↖ Topic Index

Programming Languages Support 🚧

Description	Keystroke	Function	Note
Programming Language Support ○ Help & Customization • Eldoc •	This PEL PDF provides information about various common features related to programming languages: • **eldoc**, which provides help about symbol at point. • **Language Server Protocol**: supported by Emacs built-in **eglot** and the external package **lsp-mode**. This page describes the commands available globally and shows their corresponding major-mode specific key bindings. Those are available for some of the major modes that PEL supports and are also described in the PDF for the major modes. PEL controls installation and activation of the following external packages with the corresponding **pel-use-** user-option listed below.		
	📦 **c-eldoc**	☑️ **pel-use-c-eldoc**	eldoc support for C when the major mode does not use LSP support.
	📦 **lsp-mode**	☑️ **pel-use-lsp-mode**	**lsp-mode** is an alternative external **Language Server Protocol** system for Emacs.
	📦 **lsp-ui**	☑️ **pel-use-lsp-ui**	**lsp-ui** : UI integrations for lsp-mode.
	📦 **helm-lsp**	☑️**pel-use-helm-lsp**	Provides a helm interface to lsp-mode
	📦 **lsp-ivy**	☑️ **pel-use-lsp-ivy**	Provides an interactive ivy interface to the workspace symbol functionality offered by lsp-mode
	📦 **lsp-origami**	☑️ **pel-use-lsp-origami**	Provides lsp support for origami folding
	📦	☑️	
	📦 **lsp-treemacs**	☑️ **pel-use-lsp-treemacs**	lsp mode extension for ⅀𝔛 **Treemacs**
	📦	☑️	
	📦 **emacs-ccls**	☑️ **pel-use-lsp-ccls**	Emacs client for C/C++/Objective-C LSP server supported by libclang
Last updated on:	2025-12-08		
Open this PDF file. See also: ⅀ **Help/Info**	`<f11> SPC * <f1>`	(**pel-help-pdf** &optional OPEN-WEB-PAGE)	Open the ⅀ **Programming** 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.
⅀ **Customize** PEL Programming Language Support	`<f11> SPC * <f2>`	(**pel-customize-pel** &optional OTHER-WINDOW)	Customize PEL support for programming language support. • If OTHER-WINDOW is non-nil (use **`C-u`**), display in other window.
⅀ **Customize** Emacs Programming Language Support	`<f11> SPC * <f3>`	(**pel-customize-library** &optional OTHER-WINDOW)	Customize Emacs support for: eldoc, eldoc-box, eaglet, lsp
Eldoc and Global Eldoc Modes	Emacs **eldoc-mode** is a buffer-local minor mode that displays information about symbol (functions, methods, classes, variables, etc). • When the mode is enabled it displays information about symbol at point tin the echo area. • It was originally developed to support Emacs Lisp mode but evolved supporting more programming modes. Emacs also provides the **global-eldoc-mode**, enabled by default, that activates **eldoc-mode** in buffers that have various eldoc variables turned on. • List these variables with the **pel-eldoc-setup-info** command, shown below. The eldoc system is supported by several programming language major modes. More information about the support of each mode is available in the PDF page for these programming language modes.		

Show eldoc setup information for the current buffer	`<f11> SPC * d ?`
	`<f12> <f4> d ?`
		(**pel-eldoc-setup-info** &optional APPEND)	Display eldoc setup information inside a *pel-eldo-info* buffer with buttons providing quick access to the customization buffer of each variable shown. The information shown includes the value and interpretation of the variables controlling the activation and behaviour of eldoc.
			• To append information in the buffer instead of clearing the previous content type any prefix argument (such as **`C-u`**) before the command keystroke.

Documentation	Display code documentation based on **Emacs-lisp docstrings**. eldoc is active for all lisp-based modes.		

Toggle eldoc-mode Emacs Lisp Documentation Lookup	`<f11> SPC * d d`
	`<f12> <f4> d d`
		(**eldoc-mode** &optional ARG)	Toggle echo area display of Lisp objects at point (ElDoc mode).
			• With a prefix argument ARG, enable ElDoc mode if ARG is positive, and disable it otherwise.

Echo area display of the Lisp object at point.	• ElDoc mode is a buffer-local minor mode. When enabled, the echo area displays information about a function or variable in the text where point is. ◦ If point is on a documented variable, it displays the first line of that variable's doc string. ◦ Otherwise it displays the argument list of the function called in the expression point is on.		
Toggle global eldoc mode		(**global-eldoc-mode** &optional ARG)	Toggle Eldoc mode in all buffers. • With prefix ARG, enable Global Eldoc mode if ARG is positive; otherwise, disable it. • Eldoc mode is enabled in all buffers where 'turn-on-eldoc-mode' would do it. • See 'eldoc-mode' for more information on Eldoc mode.
		(**turn-on-eldoc-mode**)	Turn on 'eldoc-mode' if the buffer has ElDoc support enabled. • See variable 'eldoc-documentation-function' for more detail.
Eldoc-box	📦 Require **eldoc-box** external package. ☑️ activated by **pel-use-eldoc-box** user option. Show eldoc info in a box instead of echo area. For GUI mode only.		

Toggle eldoc-box at point	`<f11> SPC * d b`
	`<f12> <f4> d b`
		(**eldoc-box-hover-at-point-mode** &optional ARG)	Toggle eldoc-box that displays eldoc text at point.
			• You can use **`C-g`** to hide the doc.

Toggle eldoc-box on upper corner	`<f11> SPC * d B`
	`<f12> <f4> d B`
		(**eldoc-box-hover-mode** &optional ARG)	Displays hover documentations in a childframe.
			• The default position of childframe is upper corner.

Eldoc— References

Topic & Link	Notes
Eldoc extension packages	
eldoc-box	Display documentation in a child frame (works only in graphics mode, not useful in terminal (tty) mode).
eldoc-eval	
eldoc-overlay	A small minor mode that display the elder string in a text overlay directly inside the buffer. It can be enabled/disabled. When enabled, it can use the following back-ends to display the string: • quick-peek, which displays the information over 4 lines below • inline-docs, which displays the information just above in the smallest number of lines possible (and with a nicer face IMHO) • There's a small bug (I ought to submit a fix): it assumes that company-mode code is loaded and fail if it's not. • I personally prefer to use the standard eldoc ; i find it less distracting when the documentation shows up at the bottom in the mini buffer; having it pop up and off in the buffer, pushing the code below constantly annoys me. • I would probably prefer to dedicate a window for this text if I did not want the mini buffer to be affected: and then I would keep the window always opened and always showing the *last* eldoc string shown. I might perhaps implement something like that for PEL.
Eldoc Language Support	
c-eldoc	
css-eldoc	
eldoc-cmake	
eldoc-stan	
go-eldoc	
irony-eldoc	
ivy-erlang-complete	
merlin-eldoc	
php-eldoc	