much smaller one, and then resizing).</p> <p>If ffld is true , then the first row of the file contains the field names, which are then added as an array to the map passed in, using the key "fields" .</p>	key	type	description	default	cols	n	expected number of columns	8	comment	n,s	comment character if at start of line	#	first	w	REQUIRED: opens or initialize the read, returns first line to parse		ffld	T	If true, first row is field names	false	next	w	REQUIRED: reads next line to be parsed		quote	n,s	character used for quoting fields	"	row	w	REQUIRED: invoked with result array for each row		sep	n,s	field separator ("," etc)	,
key	type	description	default																																		
cols	n	expected number of columns	8																																		
comment	n,s	comment character if at start of line	#																																		
first	w	REQUIRED: opens or initialize the read, returns first line to parse																																			
ffld	T	If true, first row is field names	false																																		
next	w	REQUIRED: reads next line to be parsed																																			
quote	n,s	character used for quoting fields	"																																		
row	w	REQUIRED: invoked with result array for each row																																			
sep	n,s	field separator ("," etc)	,																																		
parse-date	T -- (as of 22.08)																																				
	<p>Controls whether or not the interpreter, if it fails to parse a number, attempts to then parse a date. The default is true , meaning that 8th will attempt a date parse if it fails to parse a number.</p>																																				
parsech	-- s																																				

word	sed/description
	<p>Parse the next non-whitespace character from the input stream into a string one character long.</p> <p>See also: G:parsews G:parseIn G:parse</p>
parseIn	-- s
	<p>Parse <text> from the input stream until the end of the line.</p> <p>See also: G:parse G:parsews G:parsech</p>
parsews	-- s
	<p>Parse <text> from the input stream, skipping leading whitespace, until the following whitespace. The resultant string will not contain any whitespace.</p> <p>See also: G:parse G:parseIn G:parsech</p>
pick	n -- x
	<p>Duplicates the item x at position n in the stack, putting it on TOS.</p> <p>TOS is 0, item under is 1, etc. If you try to pick beyond the limits of the stack, an exception will be thrown.</p> <p>See also: G:over G:rot G:rpick</p>
poke	x y --
	<p><i>needs stack/utils</i></p> <p>Pokes the value 'y' into position x on the stack, numbered as TOS=0, before the parameters are added. So: 10 20 30 40 2 200 poke will give: 10 200 30 40.</p>
pool-clear	n -- (as of 17.03)
	<p>Walks the pool for the given namespace identifier, and frees the entire "free" list. This might be used, for example, after having used a large number of items (e.g. a big array of numbers) and releasing it. If you know the application will not require a similar number of items in the future, you can invoke pool-clear to release the memory used by those unused items.</p>
pool-clear-all	-- (as of 19.05)
	<p>Clears all pools in the current task. Same as G:pool-clear iterated over all the pools.</p>
prior	w1 -- w2

word	sed/description
	<p>If a word with the same name as an existing word was created in the same namespace, this returns the prior version of that word, which would otherwise be inaccessible by name. If there is no prior word it returns null .</p>
private	-- (as of 18.05)
	<p>Puts the following words or vars in the #p namespace, which is intended for private use. When used in a library or file included using G:needs or f:include , the words or vars created in that private namespace will be <i>invisible</i> after that file is finished processing. That is, w:find will not find them.</p> <p>See also: G:public G:needs f:include</p>
process-args	m --
	<p><i>needs utils/args</i></p> <p>Pass it a map containing the argument name as a key, and the action to perform as its value, e.g.: { "-h" : ' do-help } If the action needs to grab the next argument to use, it should use next-arg to get the next argument to process. An argument which is not in the map will cause the processing to stop. All remaining (non-processed) arguments can be retrieved with **remaining-args **. The processor of an argument can issue break which will stop further processing.</p>
process-args-fancy	m T -- m (as of 21.05)
	<p><i>needs utils/args-fancy</i></p> <p>Pass it a map containing the argument name as a key, whose value is another map with 'cmd' -- the action to perform or 'var' -- the variable to put the 'nextarg' in also 'msg' -- help text to display if an unknown option is given.</p> <p>The map's keys are entered into a trie so any unique initial prefix of options will work. The T value is true if you want the options to ignore case.</p> <p>The following options are added unless they already exist: '--' 'stop processing' '-h' and '-?' 'display help'</p> <p>The map returned contains any 'var' options processed.</p>
process-args-help	-- (as of 21.05)
	<p><i>needs utils/args-fancy</i></p> <p>Invoke the default argument-processor's help screen, which displays help text based on the 'msg' key for each option.</p>
prompt	s -- s'

word	sed/description
	<p>DEFERRED</p> <p>word which returns the REPL's ok> prompt. The string it is passed contains spaces or one of the following: "\"", "{", "[", or "+" to indicate which of those items is currently still open. It returns the prompt which will be displayed.</p> <p>See also: dbg:prompt</p>
public	-- (as of 18.05)
	<p>Restores the namespace which was current prior to the previous G:private invocation.</p> <p>See also: G:private G:needs f:include</p>
putc	n --
	<p>DEFERRED</p> <p>Print the character n , which is a number representing the Unicode character to print. Default behavior is G:(putc) .</p> <p>See also: G:puts G:. G:(putc)</p>
puts	s --
	<p>DEFERRED</p> <p>Print the string s . Default behavior is G:(puts) .</p> <p>See also: G:putc G:. G:(puts)</p>
quote	-- s
	<p>IMMEDIATE</p> <p>Parses one character from the input stream, then parses the rest of the input stream up to the next instance of that character, returning a string. This is useful if you have embedded double-quotes, for instance. Note: the standard 8th "string-escapes" (for example "\n") are not processed by G:quote .</p> <p>See also: G:"</p>
r!	x -- (as of 18.04)
	<p>Replaces the r-stack TOS with the data-stack's TOS item. Equivalent to rdrop <blockquote> <p> r .</p> <p>See also: G:r@ G:>r G:r></p>
r>	-- x

word	sed/description
	<p>Pop the item x from the r-stack and push it onto the data-stack.</p> <p>See also: G:>r G:r@ G:rswap G:rdrop</p>
r@	-- x
	<p>Pushes the top of the r-stack onto the data-stack, without removing it.</p> <p><i>Note:</i> this is <i>not</i> the same as r:@ , which is used to access a regex match!</p> <p>See also: G:>r G:r> G:rswap G:rdrop</p>
rad>deg	n -- n'
	<p>Converts from radians to degrees.</p> <p>See also: G:deg>rad</p>
rand-jit	-- n (as of 19.08)
	<p>Returns a random number from the "Jitter RNG".</p> <p>See also: cr:rand G:rand-pcg</p>
rand-jsf	-- n (as of 19.08)
	<p>Generates a 64-bit pseudo-random number using the "JSF" PRNG.</p>
rand-native	-- n (as of 19.08)
	<p>Generates a 64-bit pseudo-random number using the OS-specific entropy provider. Not guaranteed to be available, cryptographically secure, or fast.</p>
rand-normal	mean sigma -- n (as of 19.08)
	<p><i>needs rand/normal</i></p> <p>Returns a Gaussian "normal" distributed random number given a mean and standard-deviation sigma desired. How good the distribution is depends on the random number generator chosen.</p>
rand-pcg	-- n

word	sed/description
	<p>Generate a 64-bit pseudo-random number using the "PCG" generator. This is much faster than cr:rand , but is not cryptographically random. The PCG state is task-specific and therefore task-safe.</p> <p>See also: cr:rand G:rand-pcg-seed G:rand-jsf G:rand-jit</p>
rand-pcg-seed	n n2 -- null --
	<p>Sets the 'seed' for the PCG PRNG to the given numbers. Using null instead will make the PRNG choose a random seed via cr:rand .</p> <p>See also: G:rand-pcg</p>
rand-range	low high -- n (as of 19.08)
	<p><i>needs rand/range</i></p> <p>Returns a random integer in the range [low,hi]</p>
rand-select	s -- (as of 19.08)
	<p><i>needs rand/util</i></p> <p>Sets whether G:random is cr:rand, G:rand-pcg, G:rand-native, or G:rand-jit. The default is G:rand-pcg. The choices for s are:</p> <p>"crypto" -- selects rand</p> <p>"pcg" -- selects rand-pcg, the default</p> <p>"jsf" -- selects rand-jsf,</p> <p>"jitter" -- selects rand-jit, a very slow CPU-jitter based one, and</p> <p>"native" -- selects rand-native, the OS-native entropy provider</p>
randbuf-pcg	n -- b
	<p>Generates a buffer of size n filled with pseudo-random bytes from the PCG random generator.</p> <p>See also: G:randbuf</p>
random	-- n (as of 19.08)
	<p>DEFERRED</p> <p>Returns a random number, initially from rand-pcg , for speed. Can be changed manually to another random number word, or you can use the rand-select word (from the library rand/util) to set it symbolically.</p>

word	sed/description
rdrop	--
	Drop TOS of the "r-stack". Same effect as G:drop but for the "r-stack". See also: G:>r G:>r G:r@ G:swap
recurse	--
	IMMEDIATE Invoke the last word created (e.g. G:last at the time this was compiled).
recurse-stack	n -- (as of 17.05)
	Set the "recursion stack size" (e.g. the hardware stack for return addresses). n must be a multiple of the system page size. Only takes effect for new tasks, not the main task.
ref@	x -- x n
	Returns the "reference-count" n of the item x . This is what's shown when invoking .s . See also: G:+ref G:-ref
reg!	s x -- T
	Sets the value of a Windows Registry key. The value may be a string, a number, or a buffer. Returns true if succeeded, false otherwise. <i>Note: Windows only!</i>
reg@	s -- x
	Gets the value of a Windows Registry key. Returns null if the value does not exist. <i>Note: Windows only!</i>
regbin@	s -- b n
	Get the value of a Windows Registry key, as a buffer containing whatever data is in the key, and a number which indicates the kind of item (the value for kind is the same as what the Windows RegQueryValueEx() returns). Returns null if the value does not exist. <i>Note: Windows only!</i>
remaining-args	-- v (as of 18.06)

word	sed/description
	<p><i>needs utils/args-common</i></p> <p>A <i>var</i> which contains an <i>array</i> of the remaining arguments to be processed. Once process-args has returned, contains arguments which have not been processed, either because break was invoked during processing, or because the argument was not known.</p>
repeat	--
	<p>IMMEDIATE</p> <p>Begin a repeated block of code terminated with either again or while . May only be used in compile-mode, throws an exception otherwise.</p> <p>See also: G:again G:while G:while! G:break G:break?</p>
requires	-- (as of 19.09)
	<p>Ensures that support for the required resources is loaded. "" is a space-separated string which may contain any of: nk snd con bt . The resource is only loaded once.</p> <p>If the required resource isn't present (requires bt , if bt:init doesn't exist in the SKU or if it returns false) then a message is thrown (because "requiring" means you really need that item to function).</p> <p>The thrown message has an explanation of what exactly happened to cause a failure.</p>
reset	--
	Remove all items from the data stack.
roll	n --
	<p>Moves the item at position n in the stack to the TOS, and moves all other items down the stack; in effect, rotating the items. TOS is position 0, item under is 1, etc. If you try to roll beyond the limits of the stack, it will throw an exception.</p> <p>2 roll has the same effect as G:rot . If ix is negative, roll takes the TOS and puts it at position ix , moving items up the stack. So -2 roll is equivalent to G:-rot .</p> <p>See also: G:rot G:-rot G:roll</p>
rop!	w -- (as of 18.04)
	<p>Invokes w on the TOS of the r-stack, using parameters from the data-stack, replaces its TOS.</p> <p>See also: st:op!</p>
rot	a b c -- b c a

word	sed/description
	<p>Move the item in the third position to TOS.</p> <p>See also: G:-rot G:swap G:over G:drop G:nip G:tuck</p>
rpick	<p>n -- x</p> <p><i>needs stack/rstack</i></p> <p>Returns the item at position n in the "r-stack". TOS is 0, item under is 1, etc. Same as G:pick , but for the "r-stack".</p> <p>See also: G:pick</p>
rreset	<p>-- (as of 23.08)</p> <p>Equivalent of G:reset , but for the r-stack.</p> <p>See also: G:reset</p>
rroll	<p>n -- st</p> <p><i>needs stack/rstack</i></p> <p>Same as G:roll but for the r-stack.</p> <p>See also: G:roll st:roll</p>
rstack	<p>-- st</p> <p>Returns an item which references the "r-stack". This permits you to operate on the r-stack using any stack word.</p> <p>See also: G:dstack</p>
rswap	<p>{x y -- y x}</p> <p>Same as G:swap , but for the r-stack</p> <p>See also: G:swap</p>
rusage	<p>-- m</p> <p>Returns a map containing the values of various resources used by the running program. Keys may be: "rss", "swap", "fault", "ixrss", "isrss", "load1", "load5", "load15", "idrss".</p> <p>The meanings of the values are OS-specific.</p>

word	sed/description
s>ns	s -- n (as of 18.08)
	Returns the numeric id of the named namespace.
same?	x y -- T
	<p>Compares the two items x and y for literal equality, e.g. whether or not they represent the same actual item. So:</p> <ul style="list-style-type: none"> • 123 dup same? is true • 123 123 same? is false • 123 clone same? is false <p>See also: n:= s:=</p>
scriptdir	-- s
	Returns the directory from which the script is running. Only applies when the script is sourced from the command-line, unlike G:scriptfile
scriptfile	-- a
	<p>Returns an array containing the full path name of the script being run as well as scripts leading up to it (e.g. via f:include or G:needs). It is empty if no script is run.</p> <p>Index -1 is the currently running script.</p>
sem	s n -- X
	<p>Returns an named IPC semaphore, with an initial count of n . Returns null if it cannot create or open the semaphore.</p> <p>See also: G:sem-post G:sem-wait G:sem-wait?</p>
sem-post	X --
	<p>Increments the count of the semaphore returned from G:sem , so another task or process using G:sem-wait can continue.</p> <p>See also: G:sem G:sem-wait G:sem-wait?</p>
sem-rm	X -- (as of 18.08)

word	sed/description
	<p>Removes the named semaphore (created by G:sem) from the system. The actual removal will be deferred until all processes using it are finished.</p> <p>See also: G:sem</p>
sem-wait	X --
	<p>Waits until the semaphore's count becomes non-zero, then continue.</p> <p>See also: G:sem G:sem-post G:sem-wait?</p>
sem-wait?	X -- T
	<p>Returns true if G:sem-wait would succeed, false if it would block.</p> <p>See also: G:sem G:sem-post G:sem-wait</p>
sem>name	X -- X s (as of 16.11)
	<p>Returns the name used when creating the semaphore with G:sem', or null` if it doesn't exist.</p> <p>See also: G:sem G:name>sem</p>
semi-throw	T -- (as of 17.08)
	<p>If false , suppresses the exception probable missing ; or) in ... which is thrown if a script file leaves the interpreter expecting more input. The default is true .</p>
set-wipe	sb -- sb
	<p>Sets "wipe" flag on a string or a buffer. That ensures that when the item's refcount has gone to zero, the data it holds is wiped before being released. This means the user does not have to manually use clear . Of particular use in cryptographic contexts.</p> <p>See also: s:clear b:clear</p>
setenv	var val --
	<p>Set the value val of the environment variable var . If val is null , clears the value from the environment.</p> <p>See also: G:getenv</p>
setjmp	w -- n (as of 20.01)

word	sed/description
	<p>Creates a non-local jump point. The return value is a number passed subsequently to G:longjmp when it's desired to branch to the word.</p> <p><i>Note: you can only do a non-local jump within the same task!</i></p> <p>See also: G:longjmp</p>
settings!	key val --
	<p><i>needs utils/settings</i></p> <p>Save the key,val pair in the settings database.</p>
settings![]	[key,val,...] --
	<p><i>needs utils/settings</i></p> <p>Save the <i>array</i> of key,val pairs in the settings database.</p>
settings@	key -- val
	<p><i>needs utils/settings</i></p> <p>Get the setting named by the string 'key', and return its value or 'null' Note the use of the r-stack to transfer the result!</p>
settings@?	key default -- val
	<p><i>needs utils/settings</i></p> <p>Get a settings value and if it is null, use 'default' instead.</p>
settings@[]	[key,...] -- [val,...]
	<p><i>needs utils/settings</i></p> <p>Get the values corresponding to an <i>array</i> of keys in the settings database.</p>
sh	s w --
	<p>Execute the external command string, invoking the word w when the process has finished. If w is null, the app will <i>not</i> be informed when the command finishes. Execution is asynchronous. The SED for w is n --, where n is the OS return code from the shelled process.</p> <p>See also: l:sh G:sh\$</p>
sh\$	s w --

word	sed/description
	<p>Same as G:sh , except that the word w is given the string result from the shell invocation. The SED for w is n s -- , where n is the OS return code from the shelled process, and s is the string output (to stdout).</p> <p>See also: l:sh G:sh</p>
short-days	-- v
	<p>A var containing a map, mapping language names (e.g. "en") to an array of short names of the week-days. The default contains the English names of the week-days. It may be set by the appropriate "lang/xx" asset.</p> <p>See also: s:intl s:lang G:curlang G:long-days G:long-months G:short-months</p>
short-months	-- v
	<p>A var containing a map, mapping language names (e.g. "en") to an array of short names of the months. The default contains the English names of the months. It may be set by the appropriate "lang/xx" asset.</p> <p>See also: s:intl s:lang G:curlang G:long-days G:long-months G:short-days</p>
sleep	n --
	<p>Sleeps the current task for that number of seconds, which does not have to be an integer. Fractional seconds are broken into milliseconds. A negative number means 'wait forever', or until t:notify wakes it up. So 1.5 sleep will sleep for approximately 1500 milliseconds.</p> <p>See also: t:notify t:wait t:q-wait</p>
sleep-msec	n -- (as of 23.05)
	<p>Same as G:sleep , but n is milliseconds, and doesn't respond to wakes; will sleep the entire time it's given.</p>
sleep-until	d -- (as of 19.08)
	<p><i>needs utils/sleep</i></p> <p>Sleep the current task until the <i>date</i> d. Contast with G:sleep which sleeps for a number of seconds. NOTE: will respond to t:notify by waking up perhaps prematurely.</p>
slog	x s -- (as of 21.02)
	<p>Same effect as s:strfmt log .</p>
space	--

word	sed/description
	<p>Print a space character, ASCII 32.</p> <p>See also: G:. G:cr G:puts G:putc s:strfmt</p>
stack-check	T -- (as of 16.13)
	<p>Determines whether or not type and bounds-checking are performed on stack operations. The default is true , and should only be changed once you have completely debugged your application.</p>
stack-size	n --
	<p>Sets the main data-stack size to the given number n . This is useful if you will need more than 128K items on the stack (but if you <i>do</i> need more than 128K items on the stack, you should probably reconsider your code!). It will not affect the data-stack size for callback words. The stack size for tasks is set using t:def-stack .</p> <p>It also resizes the current task's data-stack and moves data if necessary to the resized stack.</p>
step	n --
	<p>Increment the current loop counter by n , which may be positive or negative. Works with loop and loop- , but not with times . Quits if the index is beyond the original loop limits.</p> <p>See also: G:loop G:loop- a:map a:map= a:2map+ a:map+ a:each a:each! a:2each a:x-each a:x a:reduce+ s:map s:each s:each! b:each b:each!</p>
sthrow	x s -- (as of 20.05)
	<p>Like G:slog , but throws.</p>
string?	x -- x T
	<p>Returns true if x is a string, false otherwise.</p>
struct:	-- str
	<p><i>needs utils/structs</i></p> <p>Begin definition of a "struct", which is actually a <i>map</i> containing field and type information about the struct. The struct ends with struct; .</p> <p>See also: struct:struct;</p>
swap	a b -- b a

word	sed/description
	<p>Exchange position of top two items on the stack.</p> <p>See also: G:2swap G:rot G:-rot</p>
tab-hook	s -- a
	<p>DEFERRED</p> <p>This is the word responsible for console tab-completion. It receives a string which is the text entered in the console <i>before</i> the TAB key was pressed. Returns an array of strings (possibly empty) matching the initial string.</p> <p>This word is lazy-loaded on first use in the REPL, and is implemented in the utils/tab-hook library.</p>
tell-conflict	-- v
	<p>Returns a var which controls whether or not 8th will complain when overwriting an existing word. It contains true by default.</p>
tempdir	-- s
	<p>Returns the directory where temporary files will be created.</p> <p>See also: G:tempfilename</p>
tempfilename	-- s
	<p>Returns a string suitable for a temporary file name in the directory G:tempdir .</p> <p>See also: G:tempdir</p>
then	--
	<p>IMMEDIATE</p> <p>Ends a conditional if... else... then block. May only be used in compile-mode, throws an exception otherwise. Also throws an exception if there was no accompanying if .</p> <p>See also: G:if G:else</p>
third	a b c -- a b c a (as of 17.08)
	<p>Copies the third item on the stack to TOS. Same as 2 pick .</p>
throw	x --

word	sed/description
	<p>Throw an exception. Quits the current word and passes x to handler . x may be any type: if it is a string it will be displayed as-is; otherwise, it will be converted to a string for display. If you throw a number, its value must be greater or equal to 1000. The default behavior prints a message, logs it to the system log, and quits the application, unless 8th is running from the console rather than a file or application.</p> <p>In the latter case, 8th does not quit, on the assumption that you made a typo or other error, and the stack is left in an unpredictable state; so use reset or just ignore the stack state.</p> <p>See also: G:handler</p>
thrownull	x y -- x
	<p>If x is null , then throw the item y . Otherwise, continue (with x on TOS). This should be used when a null result is a fatal error.</p>
times	w n --
	<p>Invoke the w n times. If n is zero or negative, w will not be invoked.</p> <p>See also: G:loop G:loop- G:repeat G:while</p>
tlog	s --
	<p><i>needs debug/log</i></p> <p>Same as log, but prepends a date-time stamp, including milliseconds, to the output.</p>
toggle	v -- T (as of 24.02)
	<p>Toggles the value in the var from true to false and vice-versa. Returns the previous value of the var.</p>
tri	x w1 w2 -- w1(x) w2(x)
	<p><i>needs combinators/tri</i></p> <p>Implementation of Factor's "tri" combinator. Invokes w1 and w2 and w3 on x, leaving the results.</p>
true	-- T
	<p>Returns the boolean value true .</p> <p>See also: G:false</p>
tuck	a b -- b a b

word	sed/description
	<p>Duplicate the item on TOS under the item under TOS.</p> <p>See also: G:nip G:drop G:dup G:swap G:over</p>
type-check	T --
	<p>If T evaluates false , prevents 8th from checking data types in the built-in words. This increases performance but also the risk of unexpected failures. The default is true , meaning items are type-checked by builtin words.</p>
typeassert	n1 n2 -- (as of 17.06)
	<p>DEFERRED</p> <p>Invoked by the system when G:type-check is true and the type checking determines the types are not correct. Passed the namespace identifiers of the item and the expected namespace.</p> <p>The default behavior is to throw an exception saying what is wrong, e.g. Expected %s but got %s . You can modify the behavior to check conditions and perhaps fix the parameters.</p>
uid	x -- x s (as of 21.03)
	Returns a string uniquely identifying the item.
uname	-- m (as of 19.03)

word	sed/description																										
	<p>Returns a map with information about the system:</p> <table> <tr> <th>key</th><th>description</th></tr> <tr> <td>machine</td><td>hardware name</td></tr> <tr> <td>nodename</td><td>machine name on local network</td></tr> <tr> <td>numpages</td><td>total memory pages</td></tr> <tr> <td>numproc</td><td>number of CPU cores</td></tr> <tr> <td>numproc_conf</td><td>number of CPU cores configured</td></tr> <tr> <td>pagesavail</td><td>available memory pages</td></tr> <tr> <td>pagesize</td><td>system memory pagesize (usually 4096 or more)</td></tr> <tr> <td>proctype</td><td>CPU version</td></tr> <tr> <td>release</td><td>kernel version or Windows release</td></tr> <tr> <td>sysname</td><td>OS name</td></tr> <tr> <td>uptime</td><td>How long since the machine was rebooted</td></tr> <tr> <td>version</td><td>kernel version or Windows release</td></tr> </table> <p><i>Note:</i> not all are available on every platform.</p>	key	description	machine	hardware name	nodename	machine name on local network	numpages	total memory pages	numproc	number of CPU cores	numproc_conf	number of CPU cores configured	pagesavail	available memory pages	pagesize	system memory pagesize (usually 4096 or more)	proctype	CPU version	release	kernel version or Windows release	sysname	OS name	uptime	How long since the machine was rebooted	version	kernel version or Windows release
key	description																										
machine	hardware name																										
nodename	machine name on local network																										
numpages	total memory pages																										
numproc	number of CPU cores																										
numproc_conf	number of CPU cores configured																										
pagesavail	available memory pages																										
pagesize	system memory pagesize (usually 4096 or more)																										
proctype	CPU version																										
release	kernel version or Windows release																										
sysname	OS name																										
uptime	How long since the machine was rebooted																										
version	kernel version or Windows release																										
unlock	x -- x																										
	<p>Release the lock on the item x . Note that any task can lock or unlock any item! That means that the locking model is <i>cooperative</i>, and you are responsible to make sure the task which locks is the same one which unlocks (or conversely, that another task doesn't unlock before testing the lock state).</p> <p>See also: G:locked? G:lock-to G:lock</p>																										
unpack	sb s -- a n sb n s -- a n																										
	<p>Takes the items in the string or buffer sb and "unpacks" them according to the format string s , returning the number of <i>bytes</i> processed, and an array containing the interpreted data from the buffer.</p> <p>If a number is provided under the format string and above the buffer, it indicates an offset in bytes from the beginning of the buffer, from which to start unpacking. Format string same as for G:pack .</p> <p>See also: G:pack</p>																										
until	x -- x (as of 18.03)																										
	<p>Same effect as not while .</p> <p>See also: G:until! G:while G:while!</p>																										

word	sed/description
until!	x --
	<p>Same effect as not while! .</p> <p>See also: G:until G:while G:while!</p>
var	--
	<p>IMMEDIATE</p> <p>Create a var named <name> , initialized with the number "0". Invoking the <i>name</i> will put the var on TOS; you can then retrieve data using G:@ or store data using G:! . Note that var as well as var, may only be invoked in interpret mode (that is, not inside a colon-def or anonymous word).</p> <p>See also: G:! G:@ G:var,</p>
var,	x --
	<p>IMMEDIATE</p> <p>Same as G:var except that the var is initialized with the value "x".</p> <p>See also: G:! G:@ G:var</p>
while	x -- x
	<p>IMMEDIATE</p> <p>At runtime, peeks at TOS; if true , returns to the most recent repeat .</p> <p><i>Note:</i> this does <i>not</i> pop TOS! If you want that behavior, use G:while! instead.</p> <p><i>Only</i> available in compile-mode, throws an exception otherwise. It also throws an exception if you forgot to invoke repeat before it. A "while loop" may be stopped with ;; (which immediately exits the word), or break (which prevents the loop from repeating).</p> <p>See also: G:again G:repeat G:while! G:break G:break?</p>
while!	x -- (as of 16.02)
	<p>IMMEDIATE</p> <p>The same as G:while , but pops x from TOS.</p> <p>See also: G:while</p>
with:	--

word	sed/description
	<p>IMMEDIATE</p> <p>Add the namespace <name> to the "with list", which is the list of namespaces which will be searched. Matched by ;with . The namespace name should not have the "ns:" prefix, e.g. use with: n to add the number namespace to the with list.</p> <p>See also: G;;with</p>
word?	x -- x T (as of 20.01)
	Returns true if the item is a word.
words	--
	<p>DEFERRED <i>needs utils/words</i></p> <p>List all currently known words, alphabetically by <i>ns</i></p>
words-like	s -- a null -- a
	<p>Like G:words/ , takes a filter string, but returns an array of strings containing the fully-qualified names of words matching the filter. If s is null , returns all words.</p> <p>See also: G:words/ G:words</p>
words/	--
	<p>DEFERRED <i>needs utils/words</i></p> <p>Like words, but matches only words which match the <i>string</i> "". If the filter ends with a colon, will only display words in that namespace</p>
xchg	x v -- x' (as of 17.03)
	Puts the item x into the var v , and returns the item which it previously held.
xor	T T -- T
	<p>Returns the boolean "EXCLUSIVE OR" of the top two items on the stack.</p> <p>See also: G:or G:not G:and</p>

Graph

Namespace: **gr**

Description: Graph data type: nodes, edges, weights

word	sed/description
+edge	gr n1 n2 n3 -- gr (as of 18.08)
	<p>Adds an edge to the graph. The parameters n1 and n2 are the numeric indices of the nodes (the indices are the positions in the nodes array of the particular node). n3 is the direction, as for gr:new, and if it is non-zero, then the 'directed' parameter of gr will become true.</p> <p>See also: gr:+edge+w</p>
+edge+w	gr n1 n2 n3 n4-- gr (as of 18.08)
	<p>Same as gr:+edge, but lets you specify the weight as well. Using this word will override any "weight" word the graph would otherwise use.</p> <p>See also: gr:+edge</p>
+node	gr x -- gr (as of 18.08)
	<p>Adds a node to the graph's list of nodes.</p>
connect	gr -- gr (as of 18.08)
	<p>Connects each node to <i>every other node</i> in the graph, creating an edge for each combination. If a weight word was set, it is used to determine the edge weight for each pair. If a threshold value was specified, then only edges with weights greater than (if above is true) or less than (if above is false) will be added to the graph.</p>
edges	gr -- gr a (as of 18.08)
	<p>Returns an array with all the edges of the graph. Each 'edge' is an array: [from, to, weight, direction].</p> <p><i>Note:</i> modifying the returned array will NOT modify the nodes in the graph; to do that, use gr:edges!</p>
edges!	gr a -- gr (as of 20.05)
	<p>Sets the edges of the graph. Each entry in the array must contain [from, to, direction, weight].</p>
m!	gr s x -- gr (as of 18.08)

word	sed/description																		
	Associates the value x with the key s in the graph.																		
m@	gr s -- gr x (as of 18.08)																		
	Returns the value associated with the key s in the graph, or null if there is no such value.																		
neighbors	gr ix -- gr a (as of 18.08)																		
	Returns an array with all the nodes which share an edge with the node with index ix .																		
new	m -- gr (as of 18.08)																		
	<p>Creates a new graph defined by the map. The keys in m can be:</p> <table> <tr> <th>key</th><th>description</th></tr> <tr> <td>weight</td><td>word comparing two nodes and returning a weight (defaults to no weight)</td></tr> <tr> <td>directed</td><td>boolean (defaults to false)</td></tr> <tr> <td>nodes</td><td>array of nodes (any 8th data item; defaults to nothing)</td></tr> <tr> <td>edges</td><td>array of edge descriptors, which are each an array [start-edge, end-edge, weight, direction]</td></tr> <tr> <td>threshold</td><td>number which the weight must be above (if "above") or below to include the edge</td></tr> <tr> <td>above</td><td>boolean if true, the calculated weight must be above the threshold to include it</td></tr> <tr> <td>map</td><td>map which has values specific to this graph</td></tr> <tr> <td>autoconnect</td><td>boolean, if true connect up all nodes to each other</td></tr> </table> <p>In the case of an unweighted graph, 'weight' is 0. 'direction' is 0 for undirected, 1 source->dest, 2 dest->source, 3 source<->dest.</p> <p>m can also be null, to indicate an empty, unweighted, undirected, graph.</p>	key	description	weight	word comparing two nodes and returning a weight (defaults to no weight)	directed	boolean (defaults to false)	nodes	array of nodes (any 8th data item; defaults to nothing)	edges	array of edge descriptors, which are each an array [start-edge, end-edge, weight, direction]	threshold	number which the weight must be above (if "above") or below to include the edge	above	boolean if true , the calculated weight must be above the threshold to include it	map	map which has values specific to this graph	autoconnect	boolean, if true connect up all nodes to each other
key	description																		
weight	word comparing two nodes and returning a weight (defaults to no weight)																		
directed	boolean (defaults to false)																		
nodes	array of nodes (any 8th data item; defaults to nothing)																		
edges	array of edge descriptors, which are each an array [start-edge, end-edge, weight, direction]																		
threshold	number which the weight must be above (if "above") or below to include the edge																		
above	boolean if true , the calculated weight must be above the threshold to include it																		
map	map which has values specific to this graph																		
autoconnect	boolean, if true connect up all nodes to each other																		
node-edges	gr ix -- gr a (as of 18.08)																		
	Returns an array with all the edges connected to the node with index ix .																		
nodes	gr -- gr a (as of 18.08)																		
	<p>Returns an array with all the nodes of the graph, in the order inserted.</p> <p><i>Note:</i> modifying the returned array will modify the nodes in the graph, so be cautious.</p>																		
traverse	gr T ix w -- gr (as of 18.08)																		

word	sed/description
	<p>Traverse the given graph starting at the "root node" given by the ix . If T is true, the traversal will be "depth-first", e.g. continue down child nodes as far as possible before searching sibling nodes. Otherwise, it will traverse siblings first.</p> <p>The word w is invoked for each node visited, with a SED of gr x ix -- gr , where x is the node and ix is its index.</p> <p>The traversal continues until there are no more nodes which can be visited, or G:break was invoked in w . After the traversal has completed, you may iterate the nodes list to see which have been traversed, by invoking G:mark? on the node.</p>
weight!	gr w -- gr (as of 20.08)
	Assigns the word as the graph's weight function.

GUI

Namespace: **g**

Description: GUI Widgets and utilities

word	sed/description
media?	n -- s (as of 24.04)
	<p><i>needs gui/media</i></p> <p>Returns one of the following values, if the value passed in (in pixels) is below the threshold (and above the preceding one). The values correspond to the media query values used by "Pure CSS", as expanded using typical display resolution standards.</p> <p>Use the return value to decide UI layout parameters.</p>

Hardware

Namespace: **hw**

Description: Hardware abstraction layer

word	sed/description
arm?	-- T

word	sed/description
	Returns true if running on an ARM CPU.
camera	fmt -- hw
	Takes an array from the "fmts" member of one of the camera maps returned from hw:camera? , and requests the use of that camera from the OS. Returns a hw representing the camera on success, or null if it could not get use of the camera.
camera-img	hw -- hw img
	Grabs an image from the camera hw, or null if no image was available. Not currently on mobile.
camera-limits	hw n -- hw a (as of 20.05)
	Given a property from the enumerations in the "hw/camera" library, returns an array of the minimum, maximum, and default values for the property. If the property isn't available for the given hardware, or the camera has not been opened, returns null .
camera?	-- a
	Queries the system for information on available cameras. Returns an array of maps, one per detected camera. Because of "plug-n-play", the list of attached cameras may change, so it is necessary to invoke hw:camera? soon before hw:camera (unless one knows the camera list won't change in the meantime). The map returned per camera has the "name", a "uuid", the camera's "index" in the original array, and an array called "fmts", which contains a list of the video resolution and formats the camera can produce. Each "fmts" array contains, in order, the format's width, height, "fourcc" format code, FPS, the camera index of the camera it belongs to, and the format index in the "fmts" *array.
cpu?	-- m

word	sed/description																										
	<p>Returns a map which identifies the properties of the CPU. Possible values returned:</p> <table> <tr> <th>key</th><th>description</th></tr> <tr> <td>aes</td><td>has "AES" instructions</td></tr> <tr> <td>avx</td><td>has "AVX" instructions</td></tr> <tr> <td>avx2</td><td>has "AVX2" instructions</td></tr> <tr> <td>le</td><td>Is the CPU "little-endian"</td></tr> <tr> <td>mmx</td><td>has "MMX" instructions</td></tr> <tr> <td>neon</td><td>has "neon" instructions</td></tr> <tr> <td>sse</td><td>has "SSE" instructions</td></tr> <tr> <td>sse2</td><td>has "SSE2" instructions</td></tr> <tr> <td>sse3</td><td>has "SSE3" instructions</td></tr> <tr> <td>sse4.1</td><td>has "SSE4.1" instructions</td></tr> <tr> <td>sse4.2</td><td>has "SSE4.2" instructions</td></tr> <tr> <td>vendor</td><td>"ARM" or "Intel"</td></tr> </table>	key	description	aes	has "AES" instructions	avx	has "AVX" instructions	avx2	has "AVX2" instructions	le	Is the CPU "little-endian"	mmx	has "MMX" instructions	neon	has "neon" instructions	sse	has "SSE" instructions	sse2	has "SSE2" instructions	sse3	has "SSE3" instructions	sse4.1	has "SSE4.1" instructions	sse4.2	has "SSE4.2" instructions	vendor	"ARM" or "Intel"
key	description																										
aes	has "AES" instructions																										
avx	has "AVX" instructions																										
avx2	has "AVX2" instructions																										
le	Is the CPU "little-endian"																										
mmx	has "MMX" instructions																										
neon	has "neon" instructions																										
sse	has "SSE" instructions																										
sse2	has "SSE2" instructions																										
sse3	has "SSE3" instructions																										
sse4.1	has "SSE4.1" instructions																										
sse4.2	has "SSE4.2" instructions																										
vendor	"ARM" or "Intel"																										
device?	-- s																										
	Returns a string which is a description of the device on which 8th is currently running.																										
displays?	-- n																										
	Returns the number of displays (monitors) attached to this device.																										
displaysize?	n -- wide high																										
	Returns the dimensions of display number n : from 0 to (hw:displays? - 1), in pixels.																										
finger-match	w to -- (as of 19.03)																										
	<p><i>Professional version</i></p> <p>Takes a word and a number to , which is the seconds to allow for the system to scan fingerprints. "w" gets invoked with results (a map) containing a "msg" which is text explaining the value, and "code", a number which is 0 on successful match. If there is no hardware support, the code is -1; if the match times-out the code is -2; other codes are system specific. The callback is executed asynchronously.</p> <p>Android only.</p>																										

word	sed/description
finger-support	-- m (as of 19.03)
	<p><i>Professional version</i></p> <p>Returns a map with two keys: "hw" is true if the device has fingerprint detection hardware, false otherwise; "prints" is true if fingerprints have been registered on the device.</p> <p>Android only for now...</p>
gpio	n T -- X
	<p><i>Hobbyist version</i></p> <p>Creates a new GPIO item which will be used subsequently for I/O on the GPIO pin n . If T is true , reads from the; otherwise write to it.</p> <p>See also: hw:gpio@ hw:gpio! hw:gpio-mmap</p>
gpio!	X n --
	<p><i>Hobbyist version</i></p> <p>Writes to the GPIO pin (returned from hw:gpio). That value is evaluated as: anything which would be true evaluates as '1' (e.g. "on") and anything which would be false evaluates as '0'.</p> <p>See also: hw:gpio hw:gpio@ hw:gpio-mmap</p>
gpio-mmap	n --
	<p><i>Hobbyist version</i></p> <p>Determines how to access GPIO. If passed "0", it turns off the memory-map accessors for GPIO (the default is to use the OS accessors). If it is "1", it tries to use the Raspberry Pi specific memory-map accessors.</p> <p><i>Note:</i> If the hardware you are running on is <i>not</i> RPI, bad things might happen. You've been warned!.</p> <p>See also: hw:gpio@ hw:gpio! hw:gpio</p>
gpio@	X -- X m
	<p><i>Hobbyist version</i></p> <p>Reads from the GPIO pin (returned from hw:gpio) and returns null if it is unable to do so, or the value of the pin, which will be either '0' (for "off") or '1' (for "on").</p> <p>See also: hw:gpio hw:gpio! hw:gpio-mmap</p>
i2c	bus addr -- hw

word	sed/description
	<p><i>Hobbyist version</i></p> <p>Given two numbers bus and addr , representing the bus-id and the device-address, returns a hw describing the opened I2C device, or null if it could not.</p> <p><i>Note:</i> this functionality is currently only available on Linux, Android, and Raspberry Pi, and requires "root" (or appropriate "group") access.</p> <p>See also: hw:i2c@ hw:i2c@reg hw:i2c! hw:i2c!reg</p>
i2c!	hw b -- hw n
	<p><i>Hobbyist version</i></p> <p>Writes the data in the buffer to the I2C device from hw:2ic . Returns the number of bytes written or null if there was a failure.</p> <p>See also: hw:i2c hw:i2c@ hw:i2c@reg hw:i2c!reg</p>
i2c!reg	hw reg val T -- hw T
	<p><i>Hobbyist version</i></p> <p>Writes the number val to the I2C device from hw:2ic register reg . If T evaluates true , writes one byte; otherwise writes a word (two bytes). The returned value T will be true if the data could be written, or false otherwise.</p> <p>See also: hw:i2c hw:i2c@ hw:i2c@reg hw:i2c!</p>
i2c@	hw n -- hw b
	<p><i>Hobbyist version</i></p> <p>Reads n bytes from the hw returned from hw:i2c . Conversion of that data may be done with G:unpack , if needed. See hw:i2c for caveats regarding this word.</p> <p>See also: hw:i2c hw:i2c@reg hw:i2c! hw:i2c!reg</p>
i2c@reg	hw reg T -- hw n
	<p><i>Hobbyist version</i></p> <p>Reads from the hw returned from hw:i2c , at register reg . If T evaluates true , reads one byte; otherwise reads a word (two bytes). The returned value will be null if the data could not be read, or a number with the data otherwise.</p> <p>See also: hw:i2c hw:i2c@ hw:i2c! hw:i2c!reg</p>
isround?	-- T (as of 18.06)

word	sed/description
	<i>Professional version</i> Android wear OS only: returns true if the device has a round profile (e.g. a round-face watch). Otherwise (and on all other OSes) returns false .
iswatch?	-- T (as of 18.06)
	<i>Professional version</i> Android: returns true if the device is a 'watch', or false otherwise.
mac?	-- a
	Returns this computer's network-cards' MAC addresses as an array of strings.
mem?	-- n
	Returns the amount of RAM installed as a number in MB.
model?	-- s (as of 19.04)
	Returns a string containing the model of the board (RPI etc); or null if unable to determine.
poll	hw -- m
	Poll the current state of the hw sensor hw , returning a map whose keys specify the data for that device. Ex: a "gps" device will return "lat", "lon" and "alt" keys, at least; and a "gyro" will return "x", "y", and "z"; and a "compass" will return "head" and "true". If the device is not available or has no data, null will be returned. See also: hw:sensor
sensor	s -- hw
	Returns a new hw corresponding to the sensor type giving in the string s . Currently, that may be one of: <ul style="list-style-type: none"> • gps : GPS location • gyro : 3D attitude information • compass : Direction (N,S, etc) • accel : An accelerometer Returns null if it failed to get that kind of sensor, or a hw item which may be polled, etc. See also: hw:start hw:poll hw:stop
start	hw --

word	sed/description
	<p>Activate the hw sensor which was created using hw:sensor . Retrieve data from the sensor using hw:poll .</p> <p>See also: hw:sensor</p>
stop	hw --
	<p>Stop the hw sensor from requesting updates from the hardware.</p> <p>See also: hw:sensor</p>
uid?	-- s (as of 19.05)
	Returns a unique hardware UUID, or null if that is not possible

Heap

Namespace: **h**

Description: Ordered container

word	sed/description
+	h h2 -- h
	Moves all items from the heap h2 to the heap h1 . Both heaps are modified.
clear	h -- h
	Remove all items from the heap.
cmp!	h w -- h (as of 20.08)
	Assigns the word as the heap's comparator function.
len	h -- h n
	Returns the number of items on the heap.
max!	h T -- h (as of 19.06)

word	sed/description
	If T is true , sets the heap to return the "maximum" value (this is the default); otherwise, it will return the "minimum". This must be done <i>before</i> items are pushed into the heap, once it is created.
new	w -- h
	<p>Create a new heap using the word w as the heap's comparator. The SED for w is x1 x2 -- n where n is negative if x1 is less than x2 , 0 if they're equal, and positive if greater.</p> <p>See also: h:push h:pop h:peek</p>
peek	h -- h x
	<p>Like h:pop , but does not remove the item from the heap.</p> <p>See also: h:pop h:peek</p>
pop	h -- h x
	<p>Returns the "largest" item x from the heap h while removing it, or return null if the heap is empty. The meaning of "largest" is determined by the heap comparator word passed it on heap creation.</p> <p>See also: h:push h:peek</p>
push	h x -- h
	<p>Push the item x onto the heap.</p> <p>See also: h:pop h:peek</p>
unique	h T -- h (as of 18.05)
	If true , makes the heap "unique", which means that only <i>unique</i> entries will be accepted. In that case, pushing a non-unique item is a no-op. By default, heaps are "not unique".

html

Namespace: **html**

Description: html

word	sed/description
parse	s -- DOM (as of 21.09)
	<i>needs dom/html-parse</i>
	Parses a string containing HTML, returning a DOM

HTTP

Namespace: **HTTP**

Description: HTTP

word	sed/description
>auth	username password -- authstring
	<i>needs net/basic-auth</i>
	Given a username and password as <i>strings</i> , return another <i>string</i> which is an appropriately crafted string for "HTTP basic authentication".

Image

Namespace: **img**

Description: JPEG, PNG and BMP images

word	sed/description
>file	img s --
	Write the image to a file named by the string s . s should have an extension of ".png", ".bmp", ".jpg", ".jpeg", or ".tga". 8th will save the image in the corresponding format. The default is PNG format if no extension matches. Sets t:err?
>fmt	img n -- img' (as of 20.01)

word	sed/description																						
	<p>Takes the source img and creates a new one with the given format. The number is one of:</p> <ul style="list-style-type: none"> • 1: 256-color greyscale • 2: 256-color greyscale + 256 value alpha • 3: 256-color each RGB • 4: 256-color each RGBA 																						
copy	img dest src -- img (as of 18.01)																						
	<p>Similar to img:scroll , but copies a sub-section of the source img from src to dest . dest is an array of numbers which are: ["top","left"]. The src element is an array of numbers which are ["top", "left", "wide", "high"].</p> <p><i>Note:</i> the original img itself is modified!</p>																						
crop	img x y wide high -- img' (as of 16.12)																						
	<p>Returns a sub-image taken from the starting point (x , y) for a rectangle wide by high pixels. If the dimensions are outside the boundaries of the source img, returns null .</p>																						
data	img T -- img m (as of 17.03)																						
	<p>Returns a map containing:</p> <table> <tr> <th>key</th><th>description</th></tr> <tr> <td>data</td><td>a buffer with the raw pixel data</td></tr> <tr> <td>stride</td><td>the "line stride", number of bytes between lines</td></tr> <tr> <td>pixstride</td><td>the number of bytes between pixels</td></tr> <tr> <td>width</td><td>the img width</td></tr> <tr> <td>height</td><td>the img height</td></tr> <tr> <td>fmt</td><td>a number indicating the img format:</td></tr> <tr> <td></td><td>0: unknown</td></tr> <tr> <td></td><td>1: 3-byte RGB</td></tr> <tr> <td></td><td>2: 4-byte ARGB</td></tr> <tr> <td></td><td>3: 1-byte</td></tr> </table> <p>The T parameter should be true if you don't intend to modify the image data, false if you do.</p>	key	description	data	a buffer with the raw pixel data	stride	the "line stride", number of bytes between lines	pixstride	the number of bytes between pixels	width	the img width	height	the img height	fmt	a number indicating the img format:		0: unknown		1: 3-byte RGB		2: 4-byte ARGB		3: 1-byte
key	description																						
data	a buffer with the raw pixel data																						
stride	the "line stride", number of bytes between lines																						
pixstride	the number of bytes between pixels																						
width	the img width																						
height	the img height																						
fmt	a number indicating the img format:																						
	0: unknown																						
	1: 3-byte RGB																						
	2: 4-byte ARGB																						
	3: 1-byte																						
desat	img -- img' (as of 16.12)																						
	<p>Desaturate (convert to grayscale) the img.</p>																						

word	sed/description
draw	img rect img2 -- img (as of 22.04)
	Draws img2 within the rectangle on img .
draw-sub	img pt img2 rect -- img (as of 22.06)
	Draws the subimage at rect of img2 at position pt in img .
fill	img clr -- img (as of 17.07)
	Fill the img with the color.
fillrect	img x y w h clr -- img (as of 20.01)
	Fill the given img with the color within the rectangle, modifying the img.
filter	img a -- img' img m -- img' (as of 16.12)
	Apply the convolution matrix defined by the array or map to the img. The array should have a size which is a perfect square; it will be treated as if it is (so if it has for example 10 numbers, the first 9 will be used to make a 3x3 convolution kernel). If a map is given, it must contain a key "mat" which is the matrix to use, and it may optionally contain the keys "mode" which is a number: 0 means ignore edge (the default); 1 = wrap edge; 2 = extend edge; 3 = treat edge as transparent black; and "normalize" which is a number to which the sum of the values in the matrix will be adjusted. So "normalize": 1 will make the matrix sum to 1 so that the overall brightness of the image is not affected.
flip	img T -- img' (as of 16.13)
	Returns an img which is the first one flipped end for end (mirrored). If T is true , then it flips horizontally; otherwise, if flips vertically.
from-svg	sb -- img m -- img

word	sed/description																														
	<p>Loads an SVG image. If passed a string or buffer, then if it begins with "<" it is treated as SVG XML; otherwise, it is treated as the name of a file containing SVG.</p> <p>If passed a map, then it may contain any of these keys (one of 'file' or 'svg' is required):</p> <table><tr><th>key</th><th>description</th><th>default</th></tr><tr><td>dpi</td><td>Dots per inch</td><td>96</td></tr><tr><td>file</td><td>Name of the file to read the SVG from</td><td></td></tr><tr><td>high</td><td>Height of the image to create in pixels</td><td>in the SVG</td></tr><tr><td>ofsx</td><td>X offset to apply after scaling</td><td>0</td></tr><tr><td>ofsy</td><td>Y offset to apply after scaling</td><td>0</td></tr><tr><td>scale</td><td>Amount to scale the image by</td><td>1.0</td></tr><tr><td>svg</td><td>Buffer or string containing svg</td><td></td></tr><tr><td>units</td><td>Default image units ("px", "pt", "pc", "mm", "cm", "in")</td><td>"px"</td></tr><tr><td>wide</td><td>Width of the image to create in pixels</td><td>in the SVG</td></tr></table>	key	description	default	dpi	Dots per inch	96	file	Name of the file to read the SVG from		high	Height of the image to create in pixels	in the SVG	ofsx	X offset to apply after scaling	0	ofsy	Y offset to apply after scaling	0	scale	Amount to scale the image by	1.0	svg	Buffer or string containing svg		units	Default image units ("px", "pt", "pc", "mm", "cm", "in")	"px"	wide	Width of the image to create in pixels	in the SVG
key	description	default																													
dpi	Dots per inch	96																													
file	Name of the file to read the SVG from																														
high	Height of the image to create in pixels	in the SVG																													
ofsx	X offset to apply after scaling	0																													
ofsy	Y offset to apply after scaling	0																													
scale	Amount to scale the image by	1.0																													
svg	Buffer or string containing svg																														
units	Default image units ("px", "pt", "pc", "mm", "cm", "in")	"px"																													
wide	Width of the image to create in pixels	in the SVG																													
line	img x0 y0 x1 y1 clr -- img img pt0 pt1 clr -- img (as of 22.04)																														
	Draws a line on the image from pt0 to pt1 in the given color.																														
new	s -- img b -- img img -- img' wide high -- img																														
	<p>Load an img. Allowable formats are PNG, GIF, JPEG, and SVG.</p> <p>If given a:</p> <ul style="list-style-type: none">• string: it refers to a file name• buffer: it is the raw image data• img: it is another image, in which case a copy of that image is made.• numbers: creates a new blank img wide x high pixels.																														
pikchr	s class flags -- s w h s class flags -- null (as of 22.04)																														
	Parses "pikchr" code and returns a corresponding SVG string and its width and height, or null if the parse failed.																														
pix!	img row col clr -- img																														
	<p>Set the pixel color in the image at (row,col) . The original image is modified.</p> <p>See also: meta:color img:pix@</p>																														
pix@	img row col -- img n																														

word	sed/description
	<p>Returns the pixel color at (row,col)</p> <p>See also: meta:color img:pix!</p>
qr-gen	sb n -- m (as of 17.07)
	<p><i>Hobbyist version</i></p> <p>Generates a map with a QR-code from a string or buffer, with ECC level n . The map's keys are "size" and "data", where "data" is an array "size" X "size" of ones and zeroes describing the QR-code.</p>
qr-parse	img -- a img -- n (as of 16.13)
	<p><i>Hobbyist version</i></p> <p>Scans a black and white img for QR codes and returns null if there are none, or an array containing a string with the value encoded, or a number containing an error code. The possible error codes are:</p> <ul style="list-style-type: none"> • 1: Invalid grid size • 2: Invalid version • 3: ECC format error • 4: Data error • 5: Unknown data type • 6: Data overflow • 7: Data underflow <p>See also: img:qr-gen</p>
rect	img x0 y0 x1 y1 clr -- img img pt0 pt1 clr -- img (as of 22.04)
	<p>Draws a rectangle on the image from pt0 to pt1 in the given color.</p>
rotate	img n -- img' (as of 16.13)
	<p>Returns an img which is the first one rotated by n degrees.</p>
scale	img wide high -- img'
	<p>Resize the image to wide by high pixels.</p>
scroll	img n1 clr n2 -- img (as of 18.01)

word	sed/description
	<p>Scrolls the source img, by moving the image data n2 rows or columns of pixels. n1 is the direction to move: 0 for 'up', 1 for 'right', 2 for 'down' and 3 for 'left'.</p> <p>The clr parameter is the value which will be used to fill in vacated pixels. If it is null, then the color at the edge will be used to fill (for each row or column as appropriate).</p> <p><i>Note:</i> the original img itself is modified!</p>
size	img -- img wide high
	Returns width and height of the image in pixels.

IMAP

Namespace: **imap**

Description: IMAP access words

word	sed/description
fetch-full	imap ix uid (r: arr) -- imap (r: arr)
	<p><i>needs net/imap</i></p> <p>Fetch an IMAP message into the array on the rstack.</p>
fetch-uid-mail	imap uid -- imap result
	<p><i>needs net/imap</i></p> <p>Fetch an IMAP message by uid.</p>
login	imap -- imap flag
	<p><i>needs net/imap</i></p> <p>Log-in to the given <i>imap</i>. Returns true on success, or fail otherwise.</p>
logout	imap -- imap
	<p><i>needs net/imap</i></p> <p>Log-out of the IMAP connection. After logout, the connection "hangs", so parse-response is not possible</p>
new	m -- imap

word	sed/description
	<p><i>needs net/imap</i></p> <p>Connects to an IMAP server using the options given in the <i>map</i>. Returns an 'imap' which is used for further communications:</p> <ul style="list-style-type: none"> • debug: required boolean • port: number, if not given, defaults to 143 for IMAP with or without STARTTLS+IMAP or 993 for IMAP over SSL • pwd: required • secure: required string, one of 'none', 'tls', 'ssl' • server: required string, the name or ip address of the server to connect to • user: required
search	imap s-query -- imap A
	<p><i>needs net/imap</i></p> <p>Search IMAP e-mail and return array of e-mail objects.</p>
select-inbox	imap -- imap stats
	<p><i>needs net/imap</i></p> <p>Returns the IMAP inbox as a <i>map</i>.</p>

ISO

Namespace: **iso**

Description: Various ISO things

word	sed/description
countries	-- m (as of 18.06)
	<p><i>needs iso/countries</i></p> <p>Returns a <i>map</i> containing (abbrev, name) pairs with the current ISO 3166-1 alpha-2 country codes and their expanded names and codes. Each two-character code (e.g. "US") key's value is a <i>map</i> containing at least "a3" (the alpha-3 code), "num", the ISO numeric code, "name" (the usual country name), "name2" (the "official" name if there is one) and "name3", the "common name", if there is one.</p>
languages	-- m (as of 22.03)

word	sed/description
	<i>needs iso/languages</i> Returns a <i>map</i> whose keys are the two-letter ISO language names, and the contents are maps of (name, native); where 'name' is the English name of the language, and 'native' is the name of the language for native speakers.

JSON-RPC

Namespace: **JSONRPC**

Description: JSON-RPC access words

word	sed/description
call	rpc opts -- result
	<i>needs net/json-rpc</i> Make a JSON-RPC call.

Location

Namespace: **loc**

Description: Description of a location's information

word	sed/description
bearing	a1 a2 -- n (as of 24.03)
	<i>needs geo/bearing</i> Calculates the great-circle bearing between two geographic locations. Given two arrays containing [latitude, longitude] , returns the bearing in degrees from north, from location a1 to location a2 .
find	loc --
	<i>needs geo/location</i> Find the given location loc by name. Tries both the English and Hebrew names. Returns a <i>map</i> with the location information, or null if it failed. The search is not case-sensitive.
sort	hebrew? --

word	sed/description
	<i>needs geo/location</i> Sort the locations collection by name. Use the Hebrew location name ("loch") for sorting if hebrew? evaluates true ; otherwise, use the English name ("loc").

Map

Namespace: **m**

Description: Maps are unordered containers accessed by a string key

word	sed/description
!	m s x -- m m n x -- m
	Put the item x in the map or user-defined type using the indicated key. Normally, the key is a string; however, if it is a number it will be converted to a string first as if >s had been invoked on it. If the key is anything but a string or a number, then the item itself will be used as a key, and that item will be returned with m:keys or m:each for instance. The string displayed then will start with "(item)". This allows using arbitrary items as keys in a map, but it is not JSON friendly. See also: m:@
!?	m s x -- m (as of 16.10)
	Sets the value for the key, if there isn't already a value for that key.
+	m1 m2 -- m1
	Insert the keys and their current values from map m2 into map m1 , modifying m1 .
+?	m m2 -- m (as of 17.05)
	The same as m:+ , but only adds those keys in m2 which are not already present in m . See also: m:+
-	m s -- m
	Remove the key s and its associated item from the map m , modifying it.
↔	m s1 s2 -- m (as of 22.07)

word	sed/description
	Swap the values of the keys "s1" and "s2" in the map. Ex: <code>{a:12, b:54} "a" "b" m:<></code> returns <code>{a:54, b:12}</code> . Both keys must exist in the map.
=	m1 m2 w -- m1 m2 T (as of 22.08)
	Compares the two maps, using the word to determine equality. The maps are considered equal if they have the same number of keys, and each key's value is equal to the same key's value in the other map (including its type), according to the comparator word. That word's SED is <code>x1 x2 -- T</code> .
>arr	m -- a (as of 21.03)
	Converts the map into an array of <code>[[key1,val1], [key2,val2]...]</code> . Inverse of <code>m:arr></code> . See also: m:arr>
@	m s -- m x m n -- m x m a -- m a'
	Returns the item <code>x</code> in the map or user-defined type for the key, which is usually a string, but may also be any other item. If the key is a number, it is first converted to a string as if <code>>s</code> had been invoked on it. If the key is an array, it contains keys whose values in the map are returned in an array. If the key is any other kind of item, it is first converted to a string starting with "(item)" and looked up. If a key doesn't exist in the map, <code>null</code> is returned. See also: m:!
@?	m s x -- m x' m a x -- m a' (as of 16.10)
	Gets the value for the key in the given map, if it exists. If it doesn't exist, returns the default value <code>x</code> instead. If passed an array of keys, then <code>x</code> may be either an array of corresponding default values, or a single default value for all entries. If it is an array which has fewer items than the number of keys, the last item is used as the default for subsequent entries. See also: m:@
[]!	m s x -- m (as of 20.03)
	Same as <code>m:!</code> , but if the key already exists, then if the existing value isn't an array, converts it to one and pushes the new value into it. Basically, accumulates values into an array for that key.
_!	m x key -- m (as of 19.05)
	Same as <code>m:!</code> , but with the <code>key</code> on TOS. More convenient in many cases. See also: m:!

word	sed/description
_@	m s -- x (as of 19.07)
	<p>Same as m:@ , but removes the map from the stack.</p> <p>See also: m:@</p>
_@?	m s x -- x' m a x -- a' (as of 24.03)
	<p>Same as m:@? but discards the map.</p> <p>See also: m:@?</p>
alias	m s1 s2 -- m (as of 23.05)
	<p>Adds the key s2 to the map, pointing to the exact same item as s1 . Note: if s1 is subsequently modified, s2 will still refer to the original item (until it itself is modified).</p>
arr>	a -- m (as of 21.03)
	<p>Converts the array of [[key1,val1] , [key2,val2] ...] into a map. Inverse of m:>arr .</p> <p>See also: m:>arr</p>
bitmap	m a n -- n' (as of 19.09)
	<p>Given a map with keys whose values are numbers, and an array of strings which are some subset of those keys, returns the values ORed together in a single number. The n parameter is the default value to use if the keys aren't found, or if the array of keys was empty.</p> <p>Ex: {a:1, b:2, c:4} ["a","c"] 0 m:bitmap returns 5.</p>
clear	m -- m
	<p>Remove all elements from the map.</p>
data	m -- m'
	<p>Returns the internal map of a user-defined type. If m is not a user-defined type, returns null .</p>
each	m w -- m

word	sed/description
	<p>Iterate over the map and invoke w for each element in it. The SED of w is key x -- ; that is, it is passed the key of the item as well as the item itself, and must consume both.</p> <p>The map is not available on the stack while m:each is running, to avoid consistency problems. Modifying the map while it is being iterated may throw an exception.</p>
exists?	m s -- m T m a -- m T
	<p>Returns true if there is a value defined for the key in the map or user-defined type m . This is necessary because requesting an arbitrary key from a map will always succeed, returning null if the key does not exist as well as if the value stored is actually null .</p> <p>An array of strings may be passed, in which case each of them which must exist as a key with a value in the map for the return value to be true .</p>
filter	m w -- m' (as of 19.05)
	Filters a map. The SED for w is s x -- T , where s is the key name and x is its value. If T is true , the item is kept; otherwise it is not.
ic	m T -- m (as of 21.09)
	If true , set the map to be case-insensitive with respect to key values. The default is false , and this must be set before any keys have been added to the map.
iter	map keys wrd -- map
	<p><i>needs map/iter</i></p> <p>Iterate over a <i>map</i>, in the order of the <i>array</i> of keys given. Do not iterate over any other keys. 'wrd' has the SED \ map key -- map</p>
iter-all	map keys wrd -- map
	<p><i>needs map/iter-all</i></p> <p>Similar to m:iter, but first iterates over the keys in the given order, and then iterates over the remaining keys in alpha order.</p>
keys	m -- m a
	<p>Returns all the keys of the map m , as an array of strings.</p> <p>See also: m:vals</p>
len	m -- m n

word	sed/description
	Returns the length of the map, e.g. the number of keys it currently contains. Equivalent to <code>m:keys a:len nip</code> .
map	m w -- m'
	Create a map whose elements are formed by invoking <code>w</code> for each element of the original map. The SED for <code>w</code> is: <code>s x -- x'</code> , where <code>s</code> is the key affected and <code>x</code> is the current value. The return value will become the corresponding element in the new map.
merge	m1 m2 -- m1
	<i>needs map/merge</i> Merge the contents of m2 into m1. Iterates the keys of m2 (assumed to be maps), and merges into same key of m1, creating a map for the key, if needed
new	-- m (as of 17.05)
	Create a new, empty map.
op!	m s w -- m m a w -- m (as of 17.01)
	Invokes <code>w</code> on the map's contents at key <code>s</code> or array of keys <code>a</code> , replacing the current contents. Any operands to <code>w</code> should appear in their normal stack order under <code>m</code> .
open	m -- a1 a2 (as of 18.01)
	Opens the map <code>m</code> , producing two arrays containing, respectively, the keys of the map and their corresponding values. Inverse of <code>m:zip</code> . See also: m:zip
slice	m a -- m' (as of 19.05)
	Returns a map which is a "slice" of the original map consisting of the keys in the array. Items which don't exist will appear as <code>null</code> .
vals	m -- m a (as of 18.04)
	Returns all the <i>values</i> of the map, in the same order <code>m:keys</code> would give. See also: m:keys
xchg	m s x -- m x' (as of 18.04)

word	sed/description
	<p>Stores a new value in the map at the key s , placing the current value on TOS.</p> <p>See also: G:xchg a:xchg</p>
zip	a1 a2 -- m (as of 20.04)
	<p>Creates a map from two arrays, where a1 is the keys, and a2 is the values. All entries in a1 are processed. If a2 is shorter than a1 , the value null will be inserted as the value for the extra elements of a1 . Inverse of m:open .</p> <p>See also: m:open</p>

Markdown

Namespace: **md**

Description: Markdown text processing

word	sed/description
2console	s m -- f (as of 19.09)
	<p><i>needs md/2console</i></p> <p>Takes an input <i>string</i> containing Markdown formatted text and writes to the console. Returns true if it was able to parse the MD</p>
2html	m s -- s (as of 19.09)
	<p><i>needs md/2html</i></p> <p>Takes an input <i>string</i> containing Markdown formatted text, and a <i>map</i> with options, produces an HTML <i>string</i> as output. The <i>map</i> can have the keys: "css" (a <i>string</i> which is a file-name of CSS, or a <i>buffer</i> containing CSS; "num" - <i>boolean</i> which if true, means to number H tags; "frag" - <i>boolean</i> which if true means the HTML is not a whole document.</p>
2nk	m s -- f (as of 21.01)
	<p><i>needs md/2nk</i></p> <p>Takes an input <i>string</i> containing Markdown formatted text and writes to the GUI. Returns true if it was able to parse the MD</p>

Matrix

Namespace: **mat**

Description: Numbers: matrices (native double or complex, only)

word	sed/description
!	mat a n -- mat n1 n2... nN n mat -- mat (as of 18.08)
	<p>Puts n into the matrix at the location specified by the array or the series of numbers, as with mat:@ .</p> <p>See also: mat:new mat:@</p>
*	mat1 mat2 -- mat3 (as of 18.08)
	<p>Returns the result of a matrix-multiplication of the 2D input matrices. If the first matrix column dimension is not the same as the second's row dimension, null is returned.</p> <p>See also: mat:new</p>
+	m1 m2 -- m3 (as of 18.08)
	<p>Adds the input matrices. They must be of the same type and dimensions, otherwise the result is null .</p> <p>See also: mat:new</p>
=	mat1 mat2 -- T (as of 18.08)
	<p>Returns true if the two matrices have the same dimensions and elements' values.</p> <p>See also: mat:new</p>
@	mat a -- mat n n1 n2... nN mat -- mat n (as of 18.08)
	<p>Retrieves the value of the matrix at the position given by either the array or the numbers d1, d2... dN . In the first case, the size of the array must be the same as that of the matrix dimensions. In the second case, the number of numbers must equal the matrix dimensions.</p> <p>See also: mat:new mat:!</p>
affine	a -- mat (as of 20.01)

word	sed/description
	<p>Creates an affine-transformation matrix from an array of 6 numbers. If there are not 6 numbers, null is returned. Otherwise, a matrix is returned which can be used by mat:xform .</p> <p>See also: mat:transform</p>
col	mat n -- mat' (as of 18.08)
	<p>Returns the column n from the matrix, giving a new 1D matrix as long as the number of rows in the original.</p> <p>See also: mat:new</p>
data	mat -- mat a (as of 18.08)
	<p>Returns the data values of the matrix.</p> <p>See also: mat:new</p>
det	mat -- n (as of 18.08)
	<p>Returns the determinant of the matrix. If mat is not 2-dim and square, returns null .</p> <p>See also: mat:new</p>
dim?	m -- m a (as of 18.08)
	<p>Returns an array of numbers, the dimensions of the matrix.</p> <p>See also: mat:new</p>
get-n	mat ix -- mat n (as of 18.08)
	<p>Gets the number or complex at offset ix in the internal representation of mat</p> <p>See also: mat:new</p>
ident	n -- mat (as of 18.08)
	<p>Creates a 2D "identity matrix" of dimension n . The main diagonal is all "1" and the rest is all "0".</p> <p>See also: mat:new</p>
inv	mat -- mat' (as of 19.02)

word	sed/description
	<p>Returns the inverse of the matrix, or null if it is not possible to invert it. Does not currently handle inversion of complex matrices.</p> <p>See also: mat:new</p>
m.	mat -- (as of 18.08)
	<p>Pretty-prints a matrix.</p> <p>See also: mat:new</p>
minor	mat n1 n2 -- mat' (as of 18.08)
	<p>Returns the "minor" of the matrix, with column n1 and row n2 removed.</p> <p>See also: mat:new</p>
n*	mat n -- mat (as of 18.08)
	<p>Performs scalar multiplication of n and the matrix. Modifies the original matrix.</p> <p>See also: mat:new</p>
new	a1 a2 -- mat (as of 18.08)
	<p>Creates a new matrix, whose initial values are given in the array a1 (which may be null), with dimensions given in the array a2. If a1 contains fewer values than implied by a2, the remaining values will be 0. This is the built-in version of mat:new, which only supports native double (real or complex numbers). The kind of value in such a matrix must be all of the same type, e.g. either numbers or complex numbers.</p> <p>See also: mat:get-n mat:+ mat:dim? mat:same-size? mat:@ mat:! mat:n* mat:ident mat:row mat:col mat:trans mat:* mat:minor mat:det mat:= mat:m. mat:data</p>
new-minor	mat -- mat' (as of 19.02)
	<p>Returns the "matrix of minors" of the original, or null if that is not possible. The matrix of minors consists of the determinants of the minor of each position in the original.</p> <p>See also: mat:new</p>
rotate	n -- m (as of 20.01)

word	sed/description
	<i>needs math/affine</i> Returns an affine-transformation matrix rotating clockwise by n radians
row	mat n -- mat' (as of 18.08)
	Returns the row n from the matrix, giving a new 1D matrix as long as the number of columns in the original. See also: mat:new
same-size?	mat1 mat2 -- mat1 mat2 T (as of 18.08)
	Returns true if the two matrices are the same size, false otherwise. See also: mat:new
scale	x y -- m (as of 20.01)
	<i>needs math/affine</i> Returns an affine-transformation matrix scaling (x,y)
shear	x y -- m (as of 20.01)
	<i>needs math/affine</i> Returns an affine-transformation matrix shearing (x,y)
trans	mat -- mat' (as of 18.08)
	Transposes a 2D matrix with dimensions column,row into a new matrix with dimensions row,column. See also: mat:new
translate	x y -- m (as of 20.01)
	<i>needs math/affine</i> Returns an affine-transformation matrix translating (x,y)
xform	pt mat -- a a mat -- a' (as of 20.01)
	Applies the affine-transformation matrix (created by mat:affine) to the point or array of points. See also: mat:affine

Network

Namespace: **net**

Description: Internet and sockets

word	sed/description
!	net key val -- net
	<i>needs net/utls</i> Insert a key,val into the opts map of a 'net'.
!?	net key value -- net
	<i>needs net/utls</i> Same as m: !? for net
-	net key -- net
	<i>needs net/utls</i> remove the key from the net opts
>base64url	x -- s (as of 23.08)
	<i>needs net/base64url</i> Convert a buffer or string to "base64url" format
>url	s -- s'
	URL-encode the string as per RFC 3986. See also: net:url>
@	net key -- net val
	<i>needs net/utls</i> Get the 'key' from the opts map of a 'net'.
@?	net key default -- net val
	<i>needs net/utls</i> Same as net:@ but use the default if the key is empty
CGI	-- T (as of 23.01)

word	sed/description
	<p><i>needs net/cgi</i></p> <p>true if running as a CGI script</p>
DGRAM	-- n
	<p>Returns a number to pass to net:socket to use a datagram (UDP) protocol.</p> <p>See also: net:INET4 net:INET6 net:STREAM net:socket</p>
INET4	-- n
	<p>Returns a number to pass to net:socket to use internet v4 addressing.</p> <p>See also: net:INET6 net:STREAM net:DGRAM net:socket</p>
INET6	-- n
	<p>Returns a number to pass to net:socket to use internet v6 addressing.</p> <p>See also: net:INET4 net:STREAM net:DGRAM net:socket</p>
PROTO_TCP	-- n
	Returns a number to pass to the net:socket word.
PROTO_UDP	-- n
	Returns a number to pass to the net:socket word.
REMOTE_IP	-- s (as of 23.01)
	<p><i>needs net/cgi</i></p> <p>If net:CGI , then the IP address of the client; otherwise null</p>
REMOTE_IP	-- s (as of 23.01)
	<p><i>needs net/cgi</i></p> <p>Returns the IP address of the connecte client.</p>
STREAM	-- n

word	sed/description														
	<p>Returns a number to pass to net:socket to use a streaming (TCP) protocol.</p> <p>See also: net:INET4 net:INET6 net:DGRAM net:socket</p>														
accept	net -- net'														
	<p>Waits for a connection on net after invoking net:listen . When the connection occurs, returns a new socket on which to talk to the remote side.</p> <p>See also: net:bind net:listen net:connect</p>														
active?	-- m (as of 23.03)														
	<p><i>needs net/active</i></p> <p>Returns a map with a key IPV4 and/or IPV6 pointing to a map with information from net:ifaces? for those interfaces which are not just loopback etc.</p>														
addrinfo>o	X -- a														
	<p>Convert the X returned from net:getaddrinfo into an array of maps (one for each addressinfo) with the keys:</p> <table><tr><th>key</th><th>description</th></tr><tr><td>addr</td><td>either null or a valid IPv4 or IPv6 address string</td></tr><tr><td>canon</td><td>if present, a string with the canonical name of the address</td></tr><tr><td>family</td><td>string, either "V4" or "V6"</td></tr><tr><td>flags</td><td>number, various OS-dependent flags</td></tr><tr><td>proto</td><td>number which is the protocol</td></tr><tr><td>type</td><td>string, either "stream" or "dgram"</td></tr></table>	key	description	addr	either null or a valid IPv4 or IPv6 address string	canon	if present, a string with the canonical name of the address	family	string, either "V4" or "V6"	flags	number, various OS-dependent flags	proto	number which is the protocol	type	string, either "stream" or "dgram"
key	description														
addr	either null or a valid IPv4 or IPv6 address string														
canon	if present, a string with the canonical name of the address														
family	string, either "V4" or "V6"														
flags	number, various OS-dependent flags														
proto	number which is the protocol														
type	string, either "stream" or "dgram"														
again?	net -- net T (as of 17.01)														
	<p>Returns true if the last read or write on net had an error of "EAGAIN", which means "try again in a while".</p>														
alloc-and-read	net -- net buf net null														
	<p><i>needs net/utils</i></p> <p>Allocates a suitable buffer, and reads from the net into it. Set the 'wait-to' value to the number of msecs to wait for input from the remote side (default is 1000)</p>														

word	sed/description
alloc-buf	net -- net buf
	<i>needs net/utls</i> Allocate a new buffer based on the 'bufsize' the user set. This is a buffer which can be used to read into for the net.
avail?	-- T (as of 24.05)
	<i>needs net/avail</i> Returns true if there is internet connectivity, uses a DNS query to determine status.
base64url>	s -- b (as of 23.08)
	<i>needs net/base64url</i> Convert a string from "base64url" format to a buffer
bind	X net --
	Binds the net to the X returned from net:getaddrinfo . See also: net:connect net:accept net:listen
cgi-get	s -- m (as of 23.04)
	<i>needs net/cgi</i> Parse input string as GET
cgi-http-header	T -- (as of 23.04)
	<i>needs net/cgi</i> Control whether or not an "HTTP" header is output. Defaults true , but use false for normal CGI scripts.
cgi-init	T -- m (as of 23.01)
	<i>needs net/cgi</i> Reads the CGI data. If T is true , allows GET data; otherwise, only POST. Currently, POST data must be JSON. Returns the URL requested, and a map of the request variables passed in.
cgi-init-stunnel	-- s m (as of 23.04)

word	sed/description
	<p><i>needs net/cgi</i></p> <p>Like net:cgi-init , but for stunnel invoked scripts, where the GET or POST is in the stdin. Returns the URL requested and the parameters as a map</p>
cgi-out	mime charset data -- (as of 23.01)
	<p><i>needs net/cgi</i></p> <p>Prints data as appropriate for a CGI response. If a buffer, converts to base64 first. 'charset' is usually "utf-8". "mime" might be "application/json" or whatever you need.</p>
close	net --
	<p>Close the net. It may not be read from or written to after this.</p> <p>See also: net:socket</p>
closed?	net -- net T (as of 19.05)
	Returns true if the net was closed by the other side; false otherwise.
connect	X net --
	<p>Connects to the net with the X returned from net:getaddrinfo .</p> <p>See also: net:bind net:accept net:listen</p>
curnet	-- net (as of 22.05)
	<p><i>needs net/utls</i></p> <p>Get the net this task is handling</p>
debug?	net -- net T
	<p><i>needs net/utls</i></p> <p>Determine if the user is wanting to debug the connection (prints RECV/SEND and data)</p>
delete	url -- opts buf true opts false (as of 20.02)
	<p><i>needs net/http</i></p> <p>Same as net:get, but with HTTP DELETE.</p>
dns	m -- m' (as of 24.02)

word	sed/description																					
	<p>Makes a DNS query according to the values passed in the map.</p> <p>Input keys are:</p> <table><tr><th>key</th><th>type</th><th>description</th></tr><tr><td>dns</td><td>array</td><td>Each entry is a dotted-ip address of a DNS server. At least one is required</td></tr><tr><td>host</td><td>string</td><td>The host to query the DNS about. Required</td></tr><tr><td>type</td><td>string</td><td>What type of record to get. One of A, MX, NS, CNAME, SOA, PTR, TXT, AAAA, SRV, OPT, SSHFP, SPF, or AXFR</td></tr><tr><td>fam</td><td>number</td><td>Either 4 or 6, for INET or INET6 (default: 4)</td></tr><tr><td>proto</td><td>string</td><td>Either tcp or udp (default: tcp)</td></tr><tr><td>class</td><td>number</td><td>Either 1=in or 255=any (default: 1)</td></tr></table> <p>The only required keys are 'dns' and 'host'. The others default as indicated (type defaults to A).</p> <p>The returned map contains keys corresponding to the DNS packet's values. Generally, "QUESTION" and "ANSWER" will appear. The ANSWER key is an array of returned "pretty-printed" values.</p>	key	type	description	dns	array	Each entry is a dotted-ip address of a DNS server. At least one is required	host	string	The host to query the DNS about. Required	type	string	What type of record to get. One of A, MX, NS, CNAME, SOA, PTR, TXT, AAAA, SRV, OPT, SSHFP, SPF, or AXFR	fam	number	Either 4 or 6, for INET or INET6 (default: 4)	proto	string	Either tcp or udp (default: tcp)	class	number	Either 1=in or 255=any (default: 1)
key	type	description																				
dns	array	Each entry is a dotted-ip address of a DNS server. At least one is required																				
host	string	The host to query the DNS about. Required																				
type	string	What type of record to get. One of A, MX, NS, CNAME, SOA, PTR, TXT, AAAA, SRV, OPT, SSHFP, SPF, or AXFR																				
fam	number	Either 4 or 6, for INET or INET6 (default: 4)																				
proto	string	Either tcp or udp (default: tcp)																				
class	number	Either 1=in or 255=any (default: 1)																				
get	url -- opts buf true false																					
	<p><i>needs net/http</i></p> <p>Retrieve the web page given by the <i>string</i> or <i>map</i> url using HTTP GET. If successful, true is on TOS, and a <i>buffer</i> b with the retrieved data is below it, and the opts map with headers etc is below that.</p> <p>If unsuccessful connecting, false is on TOS, and the net may have an "errmsg" in its opts (use "errmsg" net:@ to get it).</p> <p>Note that true will be returned even if the item requested was not found on the server. It is the programmer's responsibility to check the "retcode" and "response" keys in the RAWHEADERS key of the returned opts map to validate the expected data were returned.</p>																					
getaddrinfo	addr service -- X																					
	<p>Given a string addr, representing an internet address (dotted-ip or name), and the number or string of a service (e.g. "http" or 80), returns an opaque X which is the address information for the given server. If null is passed as addr, it means "localhost". The return value is null if the lookup failed. Use net:addrinfo>o to parse the data in the X.</p>																					
getpeername	net -- net s n (as of 17.07)																					
	<p>Return the IP address string, and port number, for the connected net.</p>																					
head	url -- opts buf true opts false																					

word	sed/description																		
	<p><i>needs net/http</i></p> <p>Same as net:get, but with HTTP HEAD.</p>																		
ifaces?	-- a (as of 17.08)																		
	<p>Returns an array containing one map for each network interface known to the system. Each map contains one or more of the following keys:</p> <table> <tr> <th>key</th><th>description</th></tr> <tr> <td>addr</td><td>IPv4 address</td></tr> <tr> <td>addr6</td><td>IPv6 address</td></tr> <tr> <td>cast</td><td>broadcast address</td></tr> <tr> <td>desc</td><td>description (Windows)</td></tr> <tr> <td>fname</td><td>friendly name (Windows)</td></tr> <tr> <td>mac</td><td>MAC address</td></tr> <tr> <td>mask</td><td>netmask</td></tr> <tr> <td>name</td><td>canonical name of the interface, e.g. "eth0"</td></tr> </table> <p>It is not an error for some of these keys to be absent (in particular: "addr" and "addr6" are mutually exclusive).</p>	key	description	addr	IPv4 address	addr6	IPv6 address	cast	broadcast address	desc	description (Windows)	fname	friendly name (Windows)	mac	MAC address	mask	netmask	name	canonical name of the interface, e.g. "eth0"
key	description																		
addr	IPv4 address																		
addr6	IPv6 address																		
cast	broadcast address																		
desc	description (Windows)																		
fname	friendly name (Windows)																		
mac	MAC address																		
mask	netmask																		
name	canonical name of the interface, e.g. "eth0"																		
ipv6?	net -- net T (as of 23.03)																		
	Returns true if the net is using IPV6.																		
launch	s -- (as of 24.05)																		
	"Opens" the URL given in a separate OS-specific process (browser).																		
listen	net --																		
	<p>Given a net which has been bound to an address using net:bind , listens for an incoming connection. Blocks until the listen succeeds or times-out.</p> <p>The default is to allow a backlog of 10 connections; you can set that by creating the socket with a map having the key backlog set to whatever value you want.</p> <p>See also: net:bind net:accept net:connect</p>																		
map>url	s -- m (as of 20.03)																		

word	sed/description
	<p><i>needs net/urlencode</i></p> <p>Converts a <i>map</i> into application/x-www-form-urlencoded format so it can be transmitted in a POST request Inverse of net:map>url: converts urlencoded form to a map <i>NOTE</i>: all values are re-encoded as <i>strings</i></p>
mime-type	s -- s s2 T (as of 23.05)
	<p><i>needs net/mime</i></p> <p>Takes a file name and returns it and its "mime-type" (based upon file-extension) on TOS. If the type isn't recognized, assumes binary data. TOS is true if the type is binary (not text).</p>
net-socket	opts -- net true false
	<p><i>needs net/utls</i></p> <p>From the options, create a suitable 'net', open to that server and 'true', or just 'false' if unable to open a socket. Takes into account proxy settings.</p>
opts	net -- net m (as of 16.12)
	Retrieve the map used to initialize the net, which is also known as the "options map" of the net.
port-is-ssl?	m -- m
	<p><i>needs net/url</i></p> <p>Given a parsed URL, determine the port to use and whether it's ssl.</p>
post	url -- opts buf true opts false
	<p><i>needs net/http</i></p> <p>Same as net:get, but with HTTP POST.</p>
proxy!	opts --
	<p><i>needs net/http</i></p> <p>Pass a string like "abc.com:900", or a map like { "proxy-server" : "abc.com", "proxy-port": 900} to set the HTTP proxy parameters. You can also set a proxy on a per-connection basis by setting the proxy-server and proxy-port keys in the options map before making the connection.</p>
put	url -- opts buf true opts false (as of 20.02)
	<p><i>needs net/http</i></p> <p>Same as net:get, but with HTTP PUT.</p>
read	net s n -- net s n net n -- net b n

word	sed/description
	<p>Same semantics as f:read but for a net (e.g. a network socket).</p> <p>See also: net:socket net:write net:close</p>
read-all	net n1 n2 -- net b (as of 19.05)
	<p>Similar to net:read , but continues reading until no more can be read or n1 bytes were received. If n1 is -1, there is essentially no limit. n2 is the number of milliseconds to wait for data; if that is exceeded, the receive stops. The returned buffer's size is the number of bytes received.</p> <p>See also: net:read</p>
read-buf	net n -- net b (as of 22.04)
	<p>Same as f:read-buf , but for a net socket.</p> <p>See also: net:read net:write</p>
recvfrom	net b n -- net X b n2 net b n -- net null (as of 17.08)
	<p>Receive a message on the net into the buffer b . Returns the number of bytes received and an X which contains the ip address and port from which it received the message, if possible. If there was an error, null is pushed instead.</p> <p><i>Note:</i> The size of the buffer is not modified to reflect the number of bytes received.</p>
s>url	s -- m
	<p><i>needs net/url</i></p> <p>Breaks a string s representing a URL into a map with the keys</p> <ul style="list-style-type: none"> • "scheme" (<i>string</i>, e.g. "http") • "url" (<i>string</i>, e.g. "google.com") • "port" (<i>number</i>, e.g. 80) • "path" (<i>string</i>, e.g. "something/index.html") • "user" (<i>string</i>, e.g. "joe") <p>Missing values will be null, except for 'port' which may be set by 'scheme'. A URL may have the form: http://joe@google.com:80/index.html and some of the components may be missing.</p>
sendto	net X sb n -- net n' (as of 17.07)

word	sed/description																																																				
	<p>Sends a message to another socket, with no guarantee of delivery. X is returned by net:getaddrinfo or net:recvfrom. sb is a string or buffer to send. n is normally 0, but may be MSG_OOB or other similar constants as defined by the OS. Returns the number of bytes sent or null if there was an error sending.</p> <p>See also: net:getaddrinfo net:recvfrom</p>																																																				
server	<p>m -- net</p> <p>Creates a net which is a TCP server processing network requests. The default port number is 8080, if one is not provided using the key "port". The required key "recv" is invoked when a client connects, and is given a net which refers back to the connected client, as well as a buffer containing the data received (on TOS). It may use net:write to return data to the client using the net it was given, and must return false to continue receiving data or true to terminate the connection.</p> <p>Valid keys are:</p> <table><thead><tr><th>key</th><th>type</th><th>description</th><th>default</th></tr></thead><tbody><tr><td>addr</td><td>s</td><td>the IP address to bind to</td><td>null</td></tr><tr><td>cert</td><td>bs</td><td>if TLS, the certificate to use</td><td></td></tr><tr><td>chunk</td><td>n</td><td>max number of bytes to receive at one time</td><td>16K</td></tr><tr><td>errcb</td><td>w</td><td>invoked if an error occurred on the connection</td><td></td></tr><tr><td>key</td><td>bs</td><td>if TLS, the key to use</td><td></td></tr><tr><td>lost</td><td>w</td><td>invoked when connection was lost</td><td></td></tr><tr><td>made</td><td>w</td><td>invoked when connection is made</td><td></td></tr><tr><td>port</td><td>n</td><td>the port to listen to</td><td>8080</td></tr><tr><td>recv</td><td>w</td><td>invoked when data arrives</td><td></td></tr><tr><td>tls</td><td>T</td><td>if true, use TLS</td><td>false</td></tr><tr><td>tlsver</td><td>n</td><td>10,11,12, or 13</td><td>12</td></tr><tr><td>to</td><td>n</td><td>seconds before timing-out</td><td>1</td></tr></tbody></table> <p>If present, the "cert" and "key" keys contain either a file-name with a PEM certificate or private key, or a buffer containing them. An exception will be thrown if "tls" is true but "cert" or "key" are not present, or if "cert" or "key" are invalid.</p> <p>The SED for made, lost, and errcb is net --. The net in the lost callback is closed, or soon to be closed, so do not write to it or try to read the remote address (there probably isn't one).</p> <p>The SED for recv is net b -- T</p>	key	type	description	default	addr	s	the IP address to bind to	null	cert	bs	if TLS, the certificate to use		chunk	n	max number of bytes to receive at one time	16K	errcb	w	invoked if an error occurred on the connection		key	bs	if TLS, the key to use		lost	w	invoked when connection was lost		made	w	invoked when connection is made		port	n	the port to listen to	8080	recv	w	invoked when data arrives		tls	T	if true, use TLS	false	tlsver	n	10,11,12, or 13	12	to	n	seconds before timing-out	1
key	type	description	default																																																		
addr	s	the IP address to bind to	null																																																		
cert	bs	if TLS, the certificate to use																																																			
chunk	n	max number of bytes to receive at one time	16K																																																		
errcb	w	invoked if an error occurred on the connection																																																			
key	bs	if TLS, the key to use																																																			
lost	w	invoked when connection was lost																																																			
made	w	invoked when connection is made																																																			
port	n	the port to listen to	8080																																																		
recv	w	invoked when data arrives																																																			
tls	T	if true, use TLS	false																																																		
tlsver	n	10,11,12, or 13	12																																																		
to	n	seconds before timing-out	1																																																		
setsockopt	net m -- net (as of 17.07)																																																				

word	sed/description															
	<p>Calls the 'setsockopt' function on the socket handled by the given net . The map is in the same format as for the "sockopts" key given to net:socket .</p> <p>See also: net:socket</p>															
socket	domain type -- net m -- net															
	<p>Create a new socket with the given parameters. domain is either net:INET4 or net:INET6 , type is either net:STREAM or net:DGRAM . null is returned on error.</p> <p>You may instead pass a map, which will create an socket using "INET4 STREAM" as well as set the 'options' for further net activity to the map. You can change the defaults using the keys:</p> <table><tr><th>key</th><th>description</th><th>default</th></tr><tr><td>domain</td><td>net:INET4 or INET6</td><td>INET4</td></tr><tr><td>proto</td><td>protocol value</td><td>0</td></tr><tr><td>sockopts</td><td>map with options as per net:setsockopt</td><td></td></tr><tr><td>type</td><td>net:STREAM or DGRAM (or any value your OS supports)</td><td>STREAM</td></tr></table> <p>See also: net:write net:read net:close</p>	key	description	default	domain	net:INET4 or INET6	INET4	proto	protocol value	0	sockopts	map with options as per net:setsockopt		type	net:STREAM or DGRAM (or any value your OS supports)	STREAM
key	description	default														
domain	net:INET4 or INET6	INET4														
proto	protocol value	0														
sockopts	map with options as per net:setsockopt															
type	net:STREAM or DGRAM (or any value your OS supports)	STREAM														
tcp-connect	x -- net (as of 23.03)															
	<p><i>needs net/connect</i></p> <p>Takes a URL (including port, possibly) or a map with connection information, and attempts to create a TCP connection. Returns the connected net, or null on failure. Prefers an IPV6 connection over IPV4, but checks for that ability. If the map contains proxy-server and proxy-port , those are used instead of host and port .</p>															
tlsserr	net -- net s (as of 23.01)															
	<p>Returns the TLS error string recorded for the last action on that net. Will return null if no error was recorded, or if the net isn't using TLS.</p>															
tlshello	net m -- net T net -- net T (as of 17.01)															
	<p>Initiates a TLS connection over the. If the map is omitted, the options map given the net on creation is used. If the TLS handshake is successful, returns true otherwise return false .</p> <p>See also: net:socket</p>															
udp-connect	x -- net (as of 23.03)															

word	sed/description
	<p><i>needs net/connect</i></p> <p>Same as net:tcp-connect but for a UDP connection. information, and attempts to create a UDP connection.</p>
url>	s -- s'
	<p>URL-decode the string as per RFC 3986.</p> <p>See also: net:>url</p>
user-agent	s --
	<p><i>needs net/http</i></p> <p>Sets the "user agent" which the HTTP library will use.</p>
valid-email?	s -- T (as of 22.04)
	<p><i>needs net/validemail</i></p> <p>Checks a string for possible validity. It first checks for a single '@' character and if there is one, it assumes the part after is a domain, and checks for an MX record. If both those tests pass, true is returned; otherwise, false .</p>
vpncheck	ip apikey -- m (as of 21.08)
	<p><i>needs net/vpncheck</i></p> <p>Given an IP address and your VPNAPI.io API key, returns a map containing information about the IP address. Keys returned are 'ip', 'security', 'location', and 'network' (as of 21.08, might change in future) Will return 'null' on failure.</p>
wait	net T n -- net T
	<p>Wait to see if the net net is ready. Will wait n milliseconds. If T is true , waits for incoming data to be ready; otherwise it waits for the underlying socket to be ready to write. Returns true if the net is ready, false otherwise. Does not actually read or write on the net.</p>
webserver	m -- net (as of 22.05)

word	sed/description
	<p><i>needs net/webserver</i></p> <p>A wrapper around net:server which processes incoming requests and handles them appropriately. The map has the same options as for net:server , but any recv member is overwritten. Additional options are for the callbacks to process specific request types:</p> <p>get , post , head , put , delete , connect , options , trace , patch , ws-start , ws , ws-end , verb? .</p> <p>The SED for every non websocket callback is net -- net T . The net has several items accessible with net:@ :</p> <ul style="list-style-type: none"> • uri -- the requested page from the URL • verb -- GET/POST/etc. • rx-query -- Received "query string" • rx-header -- received HTTP headers ('Content-type', etc) • rx-body -- the received HTTP request body (after the headers) For ws the callback SED is net b -- net T where b is the raw buffer of data sent. , and the returned T is true to end the connection and false to leave the connection open. For ws-start the callback is net a -- net T , where true means "ignore" and false means "accept connection". For ws-end the callback is just net -- net , giving the app a chance to do cleanup (but not to close the net!).
write	net s -- net n
	<p>Same semantics as f:write but for a net (e.g. a network socket).</p> <p>See also: net:socket net:read net:close</p>

Nuklear

Namespace: **nk**

Description: Low-level Nuklear Interface

word	sed/description
(begin)	m -- T (as of 20.01)

word	sed/description																																				
	<p>Starts a new window based on the map. It needs to be invoked <i>every frame</i> for every window (unless hidden), otherwise the window gets removed. Returns true if the new window can be filled with widgets, or false if not. You want to use nk:begin instead. Keys are:</p> <table><tr><th>key</th><th>type</th><th>description</th><th>default</th></tr><tr><td>bg</td><td>clr</td><td>Color (or image) used to paint this window's background</td><td>win</td></tr><tr><td>bounds</td><td>rect</td><td>Initial size and placement of window</td><td>win</td></tr><tr><td>flags</td><td>n</td><td>Some combination of the nk_panel flags</td><td>0</td></tr><tr><td>font</td><td>font</td><td>The font to use for this window's elements</td><td>win</td></tr><tr><td>name</td><td>s</td><td>A unique identifier for the window</td><td></td></tr><tr><td>padding</td><td>pt</td><td>Window padding [x,y] in pixels</td><td>[0,0]</td></tr><tr><td>style</td><td>m</td><td>A "style" to use for this window.</td><td></td></tr><tr><td>title</td><td>s</td><td>A title for the window, and its identifier if name isn't given</td><td>anon</td></tr></table> <p>See also: nk:(end) nk:get nk:set</p>	key	type	description	default	bg	clr	Color (or image) used to paint this window's background	win	bounds	rect	Initial size and placement of window	win	flags	n	Some combination of the nk_panel flags	0	font	font	The font to use for this window's elements	win	name	s	A unique identifier for the window		padding	pt	Window padding [x,y] in pixels	[0,0]	style	m	A "style" to use for this window.		title	s	A title for the window, and its identifier if name isn't given	anon
key	type	description	default																																		
bg	clr	Color (or image) used to paint this window's background	win																																		
bounds	rect	Initial size and placement of window	win																																		
flags	n	Some combination of the nk_panel flags	0																																		
font	font	The font to use for this window's elements	win																																		
name	s	A unique identifier for the window																																			
padding	pt	Window padding [x,y] in pixels	[0,0]																																		
style	m	A "style" to use for this window.																																			
title	s	A title for the window, and its identifier if name isn't given	anon																																		
(chart-begin)	n1 n2 n3 n4 -- n																																				
	Starts a chart of type n1 , with n2 entries, minimum value n3 and maximum n4 . Returns non-zero if successful, 0 if failed.																																				
(chart-begin-colored)	n1 n2 n3 n4 clr1 clr2 -- n																																				
	Same as nk:(chart-begin) but using normal color clr1 and active color clr2 .																																				
(chart-end)	--																																				
	Ends a chart after nk:(chart-begin) .																																				
(end)	-- (as of 20.01)																																				
	Terminates a window definition started using nk:begin . <i>MUST</i> be paired with nk:(begin) . See also: nk:(begin)																																				
(group-begin)	name title flags -- n (as of 20.01)																																				
	Starts a new group with internal scrollbar handling. Calls nk_group_begin_titled internally.																																				

word	sed/description
(group-end)	-- (as of 20.01)
	Ends a group. Should only be called if nk:(group-begin) returned non-zero.
(property)	s min max step inc T v -- v (as of 20.01)
	Creates a "property" control named s , with a minimum value, a maximum value, a step value, and an increment per pixel. T is true if the value in v should be an integer, false if it is a float (double).
>img	-- img (as of 20.02)
	Takes a 'snapshot' of the current screen window.
addfont	font s -- (as of 20.01)
	Adds the font to the current context with a name so it can be found by nk:push-font etc.
anti-alias	T -- (as of 20.02)
	Turns on anti-aliasing for lines and shapes. Default is true , meaning anti-aliasing is turned on.
any-clicked?	rect -- T (as of 21.01)
	Returns true if any button was clicked in the rectangle.
bounds	-- rect (as of 20.01)
	Returns the bounds of the currently active NK window
bounds!	rect -- null -- (as of 20.01)
	Sets the bounds of the currently active NK window. If given null , sets to full-screen.
button	m -- label align img symbol style w --

word	sed/description																																																																														
	<p>Takes a map describing what the button should display (or a series of parameters):</p> <table><tr><th>key</th><th>type</th><th>description</th></tr><tr><td>align</td><td>n</td><td>One of the text alignment constants, if label is defined</td></tr><tr><td>cb</td><td>w</td><td>REQUIRED: The word to invoke when the button is pressed</td></tr><tr><td>img</td><td>img</td><td>An image to display</td></tr><tr><td>label</td><td>s</td><td>A text label</td></tr><tr><td>style</td><td>m</td><td>A style definition (see below)</td></tr><tr><td>symbol</td><td>n</td><td>One of the symbol constants</td></tr></table> <p>The "style" map, if present, is defined as follows. Default values are taken from the current context style:</p> <table><tr><th>key</th><th>type</th><th>description</th></tr><tr><td>bg</td><td>clr,img</td><td>What to display as the button normal background</td></tr><tr><td>bg-active</td><td>clr,img</td><td>The bg when the button is active</td></tr><tr><td>bg-border</td><td>clr</td><td>The color of the border</td></tr><tr><td>bg-hover</td><td>clr,img</td><td>The bg when hovering over the button</td></tr><tr><td>border</td><td>n</td><td>Border width in pixels</td></tr><tr><td>draw</td><td>w</td><td>Word to invoke for custom drawing</td></tr><tr><td>draw-end</td><td>w</td><td>Word to invoke after custom drawing</td></tr><tr><td>img-align</td><td>n</td><td>NK_TEXT_ALIGN... constant to align image in button</td></tr><tr><td>img-padding</td><td>pt</td><td>Padding around image in pixels</td></tr><tr><td>img-scale</td><td>n</td><td>Percentage of button height for image (<100)</td></tr><tr><td>padding</td><td>pt</td><td>Padding around text in pixels</td></tr><tr><td>rounding</td><td>n</td><td>Radius of corners in pixels</td></tr><tr><td>touch-padding</td><td>pt</td><td>Padding in pixels for touch resolution</td></tr><tr><td>txt</td><td>clr</td><td>The fg color of text</td></tr><tr><td>txt-active</td><td>clr</td><td>The color of text when active</td></tr><tr><td>txt-align</td><td>n</td><td>One of the text alignment flags</td></tr><tr><td>txt-bg</td><td>clr</td><td>The bg color of text</td></tr><tr><td>txt-hover</td><td>clr</td><td>The color of text when hovering</td></tr></table>	key	type	description	align	n	One of the text alignment constants, if label is defined	cb	w	REQUIRED: The word to invoke when the button is pressed	img	img	An image to display	label	s	A text label	style	m	A style definition (see below)	symbol	n	One of the symbol constants	key	type	description	bg	clr,img	What to display as the button normal background	bg-active	clr,img	The bg when the button is active	bg-border	clr	The color of the border	bg-hover	clr,img	The bg when hovering over the button	border	n	Border width in pixels	draw	w	Word to invoke for custom drawing	draw-end	w	Word to invoke after custom drawing	img-align	n	NK_TEXT_ALIGN... constant to align image in button	img-padding	pt	Padding around image in pixels	img-scale	n	Percentage of button height for image (<100)	padding	pt	Padding around text in pixels	rounding	n	Radius of corners in pixels	touch-padding	pt	Padding in pixels for touch resolution	txt	clr	The fg color of text	txt-active	clr	The color of text when active	txt-align	n	One of the text alignment flags	txt-bg	clr	The bg color of text	txt-hover	clr	The color of text when hovering
key	type	description																																																																													
align	n	One of the text alignment constants, if label is defined																																																																													
cb	w	REQUIRED: The word to invoke when the button is pressed																																																																													
img	img	An image to display																																																																													
label	s	A text label																																																																													
style	m	A style definition (see below)																																																																													
symbol	n	One of the symbol constants																																																																													
key	type	description																																																																													
bg	clr,img	What to display as the button normal background																																																																													
bg-active	clr,img	The bg when the button is active																																																																													
bg-border	clr	The color of the border																																																																													
bg-hover	clr,img	The bg when hovering over the button																																																																													
border	n	Border width in pixels																																																																													
draw	w	Word to invoke for custom drawing																																																																													
draw-end	w	Word to invoke after custom drawing																																																																													
img-align	n	NK_TEXT_ALIGN... constant to align image in button																																																																													
img-padding	pt	Padding around image in pixels																																																																													
img-scale	n	Percentage of button height for image (<100)																																																																													
padding	pt	Padding around text in pixels																																																																													
rounding	n	Radius of corners in pixels																																																																													
touch-padding	pt	Padding in pixels for touch resolution																																																																													
txt	clr	The fg color of text																																																																													
txt-active	clr	The color of text when active																																																																													
txt-align	n	One of the text alignment flags																																																																													
txt-bg	clr	The bg color of text																																																																													
txt-hover	clr	The color of text when hovering																																																																													
button-color	clr w --																																																																														
	A button with the given color; invokes w when pressed.																																																																														
button-label	s w -- (as of 20.01)																																																																														

word	sed/description
	<i>needs nk/buttons</i> Creates a button with the label s . If button is pushed, invoke w .
button-set-behavior	n --
	Controls whether or not the button repeats. n is one of "BUTTON_DEFAULT" or "BUTTON_REPEATER".
button-symbol	sym w -- (as of 20.01)
	<i>needs nk/buttons</i> A button with the given symbol (nk:SYMBOL_TRIANGLE_RIGHT etc); invokes w when pressed.
button-symbol-label	lbl align sym w -- (as of 20.01)
	<i>needs nk/buttons</i> A symbol button with a label; invokes w when pressed.
chart-add-slot	type count min max -- (as of 20.01)
	Adds a number slot to the chart.
chart-add-slot-colored	type count min max clr1 clr2 -- (as of 20.01)
	Adds a number slot to the chart, with normal and active colors.
chart-push	n -- n'
	Pushes a number to a chart.
chart-push-slot	value slot -- (as of 20.01)
	Pushes value into a specific chart slot .
checkbox	s T -- T
	Creates a checkbox with label s and initial state T (true is "checked"). Returns the checked state after processing.

word	sed/description
circle	-- (as of 23.05)
	Draws a filled circle at the current position with radius set by nk:set-radius . If nk:pen-width is not 0, then an outline of that thickness in nk:pen-color is drawn as well.
clicked?	btn rect down -- T (as of 20.01)
	Returns true if the button was clicked down (true for down, false for up) in the rectangle.
close-this!	nk -- (as of 20.01)
	Forces the specified screen window to be closed.
close-this?	nk -- nk T (as of 20.01)
	Returns true if the specified screen window should be closed.
close?	-- T (as of 20.01)
	Returns true if the current screen window should be closed.
color-chooser	rect m -- T (as of 24.04)
	<i>needs gui/color</i> Displays a 'drop down' color chooser, initially filling rect . The initial color is passed in the "color" member of m . If "text" is present that will be drawn in white-on-black in the center of rect , in "font". If "cb" is provided, it is invoked if the color is changed. Usually, color is the name of a var which contains the color, and it gets updated by the actual color-picker.
color-picker	v -- (as of 20.01)
	Selects a color using an initial color. The color is placed back in the variable, as an array of [R,G,B,A] in the range [0-1].
combo	a n high pt -- n' (as of 20.10)
	Creates a combo-box whose values in the dropdown are taken from an array of strings. The initial selection is n , and the height of each item in the list is high . The overall (x,y) size is given by pt . It returns the current selection. If pt is a number, then it is the number of rows of items to show.
combo-begin-color	clr a -- (as of 20.01)

word	sed/description
	Starts a combo box with the given color and size a .
combo-begin-label	s a -- (as of 20.01)
	Start a combo box with the label and size a .
combo-cb	m cnt n high pt -- n2 (as of 20.10)
	<p>Same as nk: combo , but invokes the cb word (from the map) to get the data to show. That word's SED is m n -- s .</p> <p>The map m has at least the key "cb" which is the word to invoke to get data. cnt is the count of possible items. If pt is a number, it is the number of rows to show; otherwise, it is a (x,y) size. It may also have a member sep which is a number of pixels the "separator" width is.</p>
combo-end	-- (as of 20.01)
	Ends a combo box.
contextual-begin	flags pt rect -- n (as of 20.10)
	Starts a contextual menu with the given flags, of size pt , triggered by right-clicking in the bounds rectangle.
contextual-close	-- (as of 20.10)
	Closes the contextual menu.
contextual-end	-- (as of 20.10)
	Ends the contextual menu definition started with nk: contextual-begin .
contextual-item-image-text	img s align -- (as of 20.10)
	Adds a text item along with an image to the contextual menu, with specific text alignment.

word	sed/description
contextual-item-symbol-text	sym s align -- (as of 20.10)
	Adds a text item with a built-in symbol to the contextual menu, with specific text alignment.
contextual-item-text	s align -- (as of 20.10)
	Adds a text item to the contextual menu with specific text alignment.
cp!	xpos ypos -- (as of 21.03)
	Sets the current position (x,y) where the next widget will be placed.
cp@	-- xpos ypos (as of 21.03)
	Get the current position (x,y) where the next widget will be placed.
curpos	-- pt (as of 23.05)
	Gets the current position.
cursor-load	img xofs yofs -- X (as of 23.05)
	Creates a new cursor from the given image. The "hot spot" of the cursor is at (xofs,yofs) in the image.
cursor-set	n -- X -- (as of 23.05)
	Set the cursor. If given is a number, then 0: arrow, 1: I-beam, 2: wait, 3: crosshair, 4: waitarrow, 5: size-nw-se, 6: size-ne-sw, 7: size-we, 8: size-ns, 9: size-all, 10: no, 11: hand. Numbers outside that range are ignored. It may also be given an item created by nk:cursor-load , which is user-dependent.
cursor-show	T -- (as of 23.05)
	If true , shows the cursor; otherwise hides it.
display-info	-- a (as of 20.02)
	Returns information about each connected physical display on the system.
display@	-- n (as of 20.02)

word	sed/description
	Returns the display index of the current screen-window.
do	x -- (as of 20.01)
	<p>Passes the item to the GUI (main) task for processing. If it is a:</p> <ul style="list-style-type: none"> • word: it gets invoked • string: it gets "eval"ed • map: then the word contained in the "cb" key is invoked, with x on TOS <p>Any other type is ignored.</p>
down?	n -- T (as of 20.01)
	Returns true if the mouse button is down.
draw-image	rect img clr -- (as of 20.01)
	Draws the image in the given rectangle, filling as needed with the color.
draw-image-at	pt whence img clr -- (as of 20.03)
	<p>Draws the image without resizing it, at the point. whence determines what the reference point of the image position is, one of:</p> <ul style="list-style-type: none"> • 0 = top-left • 1 = top-right • 2 = bottom-left • 3 = bottom-right • 4 = center
draw-image-centered	rect img clr -- (as of 20.03)
	Same as nk:draw-image , but centers the image in the rectangle without resizing it.
draw-sub-image	rect1 img clr rect' -- (as of 21.01)
	Draws the portion of the image given by the rectangle rect2 in the rectangle rect1 , filling as needed with the color.
draw-text	rect s font clr1 clr2 -- (as of 20.01)

word	sed/description
	Draws the text using the font in the rectangle. clr1 is the background color, clr2 is the foreground color.
draw-text-centered	rect s font clr1 clr2 -- (as of 22.02)
	Same as nk:draw-text , but centers the text in the rectangle (or on it, if the text is wider than the rectangle).
draw-text-high	rect s font clr1 clr2 high -- (as of 20.01)
	Same as nk:draw-text , but resizes the font to high pixels.
draw-text-wrap	rect n s font clr1 clr2 high -- rect' xofs yofs [rect,n,s,font,clr1,clr2,high,show] -- rect' xofs yofs (as of 21.01)
	<p>Like nk:draw-text-high , but wraps the text in the given rectangle. high is the "row height", which if</p> <ul style="list-style-type: none"> • 0: the row height is the font height X 1.2 • negative: the row height is font height-high • positive: row height is that many pixels <p>Returns the unused portion of rect , and the x-offset (in pixels) where the text drawing ended (on the first 'row' of the returned rect').</p> <p>The input parameter n is the offset from the left of rect to start drawing. This allows you to pass the results of a previous call to start drawing.</p> <p>The yofs returned is the height of total text output.</p> <p>Also accepts an array of the parameters; if "show" is present, it determines whether or not the text is actually output.</p>
drivers	-- a (as of 21.08)
	Returns an array containing the SDL2 video drivers. The first entry is the current driver, or null if no driver is being used.
dropped	nk s T -- (as of 21.08)
	<p>DEFERRED</p> <p>Invoked when a file name or text is dropped. The string is the name of the item and true is on TOS if the item is a file.</p>
dropping	nk T -- (as of 21.08)

word	sed/description
	DEFERRED Invoked when a "drop" event begins or ends, passed true if the drop is beginning, false otherwise.
edit-focus	x --
	Sets focus (if x is a number of flags) or removes focus (if x is null) for an edit control.
edit-string	s n1 n2 n3 -- s n3 (as of 20.01)
	s is the string to edit; n1 is its max size, n2 is flags, n3 is a nk_plugin_filter enum, Returns the edit state, modifies the string
event	-- (as of 20.01)
	Checks for and processes events. "Events" can be mouse input or motion, keyboard input, OS window messages, etc. See also: nk:event-msec
event-boost	-- (as of 20.06)
	Temporarily sets the nk:event-msec value to 1 msec, for 1 second. This is necessary in some scenarios (such as orientation change if msec was -1, in which case it's done automatically) to ensure events gets properly processed.
event-msec	n -- (as of 20.01)
	Sets the number of milliseconds nk:event will wait for events to occur before timing-out. If 0, events are polled; that is, nk:event returns immediately if no events need to be processed. The default wait is 200 msec. See also: nk:event nk:event-wait
event-wait	n -- (as of 20.02)
	Sets the number of milliseconds nk:event sleeps while waiting for nk:event-msec milliseconds to pass before timing-out. If less than zero, it is set to zero. See also: nk:event nk:event-msec
event?	-- n (as of 23.05)

word	sed/description												
	<p>Returns a bitmap of the event(s) which occurred for the current render. Values are some combination of:</p> <table> <tr> <th>value</th><th>description</th></tr> <tr> <td>0x01</td><td>caused by nk:do</td></tr> <tr> <td>0x02</td><td>input: mouse or keyboard</td></tr> <tr> <td>0x04</td><td>window: resize, focus, etc.</td></tr> <tr> <td>0x08</td><td>os: app quit, etc.</td></tr> <tr> <td>0x10</td><td>drag-n-drop</td></tr> </table> <p>See also: nk:event-msec nk:event</p>	value	description	0x01	caused by nk:do	0x02	input: mouse or keyboard	0x04	window: resize, focus, etc.	0x08	os: app quit, etc.	0x10	drag-n-drop
value	description												
0x01	caused by nk:do												
0x02	input: mouse or keyboard												
0x04	window: resize, focus, etc.												
0x08	os: app quit, etc.												
0x10	drag-n-drop												
fill-arc	pt n a clr -- (as of 20.01)												
	Fills the given arc with a center at pt , a radius of n , and a min and max angle in the array a [startangle, endangle].												
fill-circle	rect clr -- (as of 20.01)												
	Fills the circle fitting in the rect with the clr.												
fill-color	n -- (as of 23.05)												
	Sets the "pen color" for drawing with line-to, move-to etc.												
fill-poly	a clr -- (as of 20.01)												
	Fills in a polygon from an array of points [x,y] which are its vertices.												
fill-rect	rect n clr -- (as of 20.01)												
	Fills the rectangle, with corners of a radius n and the given color.												
fill-rect-color	rect n l t r b -- (as of 20.01)												
	Fills the rectangle, with the colors [l,t,r,b] for 'left', 'top', 'right', and 'bottom'.												
fill-triangle	pt1 pt2 pt3 clr -- (as of 20.01)												
	Fills the triangle defined by the three pts with the clr.												

word	sed/description																					
finger	m -- (as of 23.06)																					
	<p>DEFERRED</p> <p>If the system detects a finger motion, this is invoked with m:</p> <table><tr><th>key</th><th>type</th><th>description</th></tr><tr><td>x</td><td>n</td><td>normalized x coordinate of the gesture. (0..1)</td></tr><tr><td>y</td><td>n</td><td>normalized y coordinate of the gesture. (0..1)</td></tr><tr><td>dx</td><td>n</td><td>normalized x motion (-1..1)</td></tr><tr><td>dy</td><td>n</td><td>normalized y motion (-1..1)</td></tr><tr><td>press</td><td>n</td><td>normalized finger pressure (0..1)</td></tr><tr><td>kind</td><td>n</td><td>0=up, 1=down, 2=motion</td></tr></table> <p>A finger-down event gets translated as well into a left-mouse-down, and finger-up into left-mouse-up.</p>	key	type	description	x	n	normalized x coordinate of the gesture. (0..1)	y	n	normalized y coordinate of the gesture. (0..1)	dx	n	normalized x motion (-1..1)	dy	n	normalized y motion (-1..1)	press	n	normalized finger pressure (0..1)	kind	n	0=up, 1=down, 2=motion
key	type	description																				
x	n	normalized x coordinate of the gesture. (0..1)																				
y	n	normalized y coordinate of the gesture. (0..1)																				
dx	n	normalized x motion (-1..1)																				
dy	n	normalized y motion (-1..1)																				
press	n	normalized finger pressure (0..1)																				
kind	n	0=up, 1=down, 2=motion																				
flags!	n -- (as of 21.04)																					
	<p>Sets the Nuklear window flags for the current active widget/window. Use with caution, as it might have unexpected side-effects!</p> <p>See also: nk:flags@</p>																					
flags@	-- n (as of 21.03)																					
	<p>Gets the Nuklear window flags for the current active widget/window.</p> <p>See also: nk:flags!</p>																					
flash	nk n -- (as of 21.08)																					
	<p>Flashes the specified system-window (the current one if 'nk' is null). The parameter 'n' is 0 for "stop flashing", 1 for "flash briefly" and 2 for "flash until focused".</p>																					
fullscreen	T -- (as of 20.04)																					
	<p>If true , make the current system-window "full screen"; otherwise, make it "windowed".</p>																					
gesture	m -- (as of 21.08)																					

word	sed/description																		
	<div>DEFERRED</div> <div>If the system supports gestures, then if a gesture is input by the user, a map containing information on the gesture is received:</div> <table><thead><tr><th>key</th><th>type</th><th>description</th></tr></thead><tbody><tr><td>x</td><td>n</td><td>normalized x coordinate of the gesture. (0..1)</td></tr><tr><td>y</td><td>n</td><td>normalized y coordinate of the gesture. (0..1)</td></tr><tr><td>theta</td><td>n</td><td>amount the fingers rotated during this gesture</td></tr><tr><td>dist</td><td>n</td><td>amount the fingers moved during this gesture</td></tr><tr><td>fingers</td><td>n</td><td>number of fingers in the gesture</td></tr></tbody></table>	key	type	description	x	n	normalized x coordinate of the gesture. (0..1)	y	n	normalized y coordinate of the gesture. (0..1)	theta	n	amount the fingers rotated during this gesture	dist	n	amount the fingers moved during this gesture	fingers	n	number of fingers in the gesture
key	type	description																	
x	n	normalized x coordinate of the gesture. (0..1)																	
y	n	normalized y coordinate of the gesture. (0..1)																	
theta	n	amount the fingers rotated during this gesture																	
dist	n	amount the fingers moved during this gesture																	
fingers	n	number of fingers in the gesture																	
get	s -- x a -- x (as of 20.01)																		
	Gets the item x corresponding to the key s in the currently active begin..end block. If passed an array, gets all associated values.																		
get-row-height	-- n (as of 20.02)																		
	Returns the current minimum layout row-height (including padding).																		
getfont	s -- font null -- font																		
	Returns a font of the given name in the current NK context. null returns the current font.																		
getmap	-- m																		
	Returns a map of the current window's backing store.																		
getmap!	nk -- nk m (as of 21.09)																		
	Returns the entire map of the specified window's backing store.																		
gl?	-- m (as of 20.02)																		
	Returns either null or a map containing information about the GL subsystem.																		
grid	row #rows col #cols -- rect a -- rect (as of 21.07)																		

word	sed/description
	<p>Gets the rectangle corresponding to the given (row,col), for as many #rows high and #cols wide. A negative row or col means take that row or col from the end of the grid; that is, row -1 corresponds to the last row of the grid.</p> <p>Note that the values given are not checked, so if you want valid rectangles you need to make sure the (row,col) is less than the (rows,cols) given to nk:layout-grid-begin .</p>
grid-peek	-- rect (as of 24.04)
	Returns the current grid passed to nk:grid-push .
grid-push	rect -- (as of 21.07)
	Like nk:layout-space-push but for grids. Use this to position any widget within a grid. The rectangle given does not necessarily have to have come from nk:grid , but why would you do that?
group-scroll-ofs	s -- xofs yofs (as of 21.03)
	Gets the scroll offset of the named group.
group-scroll-ofs!	s xofs yofs -- (as of 21.03)
	Sets the scroll offset of the named group.
hovered?	rect -- T (as of 20.01)
	Returns true if the mouse is over the rectangle.
hrule	clr T -- (as of 23.08)
	Creates a "horizontal rule" filling the current space, in the given color. If TOS is true and there's enough space, does rounded ends.
image	img -- (as of 20.01)
	Draws the image in the layout area.
init	-- n (as of 20.01)
	Initializes the NK (e.g. GUI) subsystem; returns 0 on failure, otherwise 1.

word	sed/description
input-button	id xofs yofs T --
	Presses the mouse button id at position (xofs,yofs). If T is <i>true</i> , button is 'down'.
input-key	n T --
	Inputs a key as if it were typed. n is one of the KEY_ * enums, and T is true if the key is down, false if up.
input-motion	xofs yofs --
	Moves the mouse to position (xofs,yofs).
input-scroll	pt --
	Scrolls to the given position.
input-string	s --
	Inputs the string as if it were typed.
key-down?	n -- T (as of 20.01)
	Returns true if the key is being pressed.
key-pressed?	n -- T (as of 20.01)
	Returns true if the key is currently pressed.
key-released?	n -- T (as of 20.01)
	Returns true if the key is currently released.
knob	min max step n dir dead T -- n' (as of 24.06)
	Draws a "knob control". If T is true , n is a float; otherwise an int.
label	s flags -- (as of 20.01)
	Draws a label using the given flags .

word	sed/description																		
label-colored	s flags clr --																		
	Same as <code>nk:label</code> but making the text colored.																		
label-wrap	s --																		
	Same as <code>nk:label</code> , but wrapping the text within the bounds of the widget.																		
label-wrap-colored	s clr --																		
	Same as <code>nk:label-wrap</code> , but with the given color.																		
layout-bounds	-- rect (as of 20.01)																		
	Returns the bounds rectangle allocated after <code>nk:layout-space-begin</code>																		
layout-grid-begin	rect rows cols -- rect m -- (as of 21.07)																		
	<p>Begins a "grid layout", with as many evenly-spaced rows and columns as indicated. If "rect" is <code>null</code> , uses the entire available space in the window; otherwise, uses the rectangle. If a map is given instead, the following keys may be used:</p> <table><tr><th>key</th><th>type</th><th>description</th></tr><tr><td>cgap</td><td>n</td><td>If present, the number of pixels to leave between each column</td></tr><tr><td>cols</td><td>n,a</td><td>If a single number, split evenly into that many columns; if an array, see below</td></tr><tr><td>rgap</td><td>n</td><td>If present, the number of pixels to leave between each row</td></tr><tr><td>rows</td><td>n,a</td><td>If a single number, split evenly into that many rows; if an array, see below</td></tr><tr><td>margin</td><td>n,a</td><td>If a single number, margin in pixels ; if an array, [left,top,right,bottom] margins</td></tr></table> <p>Arrays of numbers for 'rows' and 'cols': the length of the array determines the number of rows or columns. If the number is greater than 1, it means the item should be that many pixels wide (or high). If it is between 0 and 1, it means it should be that percentage of the space. If it is negative, it means it should take up the rest of the available space (after the other items have been allocated).</p> <p>Note that grids may be nested, so that you may use the result of <code>nk:grid</code> as input to <code>nk:layout-grid-begin</code> , to create arbitrarily complex layouts.</p> <p>The "margin" is taken around the enclosing rectangle, not around each grid item.</p>	key	type	description	cgap	n	If present, the number of pixels to leave between each column	cols	n,a	If a single number, split evenly into that many columns; if an array, see below	rgap	n	If present, the number of pixels to leave between each row	rows	n,a	If a single number, split evenly into that many rows; if an array, see below	margin	n,a	If a single number, margin in pixels ; if an array, [left,top,right,bottom] margins
key	type	description																	
cgap	n	If present, the number of pixels to leave between each column																	
cols	n,a	If a single number, split evenly into that many columns; if an array, see below																	
rgap	n	If present, the number of pixels to leave between each row																	
rows	n,a	If a single number, split evenly into that many rows; if an array, see below																	
margin	n,a	If a single number, margin in pixels ; if an array, [left,top,right,bottom] margins																	
layout-grid-end	-- (as of 21.07)																		

word	sed/description
	Terminates the current grid layout.
layout-push-dynamic	-- (as of 20.01)
	Adds a column to a templated row that dynamically grows and can go to zero if not enough space.
layout-push-static	n -- (as of 20.01)
	Adds a static column that does not grow and will always have the same size.
layout-push-variable	n -- (as of 20.01)
	Adds a column that dynamically grows but does not shrink below specified pixel width.
layout-ratio-from-pixel	n -- n' (as of 20.01)
	Utility function to calculate window percentage from a width in pixels.
layout-reset-row-height	-- (as of 20.01)
	Resets the currently used minimum row height to the font height.
layout-row	fmt h [ratio] --
	Starts layout rows where the column widths are determined by an array. There are as many columns as elements in the array, and the width is determined to be a number of pixels, if it's ≥ 1 , or a percentage of the window width otherwise.
layout-row-begin	fmt high cols --
	Begin layout of rows using nk:layout-push , until nk:layout-row-end . fmt is one of DYNAMIC or STATIC.
layout-row-dynamic	hi n -- (as of 20.01)

word	sed/description
	Sets current row layout to share horizontal space evenly between n columns of widgets. Once invoked, all subsequent widget calls beyond n will allocate a new row with same layout. The height hi may be 0 to allow auto-layout.
layout-row-end	-- (as of 20.01)
	Finishes previously started row.
layout-row-height	n -- (as of 20.01)
	Sets the currently used minimum row height.
layout-row-push	n --
	Pushes the ratio or width n
layout-row-static	hi wide n -- (as of 20.01)
	Sets the current row layout to fill n widgets in a row with the same horizontal size wide . Once invoked, all subsequent widget calls beyond n will allocate a new row with same layout. The height hi may be 0 to allow auto-layout.
layout-row-template-begin	n -- (as of 20.01)
	Begins the row template declaration with a row-height of n .
layout-row-template-end	-- (as of 20.01)
	Marks the end of the row template.
layout-space-begin	fmt hi n -- (as of 20.01)

word	sed/description
	Begins a new layout space that allows specifying each widget's position and size. fmt is DYNAMIC for window ratio, or STATIC for fixed size columns. hi is height of each widget in row, or 0 for auto layout. n is the count of widgets in the row, e.g. number of columns.
layout-space-end	-- (as of 20.01)
	Marks the end of the layout space.
layout-space-push	a -- (as of 20.01)
	Pushes the position and size of the next widget in own coordinate space either as pixel or ratio. a is position and size in layout space local coordinates
layout-widget-bounds	-- rect (as of 20.01)
	Calculates the bounds a static layout row can fit inside the current window.
line-rel	pt -- (as of 23.05)
	<p>Draws a line to the pt relative to the current position.</p> <p>See also: nk:line-to</p>
line-to	pt -- (as of 23.05)
	Draws a line from the "current position" set with nk:move-to , to pt , using the pen width set with nk:pen-width and the stroke color set with nk:pen-color . Afterwards, the current position is pt .
list-begin	id flgs high total b -- (as of 20.03)
	<p>Starts a list.</p> <p>See also: nk:list-end nk:list-new</p>
list-end	b -- (as of 20.03)
	<p>Ends a list.</p> <p>See also: nk:list-begin nk:list-new</p>

word	sed/description
list-new	-- b (as of 20.03)
	<p>Allocates a buffer big enough to serve as a 'nk_list_view'</p> <p>See also: nk:list-begin nk:list-end</p>
list-ofs	s n -- (as of 24.04)
	<p><i>needs nk/loaded</i></p> <p>Sets the scrollbar offset n in the list titled s .</p>
list-range	b -- begin end count (as of 20.03)
	<p>Returns the numbers for the given list's current begin, end, and count.</p> <p>See also: nk:list-begin nk:list-end</p>
m!	s x -- (as of 20.01)
	Puts the item x in the nk's backing store under the key s .
m@	s -- x a -- x (as of 20.01)
	Gets the item corresponding to the key in the nk's backing store. If passed an array, gets all associated values.
make-style	m -- X (as of 20.01)
	<p>Creates a new item containing style information from the map. Use it with nk:use-style . See the manual for details on the map keys and values.</p> <p>See also: nk:use-style</p>
max-vertex-element	n1 n2 -- (as of 20.01)
	Sets the maximum number of "elements" n1 and "vertices" n2 per render frame. The defaults are 512K and 128K respectively. Limited by the amount of memory in your video card.
maximize	T -- (as of 23.09)
	Maximizes the screen window if true , or restores it.

word	sed/description
measure	s -- pt (as of 20.01)
	Returns a pt containing the height and width the string would take in the current font.
measure-font	s font -- pt s s' -- pt (as of 20.01)
	Returns a pt containing the height and width the string would take with the font (either a string naming a defined font, or a font*).
menu-begin	s align pt -- T s align pt img -- T s align pt sym -- T
	Begins a menu with a given size pt , text-alignment, and label. May also have an image or symbol. In the latter two cases, if null is used instead of a string, no text will be shown.
menu-close	--
	Closes an open menu.
menu-end	--
	Ends a menu definition.
menu-item-image	s flg img w --
	Begins a menu item with an image and label, with a given alignment. Invokes w if the menu was selected.
menu-item-label	s flg w --
	Begins a menu item with a label, with a given alignment. Invokes w if the menu was selected
menu-item-symbol	s flg sym w --
	Begins a menu item with a symbol and label, with a given alignment. Invokes w if the menu was selected.
menubar-begin	--
	Starts menu layout.

word	sed/description
menubar-end	--
	Ends menu layout.
minimize	T -- (as of 23.09)
	Minimizes the screen window if true , or restores it.
mouse-pos	-- xpos ypos (as of 20.03)
	Returns the (x,y) position of the mouse in the currently focused screen window.
move-back	pt -- (as of 23.05)
	Moves the "current position" to the previously set one.
move-rel	pt -- (as of 23.05)
	Moves the "current position" to a new position <i>relative</i> to the current one. See also: nk:line-to
move-to	pt -- (as of 23.05)
	Moves the "current position" for drawing to the point [x,y] . See also: nk:line-to
msg	title msg -- (as of 23.07)
	Prints a simple message box (usually for errors).
msgdlg	opts -- (as of 20.01)

word	sed/description
	<p><i>needs nk/msgdlg</i></p> <p>Display a message dialog with parameters from the <i>map</i> opts. Keys:</p> <ul style="list-style-type: none"> • "title" - the title of the dialog • "msg" - the contents of the dialog • "buttons" - an array of maps containing: <ul style="list-style-type: none"> • "label" - the text of the button • "cb" - the word to call back when the button is pressed • "bounds" - a rectangle in which to locate the message; if 'null', the current window is used • "font" - the font to use for the message (default to window font) • "clr" - the color to use for the message (defaults to black) • "bfont" - the font to use for the buttons (default to window font)
ontop	nk T -- (as of 22.08)
	Makes that system window 'topmost' and keeps it above the rest (if true ; otherwise not)
option	s T -- T
	Creates an option (radio-button) with the label s and initial state T (true is "checked"). Returns the checked state after processing.
pen-color	n -- (as of 23.05)
	Sets the "pen color" for stroke-drawing with line-to, move-to etc.
pen-width	n -- (as of 23.05)
	Sets the "pen width" for drawing with line-to, move-to etc.
plot	type a -- (as of 20.01)
	Plots an array of numbers with the type of chart (CHART_LINES, CHART_COLUMN).
plot-fn	x type w count offset -- (as of 20.01)
	Plots an array of numbers with the type of chart (CHART_LINES, CHART_COLUMN), invoking w with a SED of x n -- n' for each of count items starting at the given offset.
polygon	a -- (as of 24.02)

word	sed/description
	<p>Draws a filled polygon using the current position as one vertex, and the array of other vertices as relative positions. Uses the current fill color and if a pen width is given, an outline stroke. Current position remains the starting vertex.</p>
pop-font	-- (as of 20.01)
	<p>Pops the font which was set with nk:push-font . The previous font becomes current.</p> <p>See also: nk:push-font</p>
popup-begin	kind title flag size -- n
	Starts a popup definition.
popup-close	-- (as of 20.10)
	Closes the popup.
popup-end	--
	Ends a popup definition.
popup-scroll-ofs	-- xofs yofs (as of 20.10)
	Returnss the current scroll offsets of the popup.
popup-scroll-ofs!	xofs yofs -- (as of 20.10)
	Sets the scroll offsets of the current popup.
progress	max T v --
	<p>Creates a progress bar from [0..max]. If T is true, the user can modify the bar. v is a var, a string, or a number as with nk:slider</p> <p>See also: nk:slider</p>
prop-int	min max step v -- (as of 20.01)

word	sed/description
	<p><i>needs nk/property</i></p> <p>Creates a slider which modifies the</p>
pt-in?	rect pt -- T (as of 22.02)
	Returns true if the point is within the rectangle (inclusive of rectangle boundaries).
pt>local	pt -- pt' (as of 20.01)
	Converts the point from screen coordinates to local ones.
pt>screen	pt -- pt' (as of 20.01)
	Converts the point from local coordinates to screen ones.
push-font	s -- (as of 20.01)
	<p>Uses the named font (as defined in the fonts atlas). Paired with nk:pop-font .</p> <p>See also: nk:pop-font</p>
raise	nk -- (as of 22.08)
	Raises that system window above others, and gives it focus.
rect-rel	pt -- (as of 23.05)
	Strokes and fills a rectangle relative to the current position.
rect-to	pt -- (as of 23.05)
	Strokes and fills a rectangle from the current position to pt .
rect>local	rect -- rect' (as of 20.01)
	Converts the rectangle from screen coordinates to local ones.
rect>screen	rect -- rect' (as of 20.01)
	Converts the rectangle from local coordinates to screen ones.

word	sed/description
released?	n -- T (as of 20.01)
	Returns true if the mouse button was pressed and is now released.
render	-- (as of 20.01)
	<p>Invokes the Nuklear GL rendering routine, which iterates the list of items "drawn" and causes them to be output to the screen. In most systems, the output will be to the hardware-accelerated renderer (via GL or Metal etc.). On some systems, a much slower software renderer will be used because the hardware doesn't support accelerated rendering. It is invoked by the default nk:render-loop .</p> <p>See also: nk:render-loop nk:frame</p>
render-timed	w n -- (as of 23.05)
	<p><i>needs nk/render-timed</i></p> <p>A render-loop which creates a separate task for timing the loop. Within the render loop, invoke nk:timer? to see if the render was caused by a timer-timout (or if false , some other event). n is the number of milliseconds the timer delays.</p>
rendering	--
	<p>DEFERRED</p> <p><i>needs nk/loaded</i></p> <p>Invoked when the rendering loop is first started.</p>
restore	-- (as of 21.06)
	Restores the previously saved GL transformation matrix.
rotate	angle x y z -- (as of 21.06)
	Rotates the GL canvas through the given angle around the vector (x,y,z)
rotate-rel	angle -- (as of 23.05)
	Rotates the canvas by angle degrees clockwise around the current position.
rtl!	T -- (as of 24.03)
	Sets the "RTL" (right-to-left) value for the current NK window. This inverts the meaning of "left" and "right" so that you needn't have code to switch between LEFT and RIGHT alignment if you print RTL language text (Hebrew/Arabic/Farsi etc). The default is to output LTR (e.g. false for RTL).

word	sed/description
rtl?	-- T (as of 24.04)
	Gets the "RTL" (right-to-left) value for the current NK window. Useful in widgets/libraries to determine proper RTL orientation.
save	-- (as of 21.06)
	Saves the GL transformation matrix.
scale	x y z -- (as of 21.06)
	Scales the GL canvas by the given amounts along the (x,y,z) axes.
scancode?	n -- T a -- a' (as of 20.03)
	Returns true if the key indicated by the SDL scan-code number n is pressed. needs nk/keyboard to load enums for the scan-codes. If an array of scan-codes is given, an corresponding array is returned.
screen-saver	T -- (as of 21.02)
	Enables the screen-saver (if true) or disables it.
screen-size	-- a (as of 20.01)
	Returns the screen's size [w,h] in pixels.
screen-win-close	nk -- (as of 20.01)
	Lets the given screen window know that it should close.
selectable	str align img val -- val str align n val -- val str align null val -- val (as of 20.01)
	Creates a "selectable" with a label given by str , aligned according to align , with a value (true/false/1/0) of val . Returns the possibly modified val . The item under val may be an image, a number (of a nk symbol), or null .
set	s x -- (as of 20.01)
	Puts the item x in the currently active window's backing store under the string key s .
set-font	s -- (as of 21.01)

word	sed/description
	<p>Uses the font with the given name. Unlike nk:push-font , simply sets the font rather than pushing it on the font stack.</p> <p>See also: nk:push-font</p>
set-num-vertices	n1 n2 n3 -- (as of 20.01)
	<p>Sets the number of vertices used to draw a circle n3 , a curve n2 and an arc n1 . The defaults are 21; setting to higher values makes smoother curves, with a performance and memory cost. Avoid setting to a number which makes each segment drawn less than a pixel -- that's just a waste of resources.</p>
set-radius	n -- (as of 23.05)
	<p>Sets the radius of a subsequent nk:circle .</p>
setpos	pt --
	<p>Sets the position of NK window.</p>
setwin	s -- T (as of 20.01)
	<p>Sets the current screen window to the named one. Returns true if successful; false if the name doesn't correspond to an active or known window.</p> <p>See also: nk:win nk:win?</p>
show	nk T -- (as of 21.09)
	<p>Shows (if true) or hides the screen-window. If null means "current screen-window".</p>
slider	min max step val T --
	<p>Creates a slider control. If T is true , it's for integer values; otherwise, float values. The item val is either a var, in which case its value is modified by the slider, or a string in which case the value accessed by get/put is modified.</p> <p>See also: nk:progress</p>
slider-int	min max step v -- (as of 20.01)

word	sed/description
	<i>needs nk/sliders</i> Creates a slider which modifies the
space	-- (as of 21.02)
	Takes the place of the item which was supposed to have been inserted in a row, allowing the item to appear or not, without changing the layout. That is, allows you to omit an item from a layout.
spacing	n -- (as of 20.01)
	Sets the number of columns in this layout.
stroke-arc	pp1 n1 a n2 clr -- (as of 20.01)
	Strokes an arc of thickness n2 centered on the point pt1 , with radius n1 , color clr , with starting and stopping angles from a .
stroke-circle	rect n clr -- (as of 20.01)
	Strokes a circle fitting the rectangle, with line thickness n , in the given color.
stroke-curve	pt1 pt2 pt3 pt4 n clr -- (as of 20.01)
	Strokes a cubic-Bézier curve from point p1 to end point p4 , with control points p2 and p3 , and thickness n in the given color.
stroke-line	pt1 pt2 n clr -- (as of 20.01)
	Strokes a line of thickness n using the color, from point pt1 to point p2 .
stroke-polygon	a n clr -- (as of 20.01)
	Stroke an array of points as a polygon, with line thickness n , in the given color.
stroke-polyline	a n clr -- (as of 20.01)
	Strokes an array of points, with line thickness n , in the given color.
stroke-rect	rect n1 n2 clr -- (as of 20.01)

word	sed/description															
	Strokes a rectangle with corners of radius n1 and line thickness n2 , in the given color.															
stroke-tri	pt1 pt2 pt3 n clr -- (as of 20.01)															
	Strokes a triangle between the given points, with line thickness n , in the given color.															
style-from-table	a -- null -- (as of 20.01)															
	If given an array of numbers with nk:COLOR_COUNT values, uses them to set the colors in this "style". If given null , resets to the default.															
swipe	m -- (as of 23.06)															
	DEFERRED If a finger-swipe motion occurs, this is invoked with m: <table><tr><th>key</th><th>type</th><th>description</th></tr><tr><td>x</td><td>n</td><td>normalized x coordinate of the gesture. (0..1)</td></tr><tr><td>y</td><td>n</td><td>normalized y coordinate of the gesture. (0..1)</td></tr><tr><td>t</td><td>n</td><td>angle (radians) of swipe</td></tr><tr><td>d</td><td>n</td><td>direction of swipe: 0=indeterminate ; 1= left, 2=right, 3=up, 4=down</td></tr></table>	key	type	description	x	n	normalized x coordinate of the gesture. (0..1)	y	n	normalized y coordinate of the gesture. (0..1)	t	n	angle (radians) of swipe	d	n	direction of swipe: 0=indeterminate ; 1= left, 2=right, 3=up, 4=down
key	type	description														
x	n	normalized x coordinate of the gesture. (0..1)														
y	n	normalized y coordinate of the gesture. (0..1)														
t	n	angle (radians) of swipe														
d	n	direction of swipe: 0=indeterminate ; 1= left, 2=right, 3=up, 4=down														
swipe-dir-threshold	n -- (as of 23.06)															
	Sets the threshold below which the difference between x and y values makes it impossible to choose a "direction" of swipe. The default is 0.1.															
swipe-threshold	n -- (as of 23.06)															
	Sets the threshold below which a finger motion will not be considered a swipe event. The default is 0.02.															
text	s -- (as of 23.05)															
	Outputs the text using the text-font , in color pen-color , aligned text-align , relative to the current position.															
text-align	n -- (as of 23.05)															

word	sed/description
	Sets the text alignment for nk:text .
text-font	s -- (as of 23.05)
	Sets the font for drawing.
text-pad	rect -- (as of 23.05)
	Sets the padding [left,top,right,bottom] for the text output. That is, the output rectangle is increased by the given size.
text?	-- s (as of 20.01)
	Returns any text which has been entered (key-presses). If there's no input, an empty string is returned.
timer-delay	n -- (as of 23.05)
	<i>needs nk/render-timed</i> Set / change the number of milliseconds the timer-task delays, and notify the timer task of the change.
timer?	-- T (as of 23.05)
	<i>needs nk/render-timed</i> Returns true if the current event causing the render is a user-event
toast	s n -- (as of 24.05)
	<i>needs nk/toast</i> Displays the given text in a notice at the bottom of the screen for n seconds.
tooltip	s -- (as of 20.10)
	Presents a tooltip.
translate	x y z -- (as of 21.06)
	Translates the GL canvas by the given amounts along the (x,y,z) axes.
tree-pop	--

word	sed/description
	Pops the most recent tree state; only invoke if nk:tree-state or nk:tree-state-push returns true .
tree-state-push	kind img s state -- newstate kind s state -- newstate
	<p>Pushes a new tree node. state is either MAXIMIZED or MINIMIZED. It returns the new state after the call. Prefer: nk:tree-push as it manages state on its own.</p> <p>See also: nk:tree-pop nk:tree-push</p>
triangle	pt pt -- (as of 24.02)
	Draws a filled triangle using the current position as one vertex, and the other two (relative) points as the other vertices. Uses the current fill color and if a pen width is given, an outline stroke. Current position remains the starting vertex.
use-style	X -- m -- (as of 20.01)
	<p>Uses the style created by nk:make-style , or an equivalent map. If X is null , pops the style (reverts to previous style).</p> <p>See also: nk:make-style</p>
vsync	T --
	If false , doesn't wait for 'vertical sync' to swap GL buffers. Defaults to true , meaning buffers swap at your monitor's vertical sync rate.
widget	-- rect n (as of 20.01)
	Creates space for a new widget in the current layout. Returns the allocated bounds rect , and the status n which is one of WIDGET_INVALID, WIDGET_VALID, or WIDGET_ROM.
widget-bounds	-- a (as of 20.01)
	Returns the bounds [x,y,w,h] of the current widget.
widget-disable	T -- (as of 23.09)

word	sed/description
	If true , begins a "disabled widgets" section until false nk:widget-disable . Widgets in the disabled area will be shown differently and are incapable of interaction.
widget-fitting	pt -- rect n (as of 20.01)
	Same as nk:widget but with [x,y] padding given by pt
widget-high	-- n (as of 20.01)
	<i>needs nk/widgets</i> Returns the height of the current widget.
widget-hovered?	-- T (as of 20.01)
	Returns true if the mouse is over the current widget.
widget-mouse-click-down?	n T -- T (as of 20.01)
	Returns true if the specified mouse button is down (true) or up (false) in the current widget.
widget-mouse-clicked?	n -- T (as of 20.01)
	Returns true if the specified mouse button was clicked in the current widget.
widget-pos	-- pt (as of 20.01)
	<i>needs nk/widgets</i> Returns the local position [x,y] of the current widget.
widget-size	-- pt (as of 23.09)
	Returns the size allocated for the current widget.
widget-size-allot	-- pt (as of 20.01)

word	sed/description
	<i>needs nk/widgets</i> Returns the size [w,h] of the current widget.
widget-wide	-- n (as of 20.01)
	<i>needs nk/widgets</i> Returns the width of the current widget.
win	m -- (as of 20.01)

Creates a screen window from the map. A "screen window" is an OS-specific window, an outermost container for **nk** windows and widgets. The created window will become the "current" window. To use other windows you must make them current using **nk:setwin** . Throws on failure to create the window. Keys are:

key	type	description	default
alpha	n	Opacity of the window, in range [0,1.0]	1.0
bg	clr	Background color to paint the window between frames	0xFF808080
decorated	T	Does the window have a title bar etc.	true
display	n	The physical display to use	0
font	s	Default font for items drawn	system
fontheight	n	Default height of font	13
fonts	m	Map of (id,filename) of all fonts this window or its children will use	[system]
fullscreen	T	If true, make window fill the screen	false (true on mobile)
high	n	Height of the window in pixels	screen
maxh	n	Maximum height, in pixels	screen
maximize	T	Create the window maximized	false
maxw	n	Maximum width, in pixels	screen
minh	n	Minimum height, in pixels	0
minimize	T	Create the window minimized	false
minw	n	Minimum width, in pixels	0
name	s	REQUIRED: The name by which this window is known to 8th	
onclose	w	When window clicked 'close', SED nk -- T (return true to close)	
onenter	w	When window entered (true) or left, SED nk T --	
onfocus	w	When window got (true) or lost focus, SED nk T --	
onminmax	w	When window maximized (1), restored (0), or minimized (-1) SED nk n --	
onmove	w	When window moved, SED nk x y --	
onshow	w	When window shown (true) or hidden, SED nk T --	
onsize	w	When window resized, SED nk w h --	
resizable	T	Does the window have a resize widget	true
title	s	The titlebar title	appname
topmost	T	Is the window 'always on top'	false
unicode-ranges	a	Ranges of Unicode glyphs to include [low,high],...	[0x0020, 0x00FF]
visible	T	Is the window visible initially	true
wide	n	Width of the window in pixels	screen
x	n	Position of left in pixels	centered
y	n	Position of top in pixels	centered

word	sed/description
	<p>The x, y, wide, and high values are in pixels, unless they are in the range (0,1]. In that case they represent a fraction of the screen size.</p> <p>The 'fonts' map keys are meaningful identifiers, like "buttonfont", while the values are font descriptor strings (the full path to the TTF file, with an optional colon and pixel size), or a buffer containing a font (e.g. from an asset). An exception will be thrown if no valid fonts were given.</p> <p>See also: nk:win? nk:setwin nk:close?</p>
win-bounds	-- rect (as of 20.01)
	Returns the bounds [x,y,w,h] of the currently processed window. Use only within begin... end.
win-bounds!	n a -- (as of 20.01)
	Sets the position and size of the named window. Changing the bounds of the window currently being processed is not permitted.
win-close	s -- (as of 20.01)
	Closes the named window. Does not work on the currently active window.
win-closed?	s -- T (as of 20.01)
	Return true if the named window was closed.
win-collapse	s n -- (as of 20.01)
	Collapses the named window. Does not work on the currently active window. n is one of MINIMIZED or MAXIMIZED.
win-collapsed?	s -- T (as of 20.01)
	Return true if the named window is minimized/collapsed.
win-content-bounds	-- rect (as of 20.01)
	Returns the bounds [x,y,w,h] of the currently processed window's currently visible and non-clipped
win-focus	s -- (as of 20.01)

word	sed/description
	Makes the named window the active one.
win-focused?	-- T (as of 20.01)
	Return true if the window currently being processed has the focus.
win-hidden?	s -- T (as of 20.01)
	Return true if the named window was hidden.
win-high	-- n (as of 20.01)
	<i>needs nk/win</i> Returns the height of the currently processed window.
win-hovered?	-- T (as of 20.01)
	Return true if the window currently being processed is being hovered over by the mouse.
win-icon!	img -- (as of 24.05)
	Sets the icon of the current screen-window.
win-pos	-- pt (as of 20.01)
	<i>needs nk/win</i> Returns the screen position [x,y] of the currently processed window.
win-scroll-ofs	-- xofs yofs (as of 20.01)
	Returns the x,y offsets which are the currently processed window's scrollbar offsets
win-scroll-ofs!	xofs yofs -- (as of 20.01)
	Sets the currently processed window's scrollbar offsets.
win-show	s n -- (as of 20.01)
	Shows the named window. n is one of HIDDEN or SHOWN
win-size	-- pt (as of 20.01)

word	sed/description
	<i>needs nk/win</i> Returns the width and height [w,h] of the currently processed window.
win-title!	s -- (as of 24.05)
	Sets the title-bar title of the current screen-window.
win-wide	-- n (as of 20.01)
	<i>needs nk/win</i> Returns the width of the currently processed window.
win?	-- nk (as of 20.01)
	Returns the current screen window (or null if there is none). See also: nk:win nk:setwin nk:close?

Number

Namespace: **n**

Description: Numbers: integers or floating-point

word	sed/description
!	n -- n!
	<i>needs math/factorial</i> Returns the factorial of the <i>number</i> n , non-recursively.
*	n n2 -- n'
	Multiplies the numbers. See also: n:/ n:+ n:- n:*/
*/	n x y -- n'

word	sed/description
	<p>Calculates $n * (x/y)$. Can be faster or more accurate than $n * y / x$.</p> <p>See also: <code>n:* n:/</code></p>
+	<code>n n2 -- n'</code>
	<p>Adds two numbers.</p> <p>See also: <code>n:- n:1+ n:* n:/</code></p>
+!	<code>n v --</code>
	<p>Adds the number <code>n</code> to the number held in the variable <code>v</code> . The variable should be invoked by name, and not accessed using <code>@</code> .</p>
-	<code>n n2 -- n'</code>
	<p>Subtracts <code>n2</code> from <code>n</code> .</p> <p>See also: <code>n:+ n:* n:/ n:1-</code></p>
/	<code>n n2 -- n'</code>
	<p>Divides <code>n</code> by <code>n2</code> .</p> <p>See also: <code>n:* n:+ n:- n:/ n:/mod n:mod</code></p>
/mod	<code>n n2 -- rem quo</code>
	<p>Divides <code>n</code> by <code>n2</code> , returning the integer remainder <code>rem</code> and quotient <code>quo</code> . The returned values satisfy the relation <code>n = (quo*n2) + rem</code> for all numeric types, and <code>rem</code> is always positive. This means that the value <code>quo</code> may not be the same as you would get with <code>n n2 /</code> .</p> <p>The reason for this is that we want <code>mod</code> to always return a positive value, so that e.g. array offset calculations are correct.</p> <p>If one of the arguments is not an 'int', then while the relation holds, the remainder could be negative.</p> <p>See also: <code>n:mod n:/</code></p>
1+	<code>n -- n'</code>

word	sed/description
	<p>Increments the number by 1.</p> <p>See also: n:1- n:+</p>
1-	n -- n'
	<p>Decrements the number by 1.</p> <p>See also: n:1+ n:-</p>
<	n n2 -- T
	<p>Compares two numbers, returning true if n < n2 , otherwise false . To compare \leq (less or equal) use n:> not .</p> <p>See also: n:= n:> n:cmp n:sgn n::~=</p>
=	n n2 -- T
	<p>Compares two numbers, returning true if they are equal, otherwise false .</p> <p>See also: n:cmp n: n: n:sgn n::~=</p>
>	n n2 -- T
	<p>Compares two numbers, returning true if n > n2 , otherwise false . To compare \geq (greater or equal) use n:< not .</p> <p>See also: n:= n:< n:cmp n:sgn n::~=</p>
>bool	n -- T (as of 22.06)
	<p>Converts a number (or anything, actually) to explicitly true or false .</p>
BIGE	-- e
	<p><i>needs math/big</i></p> <p>Returns the <i>number</i> "e" (2.7182...) as a big-float with 130 digits precision.</p>
BIGPI	-- pi

word	sed/description
	<i>needs math/big</i> Returns the <i>number</i> "pi" (3.1415...) as a big-float with 130 digits precision.
E	-- n
	The mathematical constant "e" (2.71828...) as a "float". If you want an extended "bfloat", accurate to 130 decimal places, use needs math/big . See also: n:PI n:BIGE
PI	-- n
	The mathematical constant "pi" ($\pi = 3.14159...$) as a "float". If you want an extended "bfloat", accurate to 130 decimal places, use needs math/big . See also: n:E n:BIGPI
^	n n2 -- n'
	Raises n to the power n2 . See also: n:* n:exp
_mod	x y -- n (as of 21.06)
	<i>needs math/fmod</i> "floored modulo", like the C "fmod()" function, but result has same sign as divisor.
abs	n -- n'
	Returns the absolute-value of n .
acos	n -- n'
	Calculates the arc-cosine of the number n , in radians. See also: n:cos
acosd	cos(n) -- n
	<i>needs math/trigd</i> Returns the angle n in degrees, whose cosine is given.

word	sed/description
acosh	n -- n' (as of 24.03)
	Returns the hyperbolic arc-cosine of the input number.
andor	n1 n2 n3 -- n4
	n4 is the result of n1 n2 band n3 bor
asin	n -- n'
	Calculates the arc-sine of the number n , in radians. See also: n:sin
asind	sin(n) -- n
	<i>needs math/trigd</i> Returns the angle n in degrees, whose sin is given.
asinh	n -- n' (as of 24.03)
	Returns the hyperbolic arc-sine of the input number.
atan	n -- n'
	Calculates the <i>principle</i> value of the arc-tangent of n , in radians in the range $[-\pi/2, +\pi/2]$. See also: n:atan2 n:tan
atan2	x y -- n
	Calculates the <i>principle</i> value of the arc-tangent of y/x, in radians. See also: n:atan n:tan
atand	tan(n) -- n
	<i>needs math/trigd</i> Returns the principle angle n in degrees, whose tangent is given.
atanh	n -- n' (as of 24.03)

word	sed/description
	Returns the hyperbolic arc-tangent of the input number.
band	n n2 -- n'
	Returns the <i>bitwise</i> AND of two numbers. See also: n:bor n:bxor
between	n low hi -- T
	Returns true if the number is between low and hi , inclusive; e.g. if low ≤ n ≤ hi .
bfloat	n -- n'
	Converts the number to a big floating-point representation, if possible. See also: n:int n:bint n:float
bic	n ix -- n' (as of 18.06)
	Clears the bits in the number which are set in the bitmap number ix .
bint	n -- n'
	Converts the number to a big integer representation, if possible. Floating-point n are truncated to integers first. See also: n:int n:float n:bfloat
binv	n -- n' (as of 18.06)
	Inverts all bits of the number.
bnot	n -- n'
	Returns the bitwise NOT of the number n .
bor	n n2 -- n'
	Returns the <i>bitwise</i> OR of two numbers. See also: n:band n:bxor

word	sed/description
bxor	n n2 -- n'
	<p>Returns the <i>bitwise</i> XOR of two numbers.</p> <p>See also: n:band n:bor</p>
cast	n bits T -- n2 (as of 20.02)
	<p>Ensures that the number is (in binary form) an integer of bits width. So for example, 8 true n:cast would ensure the was "cast" to a "int8_t". That is, 0xFF 8 true n:cast . prints "-127" instead of "255". n is converted first to an "int", and bits must be 64 or less. The lowest bits bits of the integer are then returned, signed if T is true .</p>
ceil	n -- n'
	<p>Returns the nearest integer, $\geq n$.</p> <p>See also: n:floor n:round n:ceil</p>
clamp	n low hi -- n'
	<p>Ensure that the number n is not less than low and not higher than hi (e.g. $low \leq n \leq hi$). So for example 23 10 20 n:clamp returns "20".</p> <p>See also: n:min n:max</p>
cmp	n n2 -- n'
	<p>Compares the two numbers, returning -1 if n < n2 , 0 if equal, and 1 if n > n2 .</p> <p>See also: n:= n:sgn</p>
comb	p k -- comb(p,k)
	<p><i>needs math/comb-perm</i></p> <p>Returns the unique combination of p items taken k at a time.</p>
cos	n -- n'
	<p>Calculates the cosine of n (radians).</p> <p>See also: n:sin n:tan n:acos</p>
cosd	n -- cos(n)

word	sed/description
	<i>needs math/trigd</i> Returns the cosine of the angle n , specified in degrees.
cosh	n -- n' (as of 24.03)
	Returns the hyperbolic cosine of the input number.
emod	x y -- n (as of 21.06)
	<i>needs math/fmod</i> "Euclidian modulo", result is always positive
erf	n -- n' (as of 24.03)
	Returns the "error function" of the input.
erfc	n -- n' (as of 24.03)
	Returns the "complementary error function" of the input, equivalent to 1 swap n:erf n:- .
exp	n -- n'
	Exponential of n : raises "e" to the n power; inverse of n:ln . See also: n:ln
expm1	n -- n' (as of 19.09)
	Returns expm1(n), e.g.: e^n-1
expmod	n n2 n3 -- n' (as of 17.05)
	Calculates the value of n raised to the n2 power, modulo n3 . Works for integer and big integer values only, otherwise returns null . Also returns null on overflow or other errors. Note: n2 must be positive and less than n3 .
float	n -- n'
	Converts the number to a (native IEEE double) floating-point representation, if possible. See also: n:int n:bint n:bfloat

word	sed/description
floor	n -- n'
	Returns the <i>nearest</i> integer, $\leq n$. See also: n:trunc n:round n:ceil
fmod	n n2 -- n' (as of 21.06)
	Remainder of n divided by n2 . Unlike n:mod , it keeps the floating-point remainder. That is, 5.25 5 fmod gives 0.25 rather than 0 . The sign of the result is the same as that of the dividend, e.g. -3 2 fmod gives -1 .
frac	n -- n'
	Returns the fractional part of the number n . That is, 1.234 n:frac returns 0.234 .
gcd	n n2 -- n' (as of 17.05)
	Calculates the greatest-common-divisor of the two numbers. Returns null if NaN or Inf etc are given.
int	n -- n'
	Converts the number to a (native 64-bit) integer representation, if possible. This is equivalent to casting to an "int" in C-like languages. If the conversion is not possible (because it is outside the range of an integer), the original number is returned. See also: n:bint n:float n:bfloat
invmod	n n2 -- n' (as of 17.05)
	Calculates the value of the inverse of n , modulo n2 . Works for integer and big integer values only, otherwise returns null . Also returns null on overflow or other errors.
kind?	n -- n n'

word	sed/description
	<p>Returns a number corresponding to the given number's "kind":</p> <ul style="list-style-type: none"> • 0: native integer • 1: native floating point • 2: big integer • 3: big floating point • 4: NaN • 5: Inf • 6: NegInf <p>If m is negative, it indicates that n is a complex number of the given type (e.g. -2 means "a complex number whose components are big integers").</p>
lcm	n n2 -- n' (as of 17.05)
	Calculates the least-common-multiple of the tow numbers. Returns null if NaN or Inf etc are given.
lerp	lo hi t -- n (as of 23.06)
	<p>Linear interpolation between lo and hi based on the parameter t which should be in the range [0,1]. Converts inputs to regular floats; use the math/interpolate library if you need big floats or big ints.</p> <p>See also: n:lnerp</p>
ln	n -- n'
	<p>Natural logarithm (log base "e") of n .</p> <p>See also: n:exp</p>
ln1p	n -- n' (as of 19.09)
	Returns ln(n+1)
lnerp	lo hi t -- n (as of 23.06)
	<p>Logarithm interpolation between lo and hi based on the parameter t which should be in the range [0,1]. The range is positive non-zero numbers. Converts inputs to regular floats; use the math/interpolate library if you need big floats or big ints.</p> <p>See also: n:lerp</p>
logistic	n -- n' (as of 24.03)

word	sed/description
	<i>needs math/sigmoid</i> Returns the "logistic" sigmoid value of the input number, normalizing it to the range 0..1. The returned value is a 'float' though intermediate values may be promoted to 'bfloat' for accuracy reasons.
max	n n2 -- n'
	Returns the larger of two numbers. See also: n:min
median	a -- a n
	<i>needs math/median</i> Calculate the median (middle) value of an <i>array of numbers</i> S.
min	n n2 -- n'
	Returns the smaller of two numbers. See also: n:max
mod	n n2 -- n'
	Divides n by n2 , returning the integer remainder, which is always positive (even if the divisor is negative). See also: n:/mod n:/
neg	n -- n'
	Negate the number n : change its sign from positive to negative and vice-versa.
odd?	n -- n' (as of 18.06)
	Returns 0 if the number is even, 1 if it is odd.
perm	p k -- perm(p,k)
	<i>needs math/comb-perm</i> Returns the permutation of p items taken k at a time.
prime?	n -- n T (as of 17.05)

word	sed/description
	Determines <i>probabilistically</i> whether or not the number n is prime. If n is not a "big integer", returns null .
quantize	val quantum -- qval
	<p>DEFERRED <i>needs math/quantize</i></p> <p>Accept a <i>number</i> val and a <i>number</i> quantum, which is the boundary to which you wish to quantize. Returns qval, which is the quantized <i>number</i>. See also: n:quantize!.</p>
quantize!	n --
	<p><i>needs math/quantize</i></p> <p>Sets the kind of quantization to do. A value of n of 0=none, 1=up, 2=down, 3=nearest.</p>
r+	n -- (as of 18.05)
	Add the number to the number on the r-stack.
range	low high -- arr
	<p><i>needs math/range</i></p> <p>Returns an <i>array</i> of integer <i>numbers</i> in the range [low,high]. So 1 3 n:range returns [1,2,3]. Handles incorrect low vs. high by rearranging the values so the lower is first.</p>
rot32l	n1 cnt -- n2
	<p><i>needs math/rot</i></p> <p>Rotate the <i>number</i> n1 as a 32-bit integer, cnt times left. Bits passed "off the end" are replaced on the right.</p>
rot32r	n1 cnt -- n2
	<p><i>needs math/rot</i></p> <p>Rotate the <i>number</i> n1 as a 32-bit integer, cnt times right. Bits passed "off the end" are replaced on the left.</p>
round	n -- n'
	<p>Round to the nearest integer. By default, if the fractional part is 0.5 or greater, rounds up (down, if n is negative).</p> <p>Rounding algorithm can be controlled using n:rounding.</p> <p>See also: n:rounding n:trunc n:floor n:ceil</p>

word	sed/description
round2	n n2 -- n (as of 16.08)
	<p>Rounds the number according to n2 . If it is:</p> <ul style="list-style-type: none"> • 0: then this is exactly the same as n:round • <p>0: then n is rounded to that many digits to the right of the decimal</p> <ul style="list-style-type: none"> • <0: then that many significant digits is kept <p>So 123.456 2 round2 returns 123.460 , while 123.456 -2 round2 returns 120.000 .</p> <p>See also: n:round</p>
rounding	s -- (as of 19.09)
	<p>Takes a string of "NEAREST", "ZERO", "INF", or "NEGINF" and sets the rounding method used for floating-point operations.</p> <p>See also: n:round</p>
running-variance	a n -- a
	<p><i>needs math/mean</i></p> <p>Calculates mean and variance of a "stream" of numbers. TOS is an <i>array</i> which should be [0,0,0] initially, so that the values accumulate correctly. The value to add is under TOS. Returns the modified <i>array</i>. When finished collecting values, use n:running-variance-finalize</p>
running-variance-finalize	a -- a
	<p><i>needs math/mean</i></p> <p>Finalizes calculations done by n:running-variance, and returns an array with [sample variance, mean, count, population variance, stddev] The original accumulator array is not modified, so this can be used periodically on a data stream to gather interim values.</p>
sgn	n -- n'
	<p>Returns a -1 if n < 0 , 0 if it is zero, and 1 if n > 0 .</p> <p>See also: n:cmp n:=</p>

word	sed/description
shl	n n2 -- n'
	Returns the value of n "shifted left" n2 bits. So 1 2 shl returns 4 . See also: n:shr n:* n:/
shr	n n2 -- n'
	Returns the value of n "shifted right" n2 bits. Thus 4 2 shr returns 1 . See also: n:shl n:* n:/
sin	n -- n'
	Calculates the sine of n (radians). See also: n:cos n:tan n:asin
sincos	n -- n1 n2 (as of 24.04)
	Returns simultaneously the sin and cos of n: n1=sin(n), n2=cos(n)
sind	n -- sin(n)
	<i>needs math/trigd</i> Returns the sin of the angle n , specified in degrees.
sinh	n -- n' (as of 24.03)
	Returns the hyperbolic sine of the input number.
sqr	n -- n'
	Returns the square of the number, e.g. n dup n:* . See also: n:sqrt n:*
sqrt	n -- n'
	Returns the square-root of the number. See also: n:sqr n:/

word	sed/description
tan	n -- n'
	<p>Calculates the tangent of n (radians).</p> <p>See also: n:sin n:cos n:atan n:atan2</p>
tand	n -- tan(n)
	<p><i>needs math/trigd</i></p> <p>Returns the tangent of the angle n, specified in degrees.</p>
tanh	n -- n' (as of 24.03)
	Returns the hyperbolic tangent of the input number.
trunc	n -- n'
	<p>Truncate the number n , dropping the fractional part and returning integer part. This means rounding-towards-zero.</p> <p>See also: n:floor n:round n:ceil</p>
~=	n1 n2 eps -- T
	<p>Returns true if the two numbers are within eps of each other. Particularly useful for determining if "floating point" numbers are close to each other (rather than testing for absolute equality, which may fail otherwise).</p> <p>See also: n:= n:cmp</p>

OAuth

Namespace: **OAuth**

Description: Words for OAuth authentication

word	sed/description
auth-string	map url post? csec tsec -- sig

word	sed/description
	<i>needs net/oauth</i> Actually generate the OAuth signature from the parameters given. See <code>samples/net/twitter.8th</code> for an example.
gen-nonce	key token -- key token nonce
	<i>needs net/oauth</i> Generate an OAuth nonce from the given key and token strings .
params	key token nonce hashtype -- params
	<i>needs net/oauth</i> From the given key , token , nonce and hashtype , produce a params <i>map</i> .

Object

Namespace: **o**

Description: Object (OOP)

word	sed/description
!	<code>o s x -- o</code> (as of 21.01)
	Stores the item x by name in the object.
+	<code>o m -- o o o' -- o</code> (as of 21.02)
	Takes the variables in the map or object, and puts them in the object o .
+?	<code>o m o o o' o</code> (as of 21.02)
	Same as o:+ , but doesn't replace an already existing value.
???	<code>s --</code> (as of 21.01)
	Word invoked by default for an unknown object method.
@	<code>o s -- o x o a -- o a'</code> (as of 21.01)

word	sed/description
	Gets the item stored by name in the object. If passed an array, it will return an array of the items stored in the string keys passed in.
class	o -- o s (as of 21.01)
	Returns the 'class' of this object.
exec	o ... s -- (as of 21.01)
	<p>Invokes the "method" s in the object. Scans down the stack to find the first object, then looks-up the method in that object and its "supers" until it finds the word. If no object is on the stack, throws an exception.</p> <p>If the neither that object nor its supers has that method, then if a method called unknown-method exists (in the object or its supers), it is invoked with the name of the method which could not be found; otherwise, it is ignored.</p>
isa	o s -- o T (as of 21.01)
	Determines whether the object or its supers is of the kind denoted by the string.
method	o s w o m -- o (as of 21.01)
	Installs the word as a "method" for the object, under the name s . Invoke it with o:exec . Can also be invoked with a map of names->words to insert all at once.
mutate	o -- o (as of 21.01)
	Allows the given object to modify methods without affecting other members of its class.
new	o s -- o' (as of 21.01)
	<p>Creates a new object based on an existing object. The string is the 'type' of the object. If o is null then a blank object is created, with no methods. Otherwise, o' derives from o and contains its methods. If o has a method named ctor, it will be invoked on the new object prior to returning it.</p> <p>If the name s is null, the item will inherit the original object's name.</p> <p>"Methods" are object-specific words which are invoked by name, using o:exec.</p>
super	-- (as of 21.01)
	Within a method, invokes the object's super's method of the same name.

OS

Namespace: **os**

Description: OS-specific functionality

word	sed/description
POSIX	-- T (as of 24.03)
	Returns true if the OS is "POSIX" (true for all but Windows).
chroot	s -- T (as of 22.02)
	Changes the root directory accessible to the program to the given directory. Returns true if successful, or false and t:err? on failure. Typically requires program have root access. Not on Windows, obviously.
devname	x -- name (as of 19.06)
	Given an open file or a string naming a file, return the device name (EVIIOCGNAME). Linux and RPI only.
docker?	-- T (as of 22.06)
	<i>needs utils/docker</i> Returns true if we are running inside a Docker instance running a Linux OS, false if not (or if running in a Docker instance running some other OS).
env	-- m (as of 20.03)
	Returns a (read-only!) map with all the environment variables. Might return null , depending on OS.
lang	-- s (as of 19.05)
	Returns the language of the user's locale as reported by the OS. The return value should be a 2 or 3 letter language code (ISO 639-1 or ISO 639-2).
locales	-- a (as of 21.08)
	Returns an array of ["lang", "region"] corresponding to the user's currently preferred locales.
notify	m -- (as of 21.02)

word	sed/description
	<i>needs os/notify</i> Send a 'desktop notification'. m has keys:
power-state	-- (as of 22.01)
	Returns the best estimate of the system's power status. The map keys are "state" (one of "Battery", "No battery", "Charging" or "Charged"), "pct" (the percentage charge, and "secs" (the number of seconds estimated runtime on the charge). The latter two might be "-1" for "not available".
region	-- s (as of 19.05)
	Returns the region of the user's locale as reported by the OS. The return value should be a 2 letter country code (ISO 3166-1 alpha-2).
waitpid	n -- n'
	Wait for the process with pid n to finish, returning its return code. If the returned value is -1, then t:err? may have something useful to say.

PDF

Namespace: **pdf**

Description: PDF Document

word	sed/description
bezier	pdf pt1 pt2 pt3 pt4 wide color -- pdf (as of 22.03)
	<i>Professional version</i> Draws a cubic Bézier curve on the current PDF page from pt1 to pt2, with control points p3 and p4, in the given (point) width and color.
bezierq	pdf pt1 pt2 pt3 wide color -- pdf (as of 22.03)
	<i>Professional version</i> Draws a quadratic Bézier curve on the current PDF page from pt1 to pt2, with control point p3, in the given (point) width and color.
circle	pdf pt1 radius wide color fillcolor -- pdf (as of 22.03)

word	sed/description
	<i>Professional version</i> Draws a circle on the current PDF page centered at pt1 with radius, in the given (point) width, line color and fill color.
color	pdf clr -- pdf (as of 22.03)
	<i>Professional version</i> Sets the text color for subsequent pdf:text invocations.
ellipse	pdf pt1 pt2 wide color fillcolor -- pdf (as of 22.03)
	<i>Professional version</i> Draws an ellipse on the current PDF page centered at pt1 with x,y radii pt2, in the given (point) width, line color and fill color.
font	pdf s -- pdf (as of 22.03)
	<i>Professional version</i> Sets the font to use, currently restricted to the standard PDF fonts: "Courier", "Courier-Bold", "Courier-BoldOblique", "Helvetica", "Helvetica-Bold", "Helvetica-BoldOblique", "Helvetica-Oblique", "Times-Roman", "Times-Bold", "Times-Italic", "Times-BoldItalic", "Symbol", or "ZapfDingbats".
img	pdf pt1 pt2 img -- pdf (as of 22.03)
	<i>Professional version</i> Draws an image at pt1, with size pt2. The image could be an img or a file name string. An img is any 8th supported format. Supported formats for an image file are JPEG, PNG, PPM, PGM, and BMP.
line	pdf pt1 pt2 wide color -- pdf (as of 22.03)
	<i>Professional version</i> Draws a line on the current PDF page from pt1 to pt2, in the given (point) width and color.
new	m -- pdf (as of 22.03)

word	sed/description															
	<p><i>Professional version</i></p> <p>Create a new PDF document in memory, with the options given in the map. The map may contain the meta-data keys creator , producer , title , author , subject , and date . Additional keys may be:</p> <table><tr><th>key</th><th>type</th><th>description</th></tr><tr><td>size</td><td>pt</td><td>width and height of PDF document (e.g. page size)</td></tr><tr><td>text-color</td><td>clr</td><td>the color to use for drawing text (black, by default)</td></tr><tr><td>text-size</td><td>n</td><td>text size in points (12 by default)</td></tr><tr><td>font</td><td>s</td><td>the font to use (defaults to "Times-Roman")</td></tr></table>	key	type	description	size	pt	width and height of PDF document (e.g. page size)	text-color	clr	the color to use for drawing text (black, by default)	text-size	n	text size in points (12 by default)	font	s	the font to use (defaults to "Times-Roman")
key	type	description														
size	pt	width and height of PDF document (e.g. page size)														
text-color	clr	the color to use for drawing text (black, by default)														
text-size	n	text size in points (12 by default)														
font	s	the font to use (defaults to "Times-Roman")														
page	pdf -- pdf (as of 22.03)															
	<p><i>Professional version</i></p> <p>Creates a new blank page in the PDF and makes it the current page for output.</p>															
page-size	pdf -- pdf wide high (as of 22.03)															
	<p><i>Professional version</i></p> <p>Gets the size of the PDF document current page in points.</p>															
rect	pdf pt1 pt2 wide color fillcolor -- pdf (as of 22.03)															
	<p><i>Professional version</i></p> <p>Draws a rectangle from pt1, with width pt2.x and height pt2.y, border width wide in color. If fillcolor` is null, don't fill the rectangle; otherwise, fill with that color.</p>															
save	pdf s -- pdf pdf f -- pdf pdf null -- pdf (as of 22.03)															
	<p><i>Professional version</i></p> <p>Save the PDF. If null , writes to stdout . String or file writes to the named or open file.</p>															
size	pdf -- pdf wide high (as of 22.03)															
	<p><i>Professional version</i></p> <p>Gets the size of the PDF document in points.</p>															
text	pdf s xofs yofs -- pdf (as of 22.03)															
	<p><i>Professional version</i></p> <p>Write the string at position (xofs,yofs) on the current page in the PDF.</p>															

word	sed/description
text-rotate	pdf s x y angle -- pdf n (as of 23.06)
	<i>Professional version</i> Same as pdf:text , but rotates through the angle (in radians)
text-size	pdf n -- pdf (as of 22.03)
	<i>Professional version</i> Sets the size of the font to use.
text-width	pdf s -- pdf n (as of 22.03)
	<i>Professional version</i> Calculates the width of the given string using the current font and size. The number returned is points. Returns 0 and sets t:err? if there was an error.
text-wrap	pdf s x y w align -- pdf n (as of 22.03)
	<i>Professional version</i> Same as pdf:text , but wraps the text at w points wide, with an text alignment (as listed in the pdf/utlis library). Returns the height of the output text (0 on error and t:err? is set).
text-wrap-rotate	pdf s x y w align angle -- pdf n (as of 23.06)
	<i>Professional version</i> Same as pdf:text-wrap , but rotates through the angle (in radians)

Pointer

Namespace: **ptr**

Description: A pointer value, used with the FFI

word	sed/description
cast	ptr s -- ptr ptr n s -- ptr (as of 17.07)

word	sed/description
	<p>When G:unpack is given a "P" to unpack, it doesn't know what the pointer refers to. ptr:cast lets you tell it what the value is, using the same characters as used in ptr:pack . If s is "b", then you must <i>also</i> provide the size n in bytes.</p> <p><i>Note:</i> This word has no effect on a ptr whose type is actually known (you cannot coerce a "string" to a "number" for instance).</p> <p>See also: G:pack G:unpack ptr:pack ptr:unpack ptr:unpack_orig ptr:len</p>
deref	-- (as of 24.01)
	<p>Assuming the ptr is a pointer to something (e.g. void **), returns the "n'th" item it points to, e.g. ptr[n]). May be given a number instead of a ptr, in which case it's assumed to be a raw memory address.</p>
len	ptr -- ptr n (as of 17.07)
	<p>Returns the number of bytes allotted the contents of this ptr*. For a string or buffer, will be the allocated size.</p> <p>See also: G:pack G:unpack ptr:pack ptr:unpack ptr:unpack ptr:unpack_orig</p>
null?	ptr -- ptr T (as of 21.03)
	<p>Returns true if the value of the pointer is 0.</p>
pack	x s -- ptr (as of 17.07)
	<p>Converts an 8th type x into a ptr suitable for use in the FFI. The string s is one of:</p> <ul style="list-style-type: none"> • "i" for "int" • "u" for "unsigned int" • "l" for "long" • "L" for "unsigned long" • "f" for "float" • "d" for "double" • "p" for "ptr" • "w" for "word" • "z" for "char **" • "b" for a buffer (could be a buffer created by G:pack if you need to pass a pointer to a structure). <p>See also: G:pack G:unpack ptr:unpack ptr:cast ptr:unpack_orig ptr:len</p>
unpack	ptr -- x (as of 17.07)

word	sed/description
	<p>Inverse of ptr:pack . Convert a ptr back into a normal 8th type.</p> <p>See also: G:pack G:unpack ptr:pack ptr:cast ptr:unpack_orig ptr:len</p>
unpack_orig	ptr -- x (as of 17.07)
	<p>Same as ptr:unpack , but returns the original pointer instead of the modified value. For instance, in the "iconv()" function, the destination pointer is modified, making it impossible to retrieve the original value because a buffer has a fixed address which is immutable.</p> <p>See also: G:pack G:unpack ptr:pack ptr:unpack ptr:unpack ptr:len</p>

PubSub

Namespace: **pubsub**

Description: Publish/Subscribe messaging framework

word	sed/description
publish	t msg --
	<p><i>needs utils/pubsub</i></p> <p>Enqueues the message msg (which can be any data type) to subscribers of the topic t (a <i>string</i>). The queued message will be dispatched to all subscribers of that topic in the order they subscribed, on the task of the message dispatcher.</p>
qsize	n --
	<p><i>needs utils/pubsub</i></p> <p>Sets the size of the queue used by the subscribe task. The default is 100</p>
subscribe	t wrd --
	<p><i>needs utils/pubsub</i></p> <p>Subscribes the <i>word</i> wrd to the topic given by the <i>string</i> t. wrd will be invoked whenever a message is published to that topic. Subscriber SED: \ topic message --</p>

Queue

Namespace: **q**

Description: FIFO Queue

word	sed/description
+	q q2 -- q
	<p>Moves as many items as will fit from the queue q2 to the queue q , popping them from one and pushing to the other. If q is not big enough to hold the data from q2 , then it will throw an exception unless q:throwing was false for q .</p> <p>Modifies both queues.</p>
clear	q -- q
	<p>Remove all items from the queue.</p>
len	q -- q n
	<p>Returns the number of items in the queue.</p>
new	n -- q
	<p>Create a new queue with a maximum capacity n , which must be a positive number. If a negative or zero size is given, returns null . Limited only by available memory.</p> <p>See also: q:push q:pop</p>
notify	q -- q (as of 17.07)
	<p>Notifies anyone waiting on that queue that it should awaken and process the data.</p> <p>See also: G:sleep q:wait t:q-wait t:q-notify</p>
overwrite	q T -- q
	<p>Set the queue to "overwrite" mode (if T is true), so that it acts as a <i>circular buffer</i> rather than a queue. This means that continued q:push without q:pop will save the last N items pushed, where N is the size of the queue. If T is false , restore normal queue behavior.</p> <p>See also: q:push q:pop</p>

word	sed/description
peek	q -- q x
	<p>Same as q:pop but does not remove the item from the queue.</p> <p>See also: q:pop q:push</p>
pick	q ix -- q x (as of 17.04)
	<p>Returns the item x at the index in the queue. The front of the queue is 0, item under is 1, etc. If you try to pick beyond the limits of the queue, it will throw an exception.</p>
pop	q -- q x
	<p>Pop the first item off the queue q , removing it from the queue.</p> <p>See also: q:push q:peek</p>
push	q x -- q
	<p>Push the item x into the queue q making it the last item in the queue.</p> <p>See also: q:pop q:peek</p>
remove	q x -- q n (as of 20.01)
	<p>Removes all instances of the item x from the queue. Returns the number of instances found and removed.</p>
shift	q -- q x
	<p>Remove the item x from the back of the queue and put it on TOS. This makes the queue one element shorter. Like q:pop but from the back of the queue.</p> <p>See also: q:slide q:pop q:push</p>
size	q -- q n (as of 16.11)
	<p>Returns the size of the given queue, which is the maximum number of items the queue may contain.</p>
slide	q x -- q

word	sed/description
	<p>Add the item x to the front of the queue. Like q:push but at the front of the queue, adding one more element to the queue (q:push puts the item at the back).</p> <p>See also: q:shift q:pop q:push</p>
throwing	q T -- q
	<p>If true, the queue q will throw an exception on failure to push or pop. By default, queues throw an exception, because popping from an empty queue is most likely an error, and null is a valid value.</p>
wait	q n -- q (as of 17.07)
	<p>Waits n milliseconds for a notification on the queue. If n is negative, waits forever. Otherwise, it exits the wait once q:notify is invoked on the queue, or if the timeout expires.</p> <p>See also: G:sleep q:notify t:q-wait t:q-notify</p>

Rational

Namespace: **rat**

Description: Rational big-number processing

word	sed/description
*	r1 r2 -- r1*r2 (as of 19.04)
	<p><i>needs math/rational</i></p> <p>Multiply two rationals</p>
+	r1 r2 -- r1+r2 (as of 19.04)
	<p><i>needs math/rational</i></p> <p>Add two rationals</p>
-	r1 r2 -- r1-r2 (as of 19.04)
	<p><i>needs math/rational</i></p> <p>Subtract two rationals</p>
/	r1 r2 -- r1/r2 (as of 19.04)

word	sed/description
	<i>needs math/rational</i> Divide two rationals
>n	r -- n (as of 20.07)
	<i>needs math/rational</i> Convert a rational to a regular <i>number</i>
>s	r -- s (as of 19.04)
	<i>needs math/rational</i> Convert the rational number to its <i>string</i> representation.
new	num denom -- rat (as of 19.04)
	<i>needs math/rational</i> Given a numerator and a denominator in integral values (not floats!), produce a new "rational number"
proper	r -- n r (as of 19.04)
	<i>needs math/rational</i> Given a (possibly) "improper" rational, e.g. where the numerator is greater than the denominator, convert it to a "proper" pair of whole number and rational remainder. Either or both of the pair may be 0.

Rectangles and points

Namespace: **rect**

Description: Rectangles

word	sed/description
!	rect ix n -- rect (as of 22.03)
	For a rect or pt, sets the value n at index ix , where 0=x, 1=y, 2=w, 3=h. Does nothing if the index is out of range.
/high	rect n -- rect1 rect2 (as of 20.01)

word	sed/description
	<p>Splits the rectangle into two such that the sum of the heights of the new ones is the same as the height of the original. If n is < 1, it's a ratio of how much of the original should be in the first. Otherwise, it's a pixel size.</p> <p>See also: rect:/wide</p>
/wide	rect n -- rect1 rect2 (as of 20.01)
	<p>Splits the rectangle into two such that the sum of the widths of the new ones is the same as the width of the original. If n is < 1, it's a ratio of how much of the original should be in the first. Otherwise, it's a pixel size.</p> <p>See also: rect:/high</p>
=	rect1 rect2 -- rect1 rect2 T (as of 22.02)
	Return true if the rectangles are the same.
>a	X -- a (as of 22.02)
	Converts an X containing the internal rectangle format to an array [x,y,w,h].
>pts	rect -- pt1 pt2 (as of 20.01)
	Converts the rectangle to its start and end points (upperleft, bottomright) .
>pts4	rect -- pt1 pt2 pt3 pt4 (as of 22.03)
	Converts the rectangle to its four corner points (upper left, upper right, lower left, lower right) .
@	rect ix -- rect x rect a -- rect a' (as of 22.03)
	For a rect or pt: if given a number, retrieves the value at index 'x', where 0=xpos, 1=ypos, 2=wide, 3=high. If the index is out of range, 0 is returned. If passed an array, returns an array of the corresponding values.
center	rect1 rect2 -- rect3 (as of 22.03)
	Creates a new rectangle which is rect2 centered on rect1 .
center-pt	rect -- rect pt (as of 20.01)
	Returns a point in the center of the rectangle.

word	sed/description
intersect	rect1 rect2 -- rect' (as of 20.01)
	Returns the "intersection" of the two rects passed in. If they don't intersect, a rectangle of [0,0,0,0] is returned.
new	a -- X (as of 22.02)
	Converts a rectangle which is an array of [x,y,w,h] to internal format.
new-pt	a -- X (as of 22.02)
	Converts a point which is an array [x,y] to internal format.
ofs	rect pt -- rect' (as of 21.01)
	Adds the point to the rectangle, moving the rectangle's upper-left by pt .
open	rect -- xpos ypos wide high (as of 22.03)
	Use instead of a:open , because a rect may be a native value rather than an array.
pad	rect pt -- rect2 (as of 20.01)
	Pads the rectangle with the [x,y] values from pt . The x value is added to the "x" and "w" components of the rectangle, the y value is added to the "y" and "h" components.
pos	rect -- pt (as of 20.01)
	Converts a rectangle to a point which is its [x,y] position.
pt-open	pt -- xpos ypos (as of 22.03)
	Use instead of a:open , because a pt may be a native value rather than an array.
pt>a	X -- a (as of 22.02)
	Converts an X containing the internal format to a point which is an array [x,y].
pt>rect	pt -- rect (as of 22.07)
	Converts the point (x,y) into the rectangle (0,0,x,y) .

word	sed/description
pts>	pt1 pt2 -- rect (as of 20.01)
	Returns a rectangle starting at point pt1 and ending at point pt2 . Assumes pt1 is left and above pt2 .
restrict	rect1 rect2 -- rect2 (as of 24.04)
	Modifies rect2 so that it is completely within rect1 , moving it as appropriate and modifying its size, if necessary.
shrink	rect n -- rect' rect a -- rect' (as of 20.01)
	Shrinks (if n is postive) or grows the rectangle by n in all directions. If given an array, it is values by which the rectangle should shrink on each side [left,top,right,bottom]. If n is a number in the range (0..1) then it shrinks the rectangle by that percentage of the width and height.
size	rect -- pt (as of 20.01)
	Converts a rectangle to an point [w,h] which is its size.
union	rect1 rect2 -- rect' (as of 20.01)
	Returns the "union" of the two rects passed in: e.g., the bounding box containing both rectangles.

Regex

Namespace: **r**

Description: PCRE regular-expression

word	sed/description
++match	r -- r n
	Same as r:++match , but matches after the full length of the previous match (instead of just one character past, as r:++match does).
	See also: r:new r:match r:+match r:@
+/	s r -- a (as of 18.02)

word	sed/description
	<p>Same as r:/ but skips past the end of the previous match. So that e.g. "1.2 abc 3.4 def" /([1-9][0-9.]+)/ r:+/ gives ["1.2", "3.4"] , whereas r:/ gives ["1.2", "2", "3.4", "4"] .</p> <p>See also: r:/</p>
+match	r -- r n
	<p>Match the regex r against the previously matched string, beginning from just after the previous match. Returns a number n which is the number of matches, or null if there was no prior match.</p> <p>See also: r:new r:match r:++match r:@</p>
/	s r -- a
	<p>Splits the string on a regex and returns an array containing the matches, or null if there were no matches.</p> <p>Ex: "1234567" /(\d\d\d)(\d\d\d\d)/ r:/ returns ["123","4567"] .</p> <p>See also: r:@ r:match</p>
@	r n -- r s r a -- r a'
	<p>Returns a string which is match number n from the regex r , or null if there was no match. If given an array of numbers, returns an array of those matches.</p> <p><i>Note:</i> do not confuse with G:r@ !</p> <p>See also: r:new r:match r:+match r:++match</p>
len	r -- r n (as of 17.02)
	Returns the number of matches currently matched by the regex.
match	x r -- r n
	<p>Match the regex r against the string or regex x . Returns the number n of matches, which may be 0 if there were no matches, or null if x was neither a string nor a previously matched regex.</p> <p>See also: r:new r:+match r:++match r:@</p>
match[]	x r -- a (as of 25.06)
	<p><i>needs regex/match</i></p> <p>Returns an array of all matched values, or null if there were no matches.</p>

word	sed/description
matchall[]	x r -- a (as of 25.06)
	<i>needs regex/match</i> Same as r:match[] , but returns all matches of the regex over the entire string or regex x .
new	s -- r
	Create a new regex from the string, or null if it failed to create one. See also: r:match r:+match r:@
rx	r -- r s (as of 17.06)
	Returns the string which is the regular-expression represented by the regex, or null if no such string exists.
str	r -- r s (as of 17.06)
	Returns the string against which the regex r was matched, or null if no such string exists.

Serial

Namespace: **sio**

Description: Serial I/O

word	sed/description
close	sio --
	<i>Hobbyist version</i> Closes a previously opened sio, and terminates any connections. See also: sio:open sio:opts@ sio:opts! sio:enum sio:read sio:write
enum	-- a
	<i>Hobbyist version</i> Returns an array of paths to valid serial devices. One of these may be passed to sio:open . Returns null if there are no valid devices. See also: sio:open sio:opts@ sio:opts! sio:write sio:read sio:close

word	sed/description
open	s -- sio
	<p><i>Hobbyist version</i></p> <p>Returns a sio which represents an opened serial port named by the string, or null if it was unable to do so.</p> <p>See also: sio:opts@ sio:opts! sio:read sio:write sio:enum sio:close</p>
opts!	sio m -- sio
	<p><i>Hobbyist version</i></p> <p>Sets the sio's options based on the map (as returned by sio:opts@).</p> <p>See also: sio:open sio:opts@ sio:read sio:write sio:enum sio:close</p>
opts@	sio -- sio m
	<p><i>Hobbyist version</i></p> <p>Returns a map containing the sio's current settings. All known settings will be returned as a key in the map, and all values are either numeric or boolean.</p> <p>See also: sio:open sio:opts! sio:read sio:write sio:enum sio:close</p>
read	sio sb n -- sio sb n'
	<p><i>Hobbyist version</i></p> <p>Reads n bytes from the sio, appending to the string or buffer sb . The number of bytes read is left on TOS, or null if there was a problem. The original sb remains on the stack and is (possibly) modified by the read.</p> <p>See also: sio:open sio:opts@ sio:opts! sio:write sio:enum sio:close</p>
write	sio sb -- sio n
	<p><i>Hobbyist version</i></p> <p>Writes the string or buffer sb to the sio. The number of bytes written is left on TOS, or null if there was a problem.</p> <p>See also: sio:open sio:opts@ sio:opts! sio:enum sio:read sio:close</p>

SMTP

Namespace: **smtp**

Description: SMTP access words

word	sed/description
new	opts -- smtp T
	<p><i>needs net/smtp</i></p> <p>Takes a <i>map</i> opts with network options and returns a new <i>smtp</i> item which is used to communicate with an SMTP server. Immediately tries to connect to the given server, returns true if successful.</p>
send	smtp msg -- smtp T
	<p><i>needs net/smtp</i></p> <p>Sends the message msg to the SMTP server smtp. The message is a <i>map</i> with "from", "to", "cc", "bcc" keys -- which may be single <i>strings</i> or <i>arrays</i> of <i>string</i>. "subject", "body" are required and must be <i>strings</i>. Returns true on success, or false and sets the "err" key.</p>

SOAP

Namespace: **SOAP**

Description: SOAP

word	sed/description
call	map -- xml errs
	<p><i>needs net/soap</i></p> <p>Performs a SOAP call based on the information in the given <i>map</i>. See samples/net/soap.8th for how it's used.</p>

Solver

Namespace: **slv**

Description: Constraint solver

word	sed/description
@	slv x -- slv (as of 21.01)
	<p>Gets the value of the variable x ; x is either a string or a number.</p>
auto	slv T -- slv (as of 21.01)

word	sed/description						
	If true , turn automatic update of variables on.						
build	m -- slv (as of 21.01)						
	Creates a new solver set up according to the map. Keys are: <table border="1"> <thead> <tr> <th>key</th><th>description</th></tr> </thead> <tbody> <tr> <td>vars</td><td>map; each key is a variable name, whose value is the initial value</td></tr> <tr> <td>constraints</td><td>array of strings with calculation constraints</td></tr> </tbody> </table>	key	description	vars	map; each key is a variable name, whose value is the initial value	constraints	array of strings with calculation constraints
key	description						
vars	map; each key is a variable name, whose value is the initial value						
constraints	array of strings with calculation constraints						
constant	slv x n -- slv (as of 21.01)						
	Adds a constant n to the constraint x . x is either a string or a number.						
constraint	slv n -- slv n' (as of 21.01)						
	Creates a new constraint for the constraint-solver. Returns the index of the constraint. The number n is the strength of the constraint, as defined in the slv/constraint library.						
dump	slv -- slv (as of 21.01)						
	For debugging: dumps the state of the solver.						
edit	slv s str -- slv slv n str -- slv (as of 21.01)						
	Adds an edit to the variable, with strength str .						
named-variable	slv s -- slv n (as of 21.01)						
	Creates a new named variable for the constraint-solver. Returns the index of the variable.						
new	-- slv (as of 21.01)						
	Creates a new constraint-solver.						
relation	slv x n -- slv (as of 21.01)						
	Adds a relation n to the constraint x . One of 1 (<=), 2 (=), or 3 (>=); x is either a string or a number.						

word	sed/description
reset	slv T -- slv (as of 21.01)
	Resets the solver. If T is true , remove all the constraints.
suggest	slv x n -- slv slv a1 a2 -- slv (as of 21.01)
	Adds a suggested value for the variable x , using the value n . If x is a string or a number, n must be a number. If x is an array, then it must contain indices or names of the variables whose values are in the array n .
term	slv x1 x2 n -- slv slv n1 n2 n -- slv (as of 21.01)
	Adds a term to the constraint x1 , using the variable x2 and a multiplier n . The "x" values can be strings (the names of the constraint or variable) or a number (index of the constraint or variable).
update	slv -- slv (as of 21.01)
	Makes the slv update the variables, if slv:auto was not set true .
v[]	slv -- slv a (as of 21.01)
	Returns an array of the values of all variables in the solver, in order of insertion.
variable	slv -- slv n (as of 21.01)
	Creates a new variable for the constraint-solver. Returns the index of the variable.
v{}	slv -- slv m (as of 21.01)
	Returns a map of the values of all named variables in the solver.

Sound

Namespace: **snd**

Description: Playing and recording sounds

word	sed/description
apply-filter	x -- snd2 (as of 18.06)

word	sed/description
	<p><i>Hobbyist version</i></p> <p>Applies the filter created by snd:filter to the playback device. If null, all filters are removed. Multiple filters may be simultaneously applied.</p> <p>See also: snd:filter</p>
devices?	-- a (as of 18.06)
	<p><i>Hobbyist version</i></p> <p>Returns an array of supported sound devices on the current system.</p>
end-record	X -- (as of 18.03)
	<p><i>Hobbyist version</i></p> <p>Stops the audio recording.</p> <p>See also: snd:record</p>
filter	m -- X (as of 18.06)
	<p><i>Hobbyist version</i></p> <p>Creates a new biquad filter based on the parameters given in the map. The result may be used with snd:apply-filter to affect playback.</p> <p>Possible keys are:</p> <ul style="list-style-type: none"> • "kind", which is one of the following: • "biquad" : values are "a0", "a1", "a2", "b0", "b1", "b2" -- coefficients for a biquad filter (see miniaudio) • "low1" : low-pass filter: "freq" • "low2" : low-pass filter: "freq", "Q" • "low" : low-pass filter: "freq", "order" • "hi1" : high-pass filter: "freq" • "hi2" : high-pass filter: "freq", "Q" • "hi" : high-pass filter: "freq", "order" • "notch" : notch filter: "freq", "Q" • "band" : band-pass filter: "freq", "order" • "peak" : band-pass filter: "freq", "gain", "Q" • "loshelf" : low-shelf filter: "freq", "gain", "slope" • "hishelf" : high-shelf filter: "freq", "gain", "slope" <p>See also: snd:apply-filter</p>
freq	snd n -- snd (as of 18.06)

word	sed/description
	<p><i>Hobbyist version</i></p> <p>For a snd created with a frequency (a <i>number</i>, generating a continuous sine wave tone), this changes its frequency.</p>
gain	<p>snd n -- (as of 18.06)</p> <p><i>Hobbyist version</i></p> <p>Sets the gain of the snd to n , which defaults to 1.</p>
gain?	<p>snd -- snd n (as of 18.06)</p> <p><i>Hobbyist version</i></p> <p>Returns the gain of the snd.</p>
init	<p>-- T (as of 19.09)</p> <p><i>Hobbyist version</i></p> <p>Ensures the sound system is running. Returns false if not. See "libs/snd/loaded" for details.</p>
len	<p>snd -- snd n</p> <p><i>Hobbyist version</i></p> <p>Returns the length of the sound in seconds. This may be used on a snd whether it is playing currently or not.</p>
loop	<p>snd T -- snd</p> <p><i>Hobbyist version</i></p> <p>If T is true , makes snd repeat continuously. Otherwise, makes it play just once (the default).</p>
loop?	<p>snd -- snd T (as of 20.01)</p> <p><i>Hobbyist version</i></p> <p>Returns true if the sound is supposed to loop when done playing.</p>
mix	<p>snd -- (as of 18.06)</p> <p><i>Hobbyist version</i></p> <p>Plays the snd, mixing it with other sounds in the mixer queue.</p>
new	<p>x -- snd</p>

word	sed/description
	<p><i>Hobbyist version</i></p> <p>Creates a new snd. If passed a:</p> <ul style="list-style-type: none"> • string: it must be a valid audio file-name • buffer: it must be valid audio data • number: it is a frequency for a continuous sine-wave tone • array: it is [kind,amplitude,frequency] for a tone, where 'kind' is 0 for sine, 1 for square, 2 for triangle, 3 for sawtooth. Or [kind,amplitude] where 0 is 'white-noise', 1 is 'pink-noise', and 2 is 'brownian-noise'. • word: it is a custom sound which is implemented by that word, whose SED is n1 n2 -- b, where n1 is the number of 32-bit float frames required, n2 is the number of channels, and b is a buffer containing the floating-point sound data. <p>To play the sound, invoke snd:play, passing it the snd item. Returns null if the sound format is not supported or the sound is otherwise unplayable.</p> <p><i>Note:</i> you must invoke requires sound first!</p> <p><i>Note:</i> only WAV, MP3, FLAC, and OGG formats are supported.</p> <p>See also: snd:play snd:stop snd:volume</p>
pause	<p>snd T -- snd (as of 18.08)</p>
	<p><i>Hobbyist version</i></p> <p>If snd is playing, stops the playback if *T is <i>true</i>, otherwise resumes playback of the sound. When snd:play or snd:mix is invoked on this sound, it will resume playback from where it stopped. This only works with sounds played from files.</p>
play	<p>snd --</p>
	<p><i>Hobbyist version</i></p> <p>Plays the snd created by snd:new.</p> <p>See also: snd:volume snd:new snd:stop</p>
played	<p>snd -- (as of 18.03)</p>
	<p>DEFERRED</p> <p><i>Hobbyist version</i></p> <p>Invoked when the snd has finished playing.</p>
rate	<p>snd n -- snd</p>
	<p><i>Hobbyist version</i></p> <p>Set the sample rate of the snd.</p>

word	sed/description																					
ready?	-- n1 n2 (as of 20.01)																					
	<i>Hobbyist version</i> Returns the number of sounds: currently playing n2 , and enqueued to be played n1 . Returns null if snd:init hasn't been invoked yet.																					
record	s -- X m -- X (as of 18.03)																					
	<i>Hobbyist version</i> Starts recording audio at 44.1KHz, 32-bit floating pt, WAV format, into the file named by the string. The return value may be used to control the recording. When X is refcounted to 0, the recording will stop. You may also use snd:end-record to explicitly stop recording. null will be returned if the recording could not be started for any reason. If given a map, then it contains some of the keys: <table><tr><th>key</th><th>type</th><th>description</th></tr><tr><td>data</td><td>w: m b --</td><td>invoked for each captured buffer of 4-byte IEEE float data</td></tr><tr><td>done</td><td>w: m --</td><td>invoked when snd:end-record is</td></tr><tr><td>init</td><td>w: m --</td><td>invoked before starting recording</td></tr><tr><td>channels</td><td>n</td><td>How many channels of audio to record (default: 2)</td></tr><tr><td>format</td><td>n</td><td>0-5, one of the miniaudio audio formats</td></tr><tr><td>rate</td><td>n</td><td>sample ratge</td></tr></table> See also: snd:end-record	key	type	description	data	w: m b --	invoked for each captured buffer of 4-byte IEEE float data	done	w: m --	invoked when snd:end-record is	init	w: m --	invoked before starting recording	channels	n	How many channels of audio to record (default: 2)	format	n	0-5, one of the miniaudio audio formats	rate	n	sample ratge
key	type	description																				
data	w: m b --	invoked for each captured buffer of 4-byte IEEE float data																				
done	w: m --	invoked when snd:end-record is																				
init	w: m --	invoked before starting recording																				
channels	n	How many channels of audio to record (default: 2)																				
format	n	0-5, one of the miniaudio audio formats																				
rate	n	sample ratge																				
resume	-- (as of 20.01)																					
	<i>Hobbyist version</i> Resumes sound playback.																					
seek	snd n -- snd (as of 18.08)																					
	<i>Hobbyist version</i> For a sound which is initialized from a file or buffer, sets the next play position to the n 'th PCM frame.																					
stop	snd --																					
	<i>Hobbyist version</i> Stops that sound from being played. See also: snd:volume snd:new snd:play																					

word	sed/description
stopall	-- (as of 18.03)
	<p><i>Hobbyist version</i></p> <p>Stops all sound playback.</p> <p>See also: snd:resume</p>
volume	n --
	<p><i>Hobbyist version</i></p> <p>Sets the system sound volume to a value between 0 and 1, where 1 is 'full volume'.</p> <p>See also: snd:new snd:play snd:volume?</p>
volume?	-- n
	<p><i>Hobbyist version</i></p> <p>Returns the system sound volume as a number between 0 and 1, where 1 is 'full volume'.</p> <p>See also: snd:volume snd:new snd:play</p>

Stack

Namespace: **st**

Description: Fixed-size stack

word	sed/description
+	st st2 -- st null st2 --
	<p>Moves as many items as will fit from the stack st2 to the stack st , popping them from one and pushing to the other. If st is null , the data-stack will be the destination.</p> <p>If st is not big enough to hold the data from st2 , then it will throw an exception unless st:throwing was false for st . If st is not null , it will remain on TOS; otherwise it is removed.</p> <p>Modifies both stacks.</p>
.	st --

word	sed/description
	<p>Print the top st:dot-depth items of the given stack, in the same manner that G:.s does for the data stack.</p> <p>See also: G:.s st:dot-depth</p>
clear	<p>st -- st</p> <p>Removes all the items currently on the given stack. Functionally equivalent to invoking repeat st:pop drop st:len while! .</p> <p><i>Note:</i> to clear the main data-stack, use G:reset rather than st:clear !</p> <p>See also: G:reset</p>
dot-depth	<p>-- v (as of 24.02)</p>
	<p>Variable containing the maximum depth of stack items to print with st:. (and .s and .r) as well as st:list .</p> <p>See also: G:.s st:.</p>
len	<p>st -- st n</p> <p>Returns the number of items currently on the given stack.</p>
list	<p>st n -- a (as of 24.02)</p> <p>Returns an array of strings, at most n long, of items on the given stack. Used by .s etc. to print the stack. Useful if you want to G:log the current stack. The strings are "escaped" so that e.g. a "\r" character displays as \r rather than causing truncation of the output.</p> <p>If the array is not empty, the first item is a number, the depth of that stack.</p> <p>See also: G:.s st:.</p>
ndrop	<p>st n -- st (as of 18.03)</p> <p>Drops n items from the given stack.</p>
new	<p>n -- st</p> <p>Create a new stack of capacity n . The value given must be a positive non-zero number; if it is not, null is returned.</p>

word	sed/description
op!	st w -- (as of 18.04)
	<p>Invokes w on the TOS of stack, replacing it. Any operands to w should appear in their normal stack order under st. The SED for w is x--x'.</p> <p>See also: G:rop!</p>
peek	st -- st x
	<p>Peeks at the top of the given stack without removing the top item. Returns that item from the stack or null if empty.</p> <p>See also: st:throwing st:push st:pop</p>
pick	st n -- st x
	<p>Same as G:pick for the specified stack.</p> <p>See also: G:pick G:rpick</p>
pop	st -- st x
	<p>Pops the item x from the given stack. If the stack is empty, throw an exception, unless st:throwing was set false, in which case it returns null.</p> <p>See also: st:throwing st:push st:peek</p>
push	st x -- st
	<p>Push the item x onto the given stack. If the stack is full, throw an exception unless st:throwing was set false, in which case it fails quietly.</p> <p>See also: st:throwing st:pop st:peek</p>
roll	st n -- st
	<p>Same semantics as G:roll for the specified stack.</p> <p>See also: G:roll G:rroll</p>
shift	st -- st x

word	sed/description
	<p>Remove the item x from the bottom of the stack and put it on TOS. This makes the stack one element shorter, and shifts all remaining elements one lower. Like st:pop but from the bottom of the stack.</p> <p>See also: st:slide st:pop st:push</p>
size	st -- st n (as of 16.11)
	Returns the size of the given stack, which is the maximum number of items the stack may contain.
slide	st x -- st
	<p>Add the item x to the bottom of the stack, moving all remaining items up one index. Like st:push but at the bottom of the stack.</p> <p>See also: st:shift st:pop st:push</p>
swap	st -- st
	<p>Same as G:swap , but for the specified stack. That is, it exchanges TOS of the given stack with the item below it.</p> <p>See also: G:swap</p>
throwing	st T -- st
	If true , make the stack st throw an exception on failure to push or pop; otherwise, the stack will return null on error. By default stacks throw on failure.

String

Namespace: **s**

Description: Arrays of UTF-8 encoded characters

word	sed/description
!	s ix n -- s'

word	sed/description
	<p>Replace the character at index ix in the string with the Unicode character n . If ix is greater than s:len-1 , nothing will happen. If it is negative, the character that is that many characters from the end of the string will be modified (-1 is the last character).</p> <p>Ex: "abcdef" 3 'X s:! results in "abcXef"</p> <p>See also: s:@</p>
*	s n -- s'
	<p>Creates a new string by replicating the initial string s n times. If n is 0 or negative, an empty string is returned.</p> <p>See also: s:+</p>
+	s s2 -- s' s n -- s'
	<p>Appends the string s2 to s . If a number is given instead, it is treated as a Unicode character.</p> <p>See also: s:append</p>
-	s ix n -- s'
	<p>Remove n characters from the string s , starting at <i>character</i> offset ix . Both n and ix must be positive numbers, and if the combined length is greater than the string length then nothing is done.</p>
/	s n -- a s a1 -- a2 s pat -- a s null -- a
	<p>Split the string s at n , splitting into an array [head,tail] at that character index (starting from 0).</p> <p>If an array of numbers a1 is given, splits into an array of strings with that many characters each, repeating the last value as long as there is still something to split (e.g. "12345" [2] s:/ will give ["12","34","5"]).</p> <p>If string or a regex pat is given, splits on all occurrences of pat , omitting it from the results.</p> <p>If null is given, splits s into its component <i>characters</i>.</p> <p>Returns an array of results, or an array with the entire string as the only element, if there are no matches.</p> <p>See also: b:/ a:join</p>
/scripts	s -- a

word	sed/description
	<p>Breaks the string into an array of runs of characters with the same script and direction characteristics (LTR vs RTL).</p> <p>See also: s:script?</p>
/ws	s -- a (as of 24.05)
	Same as s:trim /\s+/ s:/ , but more concise.
<+	s s2 -- s' s n -- s' (as of 16.08)
	Prepends the string or Unicode character to the string.
<>	s ix1 ix2 -- s' (as of 22.07)
	Swaps the string characters at those indices. Ex: "abc" 1 2 s:<> returns "acb" . Negative indices count from the end of the string.
=	s1 s2 -- T
	<p>Compare the strings s1 and s2 , returning true if they are the same UTF-8 bytes, otherwise false . This implies that if a character is improperly encoded it will not compare as equal to its properly encoded counterpart.</p> <p>See also: s:cmp s:cmpi</p>
=ic	s1 s2 -- T (as of 16.12)
	<p>Same as s:= , but ignores case differences.</p> <p>See also: s:= s:cmp</p>
>base64	s -- s'
	<p>Encode a string in base64 encoding.</p> <p>See also: s:base64></p>
>ucs2	s -- b (as of 17.04)
	Converts the string to its UCS2 representation.
@	s ix -- s n

word	sed/description
	<p>Returns the Unicode character n at index ix in the string. If ix is greater than the number of characters in s, null is returned instead of a character. If ix is negative, the character that far from the end of the s will be returned (-1 is the last character).</p> <p>Ex: "abcdef" -2 s:@ returns the character "e" (e.g. 101).</p> <p>See also: s:!</p>
append	s1 s2 -- s1 s n -- s (as of 18.04)
	Same as s:+ , but, modifies the original string rather than creating a new one.
base64>	s -- s'
	<p>Decode a string from base64 encoding.</p> <p>See also: s:>base64</p>
clear	s -- s
	<p>Overwrites the contents of s with 0, and resets its length to 0.</p> <p><i>Note:</i> this modifies the original string itself!</p> <p>See also: b:clear</p>
cmp	s1 s2 -- n
	<p>Compare two strings, returning n which is -1 if s1 < s2, 0 if they are the same, and 1 if s1 > s2.</p> <p>See also: s:cmpi s:=</p>
cmpi	s1 s2 -- n
	<p>Same as s:cmp, but compares in a case-insensitive manner.</p> <p>See also: s:cmp s:=</p>
compress	s -- s'
	<p>Compress a string using "zlib".</p> <p>See also: s:expand</p>

word	sed/description
count-match	s s1 -- s n (as of 22.06)
	Returns the number of non-overlapping times the string s1 occurs within s .
days!	short long --
	<p>Takes two arrays: short , containing the short names of the week-days, and long containing the long names of the week-days, and inserts them into the appropriate maps for localization according to the current value of G:curlang . Should be used in the appropriate "lang/xx" asset.</p> <p>See also: G:long-days G:short-days</p>
dist	s1 s2 T -- n (as of 20.07)
	Return the "Levenshtein" distance between the strings (number of insertions, deletions, and transpositions to convert one to the other). If T is true , performs folding first to eliminate diacritics and case differences.
each	s w --
	<p>Invokes w for each character of the string. w 's SED is: ix n -- The number ix is the index of the character in the original string. n is the character's Unicode encoding.</p> <p>Modifying the string while it is being iterated may throw an exception. Use s:map if you want to do that.</p> <p>See also: s:each!</p>
each!	s w -- (as of 19.04)
	<p>Same as s:each , but only passes the Unicode character, omitting the index, to the word. The SED of w is: n -- .</p> <p><i>Note:</i> Modifying the string while it is being iterated may throw an exception. Use s:map if you want to modify the string.</p> <p>See also: s:each</p>
eachline	sb w --

word	sed/description
	<p>Invokes the word w for each line of the input string or buffer, passing that line in TOS. Lines are simply CR or CRLF or LF delimited runs of characters. The SED of w is: s -- . Responds to break .</p> <p><i>NOTE:</i> the same actual string is passed to the callback, so if you want to store the data you need to clone it (or use const).</p> <p>See also: s:each</p>
escape	s1 s2 s3 -- s' (as of 19.08)
	<p>Given a source string, a string of characters s2 to escape, and a string indicating a character or string of characters s3 to use as the escape character, returns a string which consists of the original with all instances of any of the characters in s2 escaped.</p> <p>Ex: "abcd" "bc" "\\\" s:escape will result in "a\b\cd" .</p>
expand	s n -- s'
	<p>Expand a "zlib"-compressed string, where the number n is the original size of the uncompressed string. If the original size is not known, pass 0 for n .</p> <p>See also: s:compress</p>
fill	s rep -- s' s n -- s'
	<p>Fill the contents of the s with string rep or with the Unicode point n . rep is repeated until s' is filled without extending the length of s.</p> <p>See also: b:fill</p>
fold	s T -- s' (as of 20.07)
	<p>Lowercases the string, removing diacritics if T is true . Unicode-aware.</p>
gen-uid	n -- s (as of 24.02)
	<p><i>needs rand/ident</i></p> <p>Create a string of random characters, suitable for an identifier. The first character is chosen from a limited palette so the result should not be confused with a number. The first character is randomly chosen from among 19 characters; the rest are randomly chosen from 36. It uses the cr:rand word to shuffle the characters.</p>
globmatch	s1 s2 -- T (as of 16.02)

word	sed/description
	<p>Using the f:glob matching algorithm, compares the two strings. Returns true if they match.</p> <p>See also: f:glob f:rglob</p>
hexupr	T --
	<p>Controls how hexadecimal numbers are converted to strings. If T is true , then hex numbers will be uppercase. The default is false (e.g. lowercase).</p>
insert	s1 s2 ix -- s3
	<p>Insert the string s2 into the string s1 at offset ix , which must be a positive number. If ix is greater than the current number of characters in s1 , then s2 will be appended to s1 . Similarly, if it is 0 or less, s2 will be prepended to s1 .</p>
intl	s -- s'
	<p><i>needs string/i18n</i></p> <p>Returns the translation of the string, based on the language set by s:lang . If no translation was given, the original string is returned, otherwise the translated string is returned.</p> <p>See also: s:lang s:intl! G:curlang</p>
intl!	m --
	<p><i>needs string/i18n</i></p> <p>Sets the <i>map</i> to use for translating <i>strings</i> for the language G:curlang.</p> <p>See also: s:lang s:intl G:curlang</p>
lang	s --
	<p><i>needs string/i18n</i></p> <p>Set the current language for s:intl to the string s, which is the name of a sub-asset. For example, if the language is given as "de", there must be an asset called lang/de . That asset consists solely of a JSON <i>map</i> whose keys are the untranslated <i>strings</i>, and whose values are the translation of the key into the requested language.</p> <p>If the requested language asset does not exist, the current language is not changed.</p> <p>See also: s:intl s:intl! G:curlang</p>
lc	s -- s'

word	sed/description
	<p>Convert the string s to lowercase. Understands Unicode case conventions. If it is a number, returns the corresponding lowercase Unicode point as a number.</p> <p>See also: s:uc</p>
lc?	s -- s T n -- n T (as of 20.07)
	<p>Returns true if given a "lowercase" Unicode character. If given a string, only the first character is tested.</p> <p>See also: s:uc s:lc s:uc?</p>
len	s -- s n
	<p>Returns the length of the string in <i>characters</i> it currently contains (takes UTF-8 into account, so may be less than what s:size reports).</p> <p>See also: s:size</p>
len'	s -- n (as of 24.03)
	<p>Same as s:len but does not leave the string on the stack.</p>
len2	s1 s2 -- s1 s2 n1 n2 (as of 24.03)
	<p>Given two strings, returns their respective lengths (of characters, not bytes). Much faster and more efficient than the hard way.</p>
lsub	s n -- s'
	<p>Returns the leftmost n characters of the string. If n is negative, ends that many characters from the end of the string. Ex: "abcde" 2 s:lsub returns "ab", and "abcde" -2 s:lsub returns "abc".</p> <p>See also: s:rsub</p>
ltrim	s -- s'
	<p>Remove whitespace from the left end of the string.</p> <p>See also: s:rtrim s:trim</p>
map	s w -- s'

word	sed/description
	<p>Create a string whose elements are formed by executing the word w for each character of the original string. The SED of w is n -- x , meaning it gets a Unicode character and may modify it. It must return one of a:</p> <ul style="list-style-type: none"> • number: a Unicode character to replace n • string: likewise replacing n • boolean: if true , keep n ; otherwise omit it
months!	short long --
	<p>Takes two arrays: short , containing the short names of the months, and long containing the long names of the months, and inserts them into the appropriate maps for localization according to the current value of G:curlang . Should be used in the appropriate "lang/xx" asset.</p> <p>See also: G:long-months G:short-months</p>
n>	n -- s a -- s (as of 22.02)
	Converts a single number, or an array of numbers, representing Unicode code points, to a string.
new	n -- s s -- s' b -- s (as of 18.05)
	<p>Creates a new string s . If given a:</p> <ul style="list-style-type: none"> • number: returns an empty string capable of holding n bytes • string: returns its clone • buffer: returns a string containing its bytes (like >s)
norm	s n -- s' (as of 20.07)
	<p>Normalizes the string in the manner specified by n :</p> <ul style="list-style-type: none"> • 18: NFD (decompose) • 10: NFC (compose) • 22: NFKD (decompose compatible) • 14: NFKC (compose compatible) or use the constants defined in the strings/normal library
reduce	s w x -- x' (as of 19.07)
	<p>Analogous to a:reduce . Iterates the string, passing w the accumulating value x and the character at each index. SED of w is x n -- x' , where n is the Unicode character.</p>
reinsert	s n s2 -- s'

word	sed/description
	<p><i>needs string/repinsert</i></p> <p>Inserts the string s2 inside s every n characters. Example: "abcdef" 2 "xx" s:repinsert results in "abxxcdxxef"</p>
replace	s pat rep -- s'
	<p>Create a new string by replacing the pattern pat , just once, with the string rep . pat can be a string or a regex, and is replaced only once. To replace all, use s:replace! .</p> <p>See also: s:replace!</p>
replace!	s pat rep -- s'
	<p>Same as s:replace , but replaces <i>all</i> occurrences.</p> <p>See also: s:replace</p>
rev	s -- s'
	<p>Returns the string with its characters in reverse order. It is semantically the same as null s:/ a:rev "" a:join but faster, and uses only enough memory to store the new string.</p>
rsearch	haystack {ofs} needle -- haystack n
	<p>Same as s:search , but returns a number which is the offset of the <i>last</i> occurrence of needle in haystack . Returns null if there is no match.</p> <p>See also: s:search</p>
rsub	s n -- s'
	<p>Returns the rightmost n characters of the string. If n is negative, starts from the left side of the string.</p> <p>See also: s:sub</p>
rtl	s -- s' (as of 24.03)
	<p>Does an "RTL fixup" on the string, swapping position of any RTL (Hebrew/Arabic, etc) so they draw correctly.</p>
rtrim	s -- s'

word	sed/description
	<p>Remove whitespace from the right end of the string.</p> <p>See also: s:ltrim s:trim</p>
scan-match	s s1 s2 -- s ix (as of 22.06)
	<p>Scans the string and returns the offset of the instance of s2 which matches s1. That is to say, s1 and s2 are matched in number of occurrences. This is primarily used to parse strings containing code (e.g. CSS or the like) where a character (or a string tag) is used to delimit chunks of code. null is returned if no matching values were found, otherwise the character offset of the start of s2 is returned.</p>
script?	s -- s m
	<p>Returns information about a string's scripts and directionality in a map. The "dir" key will be either "LTR" or "RTL". The "script" key in the map returned will be one of "mixed", "latin", "greek", "cyrillic", "armenian", "hebrew", "arabic", "syriac", "thaana", "nko", "indic", "cjk", "number", "whitespace" or "symbol". If "mixed", then more than one script is present in the string.</p> <p>See also: s:/scripts</p>
search	haystack needle -- haystack n haystack ofs needle -- haystack n
	<p>Search the string haystack for the first occurrence of the string or regex or number needle, returning the numeric offset in the string of the found text, or null if not found. If needle is a number, it is treated as a Unicode character. If ofs is given, it is a number which indicates at which offset in characters within haystack to start searching from (a negative offset counts from the end of the string).</p> <p>See also: s:rsearch b:search</p>
size	s -- s n
	<p>Returns the size of the string in <i>bytes</i>. Not necessarily the same as the number of characters given by s:len, since strings are UTF-8 encoded.</p> <p>See also: s:len</p>
slice	s ix n -- s'
	<p>Returns a new string representing a "slice" of the string s, beginning at the position ix for n <i>characters</i> (not bytes!). A negative value of ix means "from the end of the string". If ix is beyond the end of s, null is returned. A negative value of n means "take the rest of the string".</p> <p>See also: b:slice</p>

word	sed/description
soundex	s -- s' (as of 20.07)
	Returns a "Soundex" code for the given string. Handles diacritics by ignoring them.
strfmap	m a m2 -- m s
	<i>needs string/strfmap</i> Utility to format like s:strfmt does, but taking named keys from the a <i>array</i> and pulling the values from the <i>map</i> m2 . Still positional, unlike s:tsub .
strfmt	a fmt -- s x1 x2 ... xn fmt -- s

word	sed/description																																				
	<p>Like the "printf" function in C. Takes a string fmt containing formatting characters, and either an array containing the items to interpolate into the result string, or items x1 , x2 , etc. on the stack corresponding in number to the items to be interpolated. The formatting is done by taking a % inside fmt to mean "format an item from the stack or array, right here". A literal % character is given by %% . Valid formatting codes are:</p> <table><tr><th>char</th><th>format</th></tr><tr><td>b</td><td>buffer</td></tr><tr><td>c</td><td>Unicode code point, or the first character of a string</td></tr><tr><td>d</td><td>integer</td></tr><tr><td>f</td><td>float</td></tr><tr><td>g</td><td>float with exponent</td></tr><tr><td>p</td><td>pointer (number, word, or ptr)</td></tr><tr><td>s</td><td>string</td></tr><tr><td>x</td><td>hex integer</td></tr></table> <p>Width, filler, and alignment options <i>precede</i> the format character:</p> <table><tr><th>char</th><th>description</th></tr><tr><td>*</td><td>next char is the 'filler'</td></tr><tr><td>,</td><td>next char is the 'thousands separator'</td></tr><tr><td>0</td><td>zero-pad the number</td></tr><tr><td><</td><td>left-align</td></tr><tr><td>></td><td>right-align</td></tr><tr><td>@</td><td>preceding digits are the base to use for this number</td></tr><tr><td>\</td><td></td></tr><tr><td>n.m</td><td>field width of 'n' characters, 'm' after decimal (optional)</td></tr></table> <p>Ex: 123 "%*-5d" s:strfmt . prints --123 .</p> <p>See also: G:.# G:n# G:c# s:strfmt G:,# G:e#</p>	char	format	b	buffer	c	Unicode code point, or the first character of a string	d	integer	f	float	g	float with exponent	p	pointer (number, word, or ptr)	s	string	x	hex integer	char	description	*	next char is the 'filler'	,	next char is the 'thousands separator'	0	zero-pad the number	<	left-align	>	right-align	@	preceding digits are the base to use for this number	\		n.m	field width of 'n' characters, 'm' after decimal (optional)
char	format																																				
b	buffer																																				
c	Unicode code point, or the first character of a string																																				
d	integer																																				
f	float																																				
g	float with exponent																																				
p	pointer (number, word, or ptr)																																				
s	string																																				
x	hex integer																																				
char	description																																				
*	next char is the 'filler'																																				
,	next char is the 'thousands separator'																																				
0	zero-pad the number																																				
<	left-align																																				
>	right-align																																				
@	preceding digits are the base to use for this number																																				
\																																					
n.m	field width of 'n' characters, 'm' after decimal (optional)																																				
term	s n -- s' (as of 22.07)																																				
	<p>Ensure the string s is terminated by the Unicode point n . Returns a new string, only adds n if the string is not already terminated with it. Ex: "abc" 'x s:term returns "abcx" .</p>																																				
text-wrap	s font n -- a (as of 20.01)																																				

word	sed/description
	<p>Takes a string and breaks it into sections which will display in an area at most n pixels wide, using the font. If font is null , then n is a maximum width of characters (rather than pixels). The string will be separated on word boundaries, if possible, and leading whitespace removed. The resultant array will have at least one string in it. If s cannot be 'wrapped' in the given width, it will be the sole entry in a .</p>
tr	s1 s2 s3 -- s (as of 23.08)
	<p>Translate the characters in s1 according to the translation provided between s2 and s3 , where each character of s2 , if it appears in s1 , is converted to the corresponding character in s3 .</p> <p>The length (in characters) of s2 and s3 must be the same and non-zero, otherwise the original s1 is returned.</p>
translate	s1 s2 s3 -- s (as of 22.05)
	<p><i>needs string/translate</i></p> <p>Translate the characters in s1 according to the translation provided between s2 and s3 , where each character of s2 , if it appears in s1 , is converted to the corresponding character in s3 . Example: "abbc" "abc" "123" s:translate returns "1223" NOTE: this is deprecated: use s:tr instead.</p>
trim	s -- s'
	<p>Remove whitespace from both ends of the string.</p> <p>See also: s:ltrim s:rtrim</p>
tsub	s x -- s'
	<p><i>needs string/tsub</i></p> <p>Takes a <i>string</i> with replacement parameters such as %0%, %1% or %name%, %address%, and values x in an <i>array</i> (if using %0% etc) or <i>map</i> (if using %name% etc) containing items to replace the parameters in s. The sequence "%%%" in s produces a literal "%" in the new <i>string</i>. If the value being substituted is a <i>word</i>, it is <i>invoked</i> and the result is used as a substitute. Otherwise, the item is converted to a <i>string</i> and inserted.</p>
uc	s -- s'
	<p>Convert the string s to uppercase. Understands Unicode case conventions. If it is a number, returns the corresponding uppercase Unicode point as a number.</p> <p>See also: s:lc</p>
uc?	s -- s T n -- n T (as of 20.07)

word	sed/description
	Returns true if given an "uppercase" Unicode character. If given a string, only the first character is tested. See also: s:uc s:lc s:lc?
ucs2>	b -- s (as of 17.04) Converts from a UCS2 representation to the UTF8 string.
utf8?	s -- s T (as of 16.11) Determines if the string is a valid UTF-8 sequence or not. Returns true if it is valid UTF-8, false otherwise.
zt	s -- s Ensure the string s ends with NUL (ASCII zero) and adjust its length if needed. Not generally needed, but if you convert from an external buffer (e.g using G:unpack) or you need to pass a string to an external (e.g. system) library, you may need this, because 8th strings are not always NUL terminated.

Structures

Namespace: **struct**

Description: Interface for easy FFI structures

word	sed/description
>buf	str -- str buf throw <i>needs utils/structs</i> Converts a <i>struct</i> to a <i>buffer</i> .
arr>	str1 buf -- str2 throw <i>needs utils/structs</i> Using the <i>struct</i> str1 as a pattern, and an <i>array</i> arr of data, create a new <i>struct</i> str2 .
buf	str len -- <i>needs utils/structs</i> Create a field of type "buffer" ("len" bytes, or if "0" as many bytes as the <i>buffer</i> or <i>string</i>) with the given name.

word	sed/description
buf>	str1 buf -- str2 throw
	<i>needs utils/structs</i> Using the <i>struct</i> str1 as a pattern, and a <i>buffer</i> buf of encoded data, create a new <i>struct</i> str2 .
byte	str --
	<i>needs utils/structs</i> Create a field of type "byte" (1 bytes integer) with the given name.
double	str --
	<i>needs utils/structs</i> Create a field of type "double" (8 bytes) with the given name.
field!	str fld val -- str
	<i>needs utils/structs</i> Using the <i>struct</i> str , the name of a field fld , and the value of that field val , store the value in the <i>struct</i> .
field@	str fld -- str val
	<i>needs utils/structs</i> Using the <i>struct</i> str and the name of a field fld , get the value of that field val .
float	str --
	<i>needs utils/structs</i> Create a field of type "float" (4 bytes) with the given name.
ignore	str num --
	<i>needs utils/structs</i> Create a field to ignore num bytes (e.g. skip that many bytes).
int	str --
	<i>needs utils/structs</i> Create a field of type "int" (4 bytes) with the given name.
long	str --

word	sed/description
	<i>needs utils/structs</i> Create a field of type "long" (8 bytes) with the given name.
struct;	str --
	<i>needs utils/structs</i> End the definition of a "struct", creating a new <i>word</i> which when invoked will put the empty struct on TOS.
word	str --
	<i>needs utils/structs</i> Create a field of type "word" (2 bytes integer) with the given name.

Task

Namespace: **t**

Description: Task (Thread)

word	sed/description
!	x s -- (as of 17.06)
	Stores the value x into the task-local variable named by the key s . See also: t:@ w:! w:@
@	s -- x a -- a' (as of 17.06)
	Returns the value of the task-local variable named by the string. If passed an array, returns the named values. See also: t:! w:! w:@
by-name	s -- a (as of 19.08)
	Returns an array of the task(s) named by the string. null is returned if no matching tasks were found.
curtask	-- t (as of 16.11)
	Returns the identifier for the current task.

word	sed/description
def-queue	n --
	Sets the default size of the queue to give new tasks to n , which must be between 1 and 100000. The default is 8.
def-stack	n --
	Sets the size of the stack to give new tasks to n , which must be at least 16. The default is 8192 on mobile, and 131072 otherwise. The default data stack size is set using G:stack-size .
done?	t -- t T
	Returns true if the task is done (meaning that the word used to run it has finished). Returns false as long as that word continues running.
dtor	w -- (as of 21.09)
	Sets a word invoked when the task terminates. Not on Windows.
err!	n s -- (as of 19.01)
	<p>Sets the value to be returned from t:err? . The parameter n is a number error code, where 0 is considered 'no error' and s is a string which describes the error.</p> <p>See also: t:err?</p>
err?	-- m (as of 19.01)
	<p>Returns a map containing "errno" and "msg", which are the integer error code, and a text representation of it, of the last error which occurred in the current task.</p> <p>See also: t:err!</p>
errno?	-- m (as of 19.04)
	<p>Sets the task error according to the current value of "errno", and returns the map with the error information just as t:err? does.</p> <p>See also: t:err?</p>
extra	-- x (as of 23.05)

word	sed/description
	Gets the task-specific 'extra' value, used internally; but for net:server et. al. returns the net that task is handling.
getq	-- q (as of 17.07)
	Gets the current task's queue.
handler	w -- (as of 18.07)
	<p>Install an exception handler for a task. w has the SED x -- n , where 'x' is information about the exception (usually a string, but any item can be thrown). The number n is one of:</p> <ul style="list-style-type: none"> • 0: not handled by w , so invoke G:handler • 1: ignore and continue • 2: terminate this task (if REPL task, app is terminated) • 3: restart the task (if not REPL task) • 4: resume at another word w2 , in this case the SED is x -- w2 n • 5: show a dialog (or console message) and quit afterwards (the default for GUI apps) <p>Any value other than those will invoke the default handler. The default is 0.</p>
handler@	-- w (as of 20.05)
	Returns the current value of t:handler
kill	t n --
	<p>Kill the task, waiting n seconds before forcibly killing it. A negative number means 'wait forever' as for G:sleep .</p> <p><i>Note:</i> the timeout is not implemented for Android.</p> <p>See also: t:task t:task-n</p>
list	-- a (as of 17.08)
	Retrieves a array of tasks which are currently "alive". Might not be valid at any subsequent time.
main	-- t
	Returns the identifier for the REPL, or main, task (there is always at least one task, which the main interpreter runs on).

word	sed/description
max-exceptions	n -- (as of 20.01)
	Allows up to n exceptions to occur on the current task within 100 msec, before it is killed. A value of 0 or less means "allow any number of exceptions". The default value is 10.
name!	s -- (as of 17.08)
	Sets the name of the current task.
name@	-- s (as of 17.08)
	Gets the name of the current task.
notify	t --
	Notify the task t to stop sleeping. See also: t:wait t:q-wait G:sleep
parent	-- t (as of 19.08a)
	Returns the task which launched the current one.
pop	-- x
	Pops an item (or null if there is no item on the queue and false q:throwing had been invoked on the queue) from the current task's queue. This must be done within the word defining the task (or from one invoked by it). See also: t:task t:push
priority	t n --
	Set the priority of the task t to n . The priority is a number from 0 (lowest) to 10 (highest). 5 is the normal value.
push	t x --
	Pushes the item onto the task's queue. The size of that queue is determined by t:def-queue . The item pushed can be retrieved only by the task itself, using t:pop . See also: t:task t:pop

word	sed/description
push!	t x -- (as of 22.06)
	<p>Pushes x onto the task's queue and notifies the task. Equivalent of _dup t:push t:notify but much more efficient. Notifies the task and the queue, so waiting with either -1 sleep or -1 t:q-wait will work.</p> <p>See also: t:task t:pop t:push</p>
q-notify	t -- t (as of 17.07)
	<p>Notifies the queue of the task that data awaits. Same as q:notify for the task-queue.</p> <p>See also: G:sleep q:notify t:wait t:q-wait</p>
q-wait	n -- (as of 17.07)
	<p>Waits n milliseconds for the current task's queue to contain something. Same as q:wait for the task-queue. Requires a corresponding t:q-notify on its task to wake it up and check its queue.</p> <p>See also: G:sleep q:notify t:wait t:q-notify</p>
qlen	-- n
	Returns the number of items in the current task's queue.
result	t -- t x
	Returns the last value x the task put on TOS (when it finished). The value will be null until the task is finished (and might be null after, depending on what's on TOS). Check for the task having finished using t:done? .
set-affinity	n -- a -- (as of 19.05)
	Attempts to set the CPU affinity for the current task to the CPU number n . If passed an array of numbers, sets the affinity to the set of CPUs.
setq	q -- (as of 20.01)
	Sets the current task's queue, moving items from the current queue to the new one.
task	w -- t m -- t

word	sed/description																					
	<p>Create a new "task", which is an independent thread of execution. It receives its own data stack and invokes the word you provide as its its starting point. When that word exits, the task goes away.</p> <p>You may also pass a map instead of a word, in which case the keys are:</p> <table><tr><th>key</th><th>description</th><th>default</th></tr><tr><td>auto</td><td>automatically decref task when xt done</td><td>false</td></tr><tr><td>name</td><td>the name to assign the new task</td><td>name of xt</td></tr><tr><td>num</td><td>the number of items to transfer from the stack to the new task</td><td>0</td></tr><tr><td>qsize</td><td>the size of the task's queue</td><td>8</td></tr><tr><td>stack</td><td>the size of the recursion stack</td><td>128K</td></tr><tr><td>xt</td><td>the word to execute (required)</td><td></td></tr></table> <p>The word t:task-n can be used if you want the "num" option without specifying all the rest.</p> <p>See also: t:task-n</p>	key	description	default	auto	automatically decref task when xt done	false	name	the name to assign the new task	name of xt	num	the number of items to transfer from the stack to the new task	0	qsize	the size of the task's queue	8	stack	the size of the recursion stack	128K	xt	the word to execute (required)	
key	description	default																				
auto	automatically decref task when xt done	false																				
name	the name to assign the new task	name of xt																				
num	the number of items to transfer from the stack to the new task	0																				
qsize	the size of the task's queue	8																				
stack	the size of the recursion stack	128K																				
xt	the word to execute (required)																					
task-n	xn... x2 x1 n w -- t xn... x2 x1 m -- t																					
	<p>Same as t:task , except that it takes a number of items to transfer from the current task's stack to the new task's stack. If TOS is a map, then the key "num" has the same role as n .</p> <p>See also: t:task</p>																					
task-stop	--																					
	<p>Terminates the running task, with prejudice. Normally, a task terminates when the word passed to t:task ends.</p>																					
ticks	-- n (as of 23.04)																					
	<p>Returns how many "ticks" (e.g. d:ticks) the task has been running.</p>																					
wait	t -- a -- (as of 17.06)																					
	<p>Waits for the single task (or array of tasks) to complete. If an array, waits for <i>all</i> tasks in the array to complete.</p>																					

Tree

Namespace: **tree**

Description: Tree data structures

word	sed/description
add	tree x -- tree (as of 20.07)
	Adds the item x to the tree.
binary	w -- tree (as of 20.07)
	Creates a new binary-search-tree with a word which implements the comparison function. The tree is implemented as a balanced AA-tree.
bk	w n -- tree (as of 20.07)
	<p>Creates a new BK-tree with a max distance n (for the metric function) and a word which implements the metric. The metric's SED is tree x1 x2 -- tree n, and must obey the rules for a "metric function":</p> <ul style="list-style-type: none">• $w(x_1, x_2) \geq 0$ and integer• $w(x_1, x_2) == w(x_2, x_1)$• $w(x_1, x_2) == 0$ means $x_1 == x_2$• $w(x_1, x_2) \leq w(x_1, x_3) + w(x_3, x_2)$ <p>A typical implementation of w might be (true s:dist), which folds the strings it's passed and uses Levenshtein distance as the metric. Of course, this can only apply if the items in the tree are strings (which is typical for a BK tree).</p> <p>The n parameter limits how large the value returned from the metric is allowed to be. Values larger than it will be clamped. This prevents the tree growing beyond reason.</p>
btree	w n -- tree (as of 21.04)
	Creates a new BTREE of order n , with a word implementing the comparison function.
cmp!	tree w -- tree (as of 20.07)
	Sets w as the compare function for the tree. Typically used after tree:parse .
data	X -- x (as of 20.07)

word	sed/description										
	Returns the data held in the node, not implemented for BTREE. See also: tree:find tree:next tree:prev tree:parent										
del	tree x -- tree (as of 20.07) Removes the node (if an X) or item (otherwise) from the tree. Not currently implemented for BK or BTREE trees. See also: tree:find tree:next tree:prev tree:data tree:add										
find	tree x -- tree X (as of 20.07) Does tree:search with a tolerance of 0, and returns the first matching node (not data item) which can be used to traverse the tree from that point. Not implemented for BTREEs. See also: tree:next tree:prev tree:data tree:parent										
iter	tree w n -- tree (as of 20.07) Traverses the tree, invoking w for each node. The value of n (as per tree/enums library): <table><tr><th>value</th><th>meaning</th></tr><tr><td>1</td><td>pre-order (left,root,right)</td></tr><tr><td>2</td><td>in-order (root,left,right)</td></tr><tr><td>3</td><td>post-order (left,right,root)</td></tr><tr><td>4</td><td>rev-in-order (right,root,left)</td></tr></table> The SED of w is x -- , where the item is the node's data. break stops the iteration.	value	meaning	1	pre-order (left,root,right)	2	in-order (root,left,right)	3	post-order (left,right,root)	4	rev-in-order (right,root,left)
value	meaning										
1	pre-order (left,root,right)										
2	in-order (root,left,right)										
3	post-order (left,right,root)										
4	rev-in-order (right,root,left)										
next	X -- X' (as of 20.07) Gives the next node in the tree, or null . Not implemented for BK or BTREEs (returns null). See also: tree:find tree:prev tree:data tree:parent										
nodes	tree X -- tree a (as of 20.07) Returns an array of the nodes which are children of the node.										
parent	X -- X' (as of 20.07)										

word	sed/description
	<p>Returns the node's parent, or null .</p> <p>See also: tree:find tree:next tree:prev tree:data</p>
parse	s -- tree (as of 20.07)
	Reconstitutes a tree from its string representation (from >s).
prev	X -- X' (as of 20.07)
	<p>Gives the previous node in the tree, or null . Not implemented for BK or BTREEs (returns null).</p> <p>See also: tree:find tree:next tree:data tree:parent</p>
root	tree -- tree X (as of 20.07)
	<p>Returns the root node of the tree, or null .</p> <p>See also: tree:find tree:next tree:prev tree:data</p>
search	tree x n -- tree a (as of 20.07)
	Searches the tree for items matching x within the (non-negative) tolerance n . Returns a possibly empty array of matches. A tolerance of zero means 'exact match'. For binary trees, tolerance means the number of surrounding items to return (tolerance not yet implemented for BTREEs).
trie	w T -- tree (as of 21.05)
	<p>Creates a new TRIE. If T is true , the trie is case-insensitive; otherwise, it's case-sensitive. If the items to be inserted in the TRIE are strings, then w must be null ; otherwise, a word w must be provided to create a string representing the items inserted so that they can be ordered "lexicographically". Any UTF-8 string is acceptable as data.</p> <p>If provided, the SED of w is x -- s .</p>

websocket

Namespace: **ws**

Description: Web Socket utilities

word	sed/description
close	T -- b
	<i>needs net/websocket</i> Create a 'close' frame. T is true for the client, false for a server.
decode	b -- m
	<i>needs net/websocket</i> Given a websocket packet in the buffer, return a map containing keys: "data" - the decoded data "op" - the websocket opcode "fin" - if 0, this is not the last packet of the message
encode	x op fin -- b
	<i>needs net/websocket</i> Encode the buffer or string x with the opcode 'op' and final status 'fin'.
encode-nomask	x op fin -- b
	<i>needs net/websocket</i> Encode the buffer or string x with the opcode 'op' and final status 'fin', but without masking; as appropriate for a server-to-client response.
gen-accept-header	s -- s
	<i>needs net/websocket</i> Given the raw security key from the client, return an acceptance header
gen-accept-key	s -- s
	<i>needs net/websocket</i> Convert the hashed 'Sec-WebSocket-Key' we received into the expected response
opcodes	-- n
	<i>needs net/websocket</i> Constants representing the opcodes for websocket packets: CONTINUATION, TEXT, BINARY, CLOSE, PING, PONG
open	s -- net true false

word	sed/description
	<p><i>needs net/websocket-client</i></p> <p>Try to open a websocket connection to the URL given. Returns 'true' followed by the net to use, or false if the connection was unable to be opened. A URL with 'ws://' or 'wss://' is accepted, as is 'http://' or 'https://'.</p>

Word

Namespace: **w**

Description: Words are the smallest unit of execution

word	sed/description
!	x s -- (as of 17.06)
	<p>Stores the value x into the "local variable" named by the string. Before using this you must have invoked G:locals: !</p> <p>See also: w:@ t:@ t:! G:locals:</p>
(is)	w s -- (as of 20.04)
	<p>IMMEDIATE</p> <p>Same as w:is , but takes the name of the deferred word on TOS instead of parsing it.</p> <p>See also: G:defer: G:(defer) w:is</p>
@	s -- x a -- x (as of 17.06)
	<p>Gets the "local variable" named by the string. The value x may be of any type, and if s is unknown then null is returned. If passed an array, all values corresponding to the passed in keys are returned.</p> <p>Before using this word you must have invoked G:locals: !</p> <p>See also: w:! t:@ t:! G:locals:</p>
alias:	w -- (as of 16.05)
	<p>Creates a new word named <name> and make it an alias for the word w . After this, invoking name will have the same effect as invoking w . May be used to give a name to an anonymous word, among other things.</p>
cb	w s -- X (as of 17.03)

word	sed/description
	<p>Creates a new "callback" given a parameter specification string and a word which will be called-back by the FFI call. The parameter specification is in the same format as for G:func: .</p> <p>You <i>must</i> use this if you need to pass a "callback function" to an FFI-invoked external function. If you do not, your callback will probably crash.</p>
deprecate	w -- (as of 17.03)
	<p>Marks the word as DEPRECATED. A word which is "deprecated" will print a warning message if it is invoked in interpret mode, or when it is compiled the first time into a new word. At runtime it will be silent; the warning is intended to alert the programmer that the word should be retired from use. The deprecation warning will appear only the first time the word is invoked or compiled.</p> <p><i>Note:</i> A deprecated word may be removed from 8th at any time after it was marked as such, so it is best to heed the warning message.</p>
dlcall	w -- x w n -- x (as of 21.03)
	<p>Calls the dynamic-library function encapsulated in the word w which was returned by w:dlsym , using the FFI template string assigned then. Returns whatever the function would, if it does. If a number is on TOS, then that many parameters will be left on the stack after the FFI call. Most useful if the first parameter is something you want to retain after the call, but don't want to do stack twiddling.</p>
dlopen	s -- X b -- X (as of 21.03)
	<p>Loads the dynamic library indicated by x from a path to the file to open.</p> <p>Pro+: if given a buffer, it's a dynamic library in memory.</p> <p>The returned item can be used with w:dlsym to return a word or other data item from the library. On failure, returns null .</p>
dlsym	X s nm s2 -- X w (as of 21.03)
	<p>Given an X returned by os:dlopen and a string representing an item to fetch from it, returns a word encapsulating the library entry named by nm . The string s2 is the FFI template to be used when invoking the word.</p>
exec	w --
	<p>Invoke (call, run, execute) the word. Does nothing if w is null .</p> <p>See also: w:exec?</p>
exec?	s -- T

word	sed/description
	<p>Try to invoke the word named by the string s , and return true if successful or false otherwise. If s is null returns false .</p> <p>See also: w:exec</p>
ffifail	s -- (as of 18.06)
	<p>DEFERRED</p> <p>Invoked if the FFI is unable to dynamically load the external function or library. The default behavior is to print a message and exit the app. s is a string which is the name of the "lib" or "func" which could not be loaded.</p>
find	s -- w
	<p>Look up the word named s and return the found word w , or null if it cannot be found.</p> <p>See also: w:forget G:'</p>
forget	s -- n --
	<p>Look up the word named by the string s , or the namespace designated by the number n , making it invisible to the w:find word. The word's code remains, but cannot be found or invoked by name.</p> <p>If a namespace is "forgotten", it is also <i>wiped</i> so it is actually impossible to find words which were in the namespace. However, any references to such words will still be alive.</p> <p>See also: w:find G:'</p>
is	w --
	<p>IMMEDIATE</p> <p>Assign the word w as the action for the deferred word whose name is <name> , which was created using defer: . This assignment can be reversed using w:undo .</p> <p>See also: G:defer:</p>
name	w -- w s (as of 19.09)
	Returns the word's name.
undo	w --

word	sed/description
	<p>Undoes the last assignment from w:is to the deferred word.</p> <p>See also: w:is G:defer:</p>
xt	w -- n (as of 24.06)
	<p>IMMEDIATE</p> <p>Returns the "xt", or execution address/token of the given word.</p>
xt>	n -- w (as of 24.06)
	Returns the word corresponding to an address, if possible; otherwise returns null .

XML

Namespace: **xml**

Description: XML parsing

word	sed/description
>s	x -- S
	<p><i>needs xml/parse</i></p> <p>Convert the <i>xml</i> x to a <i>string</i> representation s. This will not necessarily be a complete round-trip of a file parsed with xml:parse.</p>
>txt	x -- X S
	<p><i>needs xml/parse</i></p> <p>Given an <i>xml</i> x, return its text content, recursively, in a single string s.</p>
md-init	m -- X (as of 19.09)

word	sed/description
	<p>Prepares to parse "Markdown text" using a push parser. The map requires all the following keys: "enter_span", "leave_span", "enter_block", "leave_block", "text". They must be words with a SED of m -- T, where T is true to stop processing, or false to continue.</p> <p>The map the callback receives will contain the keys "text" and "user" if it's a "text" callback, or "tag" and "user" for all others.</p> <p>See the libraries md/2html and md/2console for details on the callbacks; samples xml/md2html.8th and xml/md2console.8th for how to use the parser library. The manual has full documentation.</p>
md-parse	X x sb -- X T (as of 19.09)
	<p>Takes a parser created with xml:md-init, a user-provided value 'x', and a string or buffer containing Markdown text, and runs the parser invoking the callbacks defined in it. Returns true if the parse succeeded, or false and sets t:err?. The user-provided value is passed to the parser callbacks unmodified. The map received contains the key "tag", which describes the kind of item received. Depending on the tag type, other information is also conveyed. See the manual for more detailed information.</p>
parse	inp -- xml xml null
	<p><i>needs xml/parse</i></p> <p>Take an item (<i>string or buffer or map</i>) inp possibly containing XML and parse it into a <i>map</i> xml. Return null on TOS if it failed to parse (and "error" in the <i>map</i> returned under will explain why it failed). If inp is a <i>map</i>, then its contents will override the default parse and it must provide either a "inp" key with text, or a "read" key with appropriate behavior. Currently requires that the XML be encoded in UTF-8 (or compatible).</p>
parse-html	o -- x
	<p><i>needs xml/parse</i></p> <p>Take an item (<i>string or buffer</i>) o containing HTML and parse it into a <i>map</i>, similar to xml:parse</p>
parse-stream	m -- (as of 18.04)

word	sed/description																												
	Parses a stream of XML according to the options in the given map, with word values whose keys are:																												
	<table><tr><th>key</th><th>description</th><th>SED</th></tr><tr><td>/attr</td><td>invoked when attribute ends</td><td>s -- T</td></tr><tr><td>/elem</td><td>invoked when element ends</td><td>s -- T</td></tr><tr><td>attr</td><td>invoked for each new XML element's attribute</td><td>-- T</td></tr><tr><td>cont</td><td>invoked when there is element content</td><td>s -- T</td></tr><tr><td>done</td><td>invoked on end of XML parse</td><td>--</td></tr><tr><td>elem</td><td>invoked for each new XML element</td><td>s -- T</td></tr><tr><td>read</td><td>returns a string or buffer to be parsed (required)</td><td>-- s</td></tr><tr><td>val</td><td>invoked for each new XML attribute's value</td><td>s -- T</td></tr></table>	key	description	SED	/attr	invoked when attribute ends	s -- T	/elem	invoked when element ends	s -- T	attr	invoked for each new XML element's attribute	-- T	cont	invoked when there is element content	s -- T	done	invoked on end of XML parse	--	elem	invoked for each new XML element	s -- T	read	returns a string or buffer to be parsed (required)	-- s	val	invoked for each new XML attribute's value	s -- T	
key	description	SED																											
/attr	invoked when attribute ends	s -- T																											
/elem	invoked when element ends	s -- T																											
attr	invoked for each new XML element's attribute	-- T																											
cont	invoked when there is element content	s -- T																											
done	invoked on end of XML parse	--																											
elem	invoked for each new XML element	s -- T																											
read	returns a string or buffer to be parsed (required)	-- s																											
val	invoked for each new XML attribute's value	s -- T																											

ZMQ

Namespace: **zmq**

Description: ZeroMQ interface

word	sed/description
getmsg[]	sock -- [buf] null
	<i>needs net/zmq</i> Get multi-part messages as an array of buffers.
sendmsg[]	sock arr --
	<i>needs net/zmq</i> Send (possibly) multi-part messages, given as an array of items to send.