












Emacs support for Forth

Description	Keystroke	Function	Note
Forth programming Language Support	Support for the Forth programming language is minimal: you can activate the forth-mode package by setting the pel-use-forth user option. <ul style="list-style-type: none">Files with the .f, .fs, .fth and .4th are recognized as Forth source files and will automatically activate forth-mode if the package has been activated via that user-option.Generic programming language features like template text insertion handle Forth comment style. See 🔗 Inserting Text . Forth support is provided by forth-mode external package  automatically downloaded and installed by PEL when the pel-use-forth user option is t.This mode supports Emacs 🔗 Speedbar : it shows Forth variables and words.		
Last updated on:	2025-10-17		
Open this PDF file. See also: 🔗 Help/Info	<f11> SPC f <f1>	(pel-help-pdf &optional OPEN-WEB-PAGE)	Open the 🔗 - Forth local PDF. If the prefix argument (like C-u or M--) is used, then it opens the remote GitHub hosted raw PDF instead. If the pel-flip-help-pdf-arg user-option is set it's the other way around.
	<f12> <f1>		
🔗 Customize PEL Forth support	<f11> SPC f <f2>	(pel-customize-pel &optional OTHER-WINDOW)	Customize PEL Forth support. <ul style="list-style-type: none">If OTHER-WINDOW is non-nil (use C-u), display in another window.
	<f12> <f2>		
🔗 Customize Emacs Forth support	<f11> SPC f <f3>	(pel-customize-library &optional OTHER-WINDOW)	Customize Emacs Forth support: forth, forth-smie, forth-spec. <ul style="list-style-type: none">If OTHER-WINDOW is non-nil (use C-u), display in another window.
	<f12> <f3>		
Show PEL setup for Forth	<f11> SPC e ? ?	(pel-forth-setup-info &optional APPEND)	Display Forth setup information inside a "pel-forth-info" buffer with buttons providing quick access to the customization buffer of each variable shown. The information shown includes the value and interpretation of: <ul style="list-style-type: none">the user options controlling indentation and hard tab width rendering. To append information in the buffer instead of clearing the previous content type any prefix argument (such as C-u) before the command keystroke.
	<f12> ? ?		
Forth Language Reference Access	The following commands connect to a web site to retrieve specification information about Forth words.		
Forth '94 word specification	C-c C-d 1	(forth-spec-lookup-1994 NAME)	View the documentation on NAME from the ANS'94 Forth Standard <ul style="list-style-type: none">Prompts for Forth word. Supports completion.Connects to http://lars.nocrew.org/dpans/dpans.htm, a draft of ANSI Forth '94.
Forth 2012 word specification  	C-c C-d 2	(forth-spec-lookup-2012 NAME)	View the documentation on NAME from the Forth 2012 Standard. <ul style="list-style-type: none">Attempts to access http://www.forth200x.org/documents/html/alpha.html  which, at the time of this writing (early 2021) does not exists. There are other files on this web site that contain information about Forth 2012 such as http://www.forth200x.org/documents/forth-2012.pdf. 
Navigation inside Forth Code See also: 🔗 Navigation	Basic navigation commands are available: both standard Emacs navigation and extra commands provided by PEL, as listed below. <ul style="list-style-type: none">This mode supports Emacs 🔗 Speedbar : it shows Forth variables and words.		
Forward to start of next word definition	<f6> <down>	(pel-beginning-of-next-defun &optional SILENT DONT-PUSH_MARK)	Move forward to the beginning of the next word definition. <ul style="list-style-type: none">Beeps if does not find beginning of next function unless SILENT is non-nil.If the beginning of next function is found, push the start location to the mark ring unless DONT-PUSH_MARK is non-nil.<ul style="list-style-type: none">Move back to previous position with M-`.  this should but does not work in Forth. ▀ Shift marking is available : hold Shift after typing <f6> .
Backward to end of previous word definition	<f6> <left>	(pel-end-of-previous-defun &optional SILENT DONT-PUSH_MARK)	Move backwards to the end of the previous word definition. <ul style="list-style-type: none">Beeps if does not find end of previous function unless SILENT is non-nil.If the end of previous function is found, push the start location to the mark ring unless DONT-PUSH_MARK is non-nil.<ul style="list-style-type: none">Move back to previous position with M-`.  this should but does not work in Forth. ▀ Shift marking is available.
Backward to beginning of function definition	<ul style="list-style-type: none">C-M-aC-M-<home><f6> <up>C-[C-aEsc C-a	(beginning-of-defun &optional ARG)	Move backward to the beginning of a word. <ul style="list-style-type: none">With ARG, do it that many times. Negative ARG means move forward to the ARGth following beginning of word. ▀ Shift marking is available in graphics mode, not in terminal mode (for C-M-a and C-M-<home>). It's always available for <f6> <up> : hold Shift after typing <f6> .
Forward to end of function and class definition	<ul style="list-style-type: none">C-M-eC-M-<end><f6> <right>C-[C-eEsc C-e	(end-of-defun &optional ARG)	Move forward to next end of word. With argument, do it that many times. Negative argument -N means move back to Nth preceding end of word ▀ Shift marking is available in graphics mode, not in terminal mode (both keys).
Marking Forth Words			
Mark current Forth word See also: 🔗 Marking	C-M-h	(mark-defun &optional ALLOW-EXTEND)	Put mark at end of this word definition, point at beginning. <ul style="list-style-type: none">The Foth word marked is the one that contains point or follows point.With positive ARG, mark this and that many next words; with negative ARG, change the direction of marking.If the mark is active, it marks the next or previous word(s) after the one(s) already marked.
Forth Evaluation	When a Forth interpreter is available, the following commands use it to provide interactive evaluation within Emacs. <ul style="list-style-type: none">The first time a command requiring a Forth executable is require it prompts for one then uses it.		
Open a Forth shell	<f11> z f	(run-forth)	Start an interactive forth session. <ul style="list-style-type: none">Prompt for a Forth executable.<ul style="list-style-type: none"><code>gforth</code> is a good free implementation.<ul style="list-style-type: none">On macOS, you can install it with brew install gforth in a terminal shell. Notice that it is integrated with the Home-brew Emacs installation and it will upgrade your Homebre-based Emacs unless its pinned (in which case Homebrew won't install gforth).  Requires the forth-mode external package  PEL installs and activates when the pel-use-forth user option is t . It also requires a Forth interpreter (which must be installed separately)
Evaluate Forth Expression	C-c C-e	(forth-eval-last-expression)	Evaluate Forth expression at point.
Restart Forth	C-c C-f	(forth-restart)	Restart the Forth process
Kill Forth process	C-c C-k	(forth-kill &optional BUFFER)	Kill Forth process associated with current buffer.
Load Forth File	C-c C-l	(forth-load-file FILE)	Load specified file in the Forth interpreter.
Evaluate region of Forth code	C-c C-r	(forth-eval-region START END)	Evaluate marked region of Forth code.
SEE code of current word	C-c C-s	(forth-see WORD)	Execute the Forth SEE command on the current word, accessing the code of that word. <ul style="list-style-type: none">Uses the word at point , showing the result inside the "see" buffer.
Opens the Forth process output buffer	C-c C-z	(forth-switch-to-output-buffer)	Opens the Forth process output buffer
Evaluate Forth expression.	C-c :	(forth-eval STRING)	Prompts for a Forth expression. On RET uses the Forth interpreter to evaluate it and display the result in the minibuffer.

Description	Keystroke	Function	Note
Evaluate the current Forth word.	C–M–x	(forth-eval-defun)	Evaluate the current word.
Evaluate current Forth expression	C–x M–e	(forth-eval-last-expression-display-output)	Evaluate current Forth expression.
Indenting Forth Code	👉 Type <f12>? to see how indentation is controlled in Forth buffers. You can change the value of the indentation via the buffer this opens.		
Indent expression	C–M–q	(prog-indent-sexp &optional DEFUN)	Indent the expression after point. When interactively called with prefix, indent the enclosing defun instead.
Commenting Forth Code			
Insert, realign, comment/uncomment region	M– ;	(comment-dwim ARG)	Insert or realign comment on current line (or region if a region is active). If line/region is already commented, uncomment it. <ul style="list-style-type: none"> On a single line, the comment is placed <i>after</i> the code. C–u M– ; executes comment-kill Forth comments are quite flexible. This command uses the \ word at the beginning of a line and () after now-whitespace. <ul style="list-style-type: none"> With PEL, use the <f11> ; ? command to get a full list of the variables used to control Forth comments.
See also: ⓘ Comments With PEL: Comment the current line with: M–0 M– ;		(pel-comment-dwim ARG)	Same as comment-dwim but comments the current line with a numeric ARG or 0.

Forth— References

Document	Notes
Forth Programming Language	Forth is a stack-based programming language designed Designed by Charles H. Moore
The Forth Programming Language - Wikipedia	
Forth Books @ forth.com	Links to several good Forth Books, including Starting Forth the original 1981 Forth book by Leo Brodie .
forth-standard.org	Get the latest information about Forth on this web site.
ANSI Forth '94 Draft Specification	
Forth 200x Extension Proposals	
Forth Implementations	
• GForth @ Wikipedia	GNU Forth, a free implementation
• GForth @ GNU	
• SwiftForth	A commercial implementation.
Forth Projects & Resources	<ul style="list-style-type: none"> Awesome Forth @ Github ForthHub @ Github