↖ Topic Index

Auto-Completion Support 🚧

Description	Keystroke	Function	Note

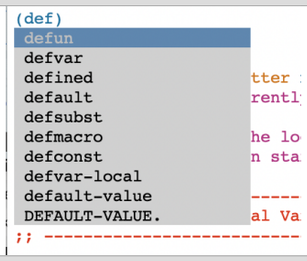
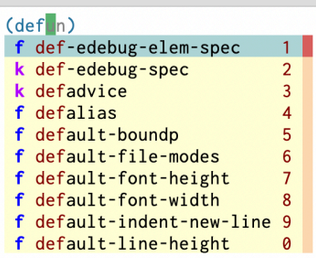
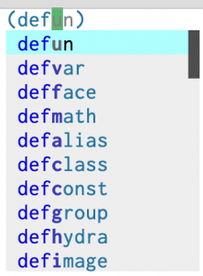
Auto Completion
○ Help @ Customization
• Display auto-completion info
• Select auto-completion mode
○ **Built-in completion**
• completion-preview-mode
○ **External completion modes**
  • auto-complete
  • company-mode
  • corfu
• marginalia
○ **Auto-completion Reference**
	Emacs provides support for text completion, called **auto-completion**.
	• Emacs provides a built-in auto-completion system, accessible via the (**completion-at-point**) command bound to **`C-M-i`** .		

	• PEL supports the following external packages that provide **alternative completion at point with pop-up menu**:		
	📦 **auto-complete**	**pel-use-auto-complete**	The oldest external completion engine.
	📦 **company-mode**	**pel-use-company**	A powerful completion engine.
	📦 **corfu**	**pel-use-corfu**	Use **CO**mpletion in **R**egion **FU**nction. A very powerful and popular auto-completion engine. Available for Emacs >= 29.1 only.
	📦 **corfu-terminal**	**pel-use-corfu-terminal**	Use corfu popup in terminal Emacs < 31. PEL activates it when pel-use-corfu is on, but Emacs < 31 and runs in terminal mode.
	PEL also support the following packages that provide supplemental features in mini buffer and pop-up completions:		
	📦 **orderless**	**pel-use-orderless**	Emacs completion style that matches multiple regexps in any order. To complete activation of orderless with **corfu** (or other system), update the following user-options: • **completion-styles** : add **orderless** to the list and ensure that **basic** follows it. • **completion-category-overrides:** set to ((file (styles partial-completion)) • **completion-pcm-leading-wildcard**: set to **t** (on Emacs >= 31)
	📦 **marginalia**	**pel-use-marginalia**	Activate **marginalia-mode** which provides marks/annotations in mini buffer or completion. • Set to **use-from-start** to activate when Emacs starts, **t** to allow later activation by the **marginalia-mode** command (bound to **`<f11> c m m`**).
	📦 **prescient**	**pel-use-prescient**	
	📦 **company-prescient**	**pel-use-company-prescient**	
	📦 **corfu-prescient**	**pel-use-corfu-prescient**	
	📦 **ivy-prescient**	**pel-use-ivy-prescient**	
	For these PEL supported modes, you can access the customization buffer quickly with the **`<f11> c <f2>`** key sequence.		
☝ Automatic activation of completion modes in specified major modes with PEL: ➽	To activate an automatic completion mode for a major mode add one of the following special PEL functions to the list of minor modes listed in the major mode specific instance of: **pel-MODE-activates-minor-modes** — Add one of the following special PEL functions that activate the corresponding engines: • pel-auto-complete-mode, pel-company-mode, pel-corfu-mode • For example, add **completion-preview-mode** and **pel-company-mode** to **pel-elisp-activates-minor-modes** to automatically activate these in all Emacs Lisp buffers.		
	• Emacs >= 30 provides the **completion-preview-mode** which provides an easy-to-use in-buffer completion help that complements the above. ☝ More on abbreviation completion is available in the 𝕀 **Abbreviations** table. Both abbreviation completion and one auto-completion mechanism can be used at the same time (using different keys if any), in some case the abbreviation choices can also be available via auto-completion. ☝ More dynamic completion modes based on **Eglot** or LSP are supported by Emacs but not yet documented in PEL.		
Last updated on:	2025-12-22		
Open this PDF file. See also: 𝕀 **Help/Info**	**`<f11> c <f1>`**	(**pel-help-pdf** &optional OPEN-WEB-PAGE)	Open the 𝕀 **Auto-Completion** 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 auto-completion support. See also: 𝕀 **Customize**	**`<f11> c <f2>`**	(**pel-customize-pel** &optional OTHER-WINDOW)	Open the PEL customize group(s) for the current context: auto-completion support. Use this to open to change PEL user option variables the activate and control the various Apple script features such as the name of the narrator voice. • When a prefix argument (like **`C-u`**) opens the buffer inside another window.
Customize Emacs built-in auto-completion support See also: 𝕀 **Customize**	**`<f11> c <f3>`**	(**pel-customize-library** &optional OTHER-WINDOW)	Customize Emacs auto-completion group. • When prefix arg. (like **`C-u`**) opens the buffer inside another window.
	The groups include: auto-complete, company, completion, completion-preview, corfu, corfu-prescient, corfu-terminal, hippie-expand, marginalia, prescient. ☝ Group belonging to files that have not yet been loaded are normally not accessible in Emacs via the customize-group command. PEL, however, attempts to locate the file that defines a non-loaded customization group and will prompt you for loading the file if it finds it.		
Display Auto-completion info: • **status and configuration** ☝ Provides quick access buttons to change customizable user-options.	**`<f11> c ?`**	(**pel-completion-info** &optional APPEND)	Print information about available auto-completion info in a *pel-autocomplete-info* **help-mode** buffer. Clear previous buffer content unless a prefix arg (like **`C-u`**) is used.
	• Prints current state and values of relevant user-options as buttons you can use to get more info and change their customized values. • Shows which one is enabled via customization and their current activation state. ☝ Underlines links are buttons that open the customization buffer where you can change the customized value.		
Select auto-completion mode	PEL allows automatic selection of auto-completion mode (see above) as well as dynamic selection inside a buffer with the following command:		
	`<f11> c =`	(**pel-select-auto-complete-tool** &GLOBALLY)	Prompt user to select auto-completetion mode to use in current buffer. • With prefix arg (such as **`C-u`**) select for all buffers. Select: Emacs built-in auto-completion, **auto-complete**, **company-mode**, **corfu**.

Description	Keystroke	Function	Note
Emacs Built-in Completion	Emacs built-in completion is provided by the **completion-at-point** command, described below. • This automatic text completion is always available but can be extended or overridden by mechanisms used by the external auto-completion packages. • The **pel-completion-info** command prints the values of the various variables and user-options controlling its behaviour.		

Symbol Completion at point
See also: ⧖ **Xref**
	• **C-M-i**
	• **<Esc> <tab>**
	• **M-<tab>**
		(**completion-at-point**)	Perform completion on the text around point.
			• The completion method is determined by '**completion-at-point-functions**'.
			• The tags-completion-at-point-function is used for Emacs Lisp code by default.
			  • It provides a list of possible values in the *Completions* buffer.
			  • The key bindings to select from *completions* buffer are shown below.

☝	**C-M-i** is also used for Flyspell, which can be used to spell check only moments and strings.		
Symbol Completion at point • **and language specific completion**	**<f6> c**	(**complete-symbol** ARG)	Perform completion of the text around point, using the method identified by the variable **completion-at-point-functions**, acting as the **completion-at-point** command above. • With prefix argument, such as **C-u**, this does completion within the collection of symbols listed in the index of the manual for the language you are using.
Complete language specific symbol at point	**<f6> C**	(**info-complete-symbol** &optional MODE)	Perform completion of symbol at point using mode-specific language items. • For example, inside a Emacs-Lisp buffer it finds the Emacs-Lisp functions, variables names.
Completion function - using tags Candidate for: completion-at-point-functions		(**tags-completion-at-point-function**)	Using tags, return a completion table for the text around point. If no tags table is loaded, do nothing and return nil. This uses the tag facility.
Using Emacs Built-in Completion	The following keys		
☝ The following keys are available whenever the *Completions* buffer shows a list of potential completions. The keys are typed at point, there's no need to switch to the window of the *Completion* buffer.	**<tab>**	Expand suggested completion in buffer. Type more characters to further refine the search.	
	M-i	Open a *Completion* buffer listing all possible completion candidates. Does not move point. • Select other candidate show in the *Completion* buffer with **M-<up>** and **M-<down>**. Then chose one with **M-RET**.	
	M-<down>	(**minibuffer-next-completion** &optional N VERTICAL)	Selects and show next completion candidate. • Move by N positions with numeric prefix N
	M-<up>	(**minibuffer-previous-completion** &optional N)	Selects and show previous completion candidate. • Move by N positions with numeric prefix N
	M-RET	(**minibuffer-choose-completion** &optional NO-EXIT NO-QUIT)	Select candidate. Run 'choose-completion' from the minibuffer in its completions window. • With prefix argument NO-EXIT, insert the completion candidate at point to the minibuffer, but don't exit the minibuffer. When the prefix argument is not provided, then whether to exit the minibuffer depends on the value of 'completion-no-auto-exit'. • If NO-QUIT is non-nil, insert the completion candidate at point to the minibuffer, but don't quit the completions window.
Using Completion Preview mode For **Emacs >= 30**	The completion preview mode is a minor mode available since Emacs 30. It completes the current symbol with a light face overlay when possible. • Can be available while another completion major-mode is used.		
Toggle completion preview mode in buffer	**<f11> c p**	(**completion-preview-mode** &optional ARG)	Toggle completion-preview-mode in current buffer. • Activate it with positive prefix argument, disable it with negative prefix argument.
Toggle completion preview mode globally	**<f11> c P**	(**global-completion-preview-mode** &optional ARG)	Toggle Completion-Preview mode in all buffers. • Activate it with positive prefix argument, disable it with negative prefix argument.
Insert completion candidate	**<tab>**	(**completion-preview-insert**)	Insert the completion candidate that the preview is showing.
Next candidate	**M-n**	(**completion-preview-next-candidate** N)	Cycle the candidate the preview is showing N candidates forward. • If N is negative, cycle -N candidates backward. • Interactively, N is the prefix argument and defaults to 1.
Previous candidate	**M-p**	(**completion-preview-prev-candidate** N)	Cycle the candidate the preview is showing N candidates backward. • If N is negative, cycle -N candidates forward. • Interactively, N is the prefix argument and defaults to 1.
List all available candidates	**M-i**	(**completion-preview-complete**)	Complete up to the longest common prefix of all completion candidates. • If you call this command twice in a row, or otherwise if there is no common prefix to insert, it displays the list of matching completion candidates unless 'completion-auto-help' is nil. • If you repeat this command again when the completions list is visible, it scrolls the completions list.
☝ The following keys are available whenever the *Completions* buffer shows a list of potential completions. The keys are typed at point, there's no need to switch to the window of the *Completion* buffer.	**M-<down>**	(**minibuffer-next-completion** &optional N VERTICAL)	Selects and show next completion candidate. • Move by N positions with numeric prefix N
	M-<up>	(**minibuffer-previous-completion** &optional N)	Selects and show previous completion candidate. • Move by N positions with numeric prefix N
	M-RET	(**minibuffer-choose-completion** &optional NO-EXIT NO-QUIT)	Select candidate. Run 'choose-completion' from the minibuffer in its completions window. • With prefix argument NO-EXIT, insert the completion candidate at point to the minibuffer, but don't exit the minibuffer. When the prefix argument is not provided, then whether to exit the minibuffer depends on the value of 'completion-no-auto-exit'. • If NO-QUIT is non-nil, insert the completion candidate at point to the minibuffer, but don't quit the completions window.

Description	Keystroke	Function	Note
External completion with pop-up drop-down list	As described above, PEL allows automatic and manual selection of the completion mode used in a buffer. • To activate one of the external auto-completion modes, add the name of the PEL specific function to the **pel-MODE-activates-minor-modes** user-option for the corresponding major mode. The PEL functions are: pel-auto-complete-mode, pel-company-mode or pel-corfu-mode. • Inside a buffer, use the **pel-select-auto-complete-tool** command (bound to **<f11> c =**) to select one of the available auto-completion mode. The key bindings to perform auto-completion are the same for all except for Emacs built-in and auto-complete and described below.		
Explicitly List Completion Candidates with the currently active auto completion system • Use auto-complete or company-mode, whichever that is currently active globally or in the buffer.	• **<f11> c c** • **M-l**	(**pel-complete**)	List completion candidates. There must be at least 1 character preceding point. • Force auto-completion of text at point, don't wait for timeout, using the currently active auto-completion system (either auto-complete-mode or company-mode). • If no auto completion system is active in the current buffer, the command issues an error. • ✂ **M-l** default binding is to downcase-word. PEL rebinds this key company-mode is activated via customization and company-mode is active. • ☝ With PEL, the **M-l** key is close to the **M-/** key, bound to the command used for abbreviation expansions. It becomes easy to use either.
Completion Menu keys • **Auto-completion Menu Operations** • **Company-Mode Menu Operations** See also: Σ **Scrolling**	When an completion pop-up menu generated **either** by auto-complete or company-mode is shown, you can use the following keys for operating on that menu: • **M-n** : next candidate (or <down> cursor) • **M-p** : previous candidate (or <up> cursor) • **M-1, M-2, M-3,** etc…: select candidate by line number • **<tab>** : complete using 1 candidate (if 1 choice), using the prefix part among many candidates, or cycle through all candidates. • **<DEL>** : Delete 1 char of the current candidate prefix • **<RET>** : Select current candidate, execute action for candidate if any (eg. when template selection used) • **C-?** : Show candidate help in separate buffer • **<f1>** : Show candidate help in separate buffer. ☝ This is **very** handy to quickly review documentation of several symbols! • **C-M-v** : Scroll help buffer forward (note: see the Σ **Scrolling** table for more info on scrolling) • **Esc <PgDown>** : Scroll help buffer forward • **C-M-S-v** : Scroll help buffer backward • **Esc <Pg-up>** : Scroll help buffer backward • **C-g** : Stop completion		
auto-complete • For Emacs >= 25.1 ⚠ A bug affects auto-complete on Emacs 30: **bug 76430**	Auto-Complete is the oldest of the 3 external auto-completion supported by PEL. 📦 Requires the **auto-complete** package that PEL supports if the **pel-use-auto-complete** customization variable is set to **t.** Completion done by **auto-complete** pops a drop-down menu when invoked by: • **<f11> c c** • **M-l** The other key bindings are still bound to the built-in completion-at-point command which brings up the "Completions" buffer: • **C-M-i** • **<Esc> <tab>** • **M-<tab>**		
company-mode • For Emacs >= 26.1	Company-Mode is the other auto completion package supported by PEL. 📦 Requires the **company-mode** external package that PEL activates when the **pel-use-company** customization variable is set to **t.** • Available for Emacs >= 26.1 only. The completion done by **company-mode** pops a drop-down menu with categorized entries (**f**: function, **k**: macro, **v**: variable) and entry index. The menu pops up on: • automatic, by timeout • manual by typing: • **C-M-i** • **<Esc> <tab>** • **M-<tab>** • **<f11> c c** • **M-l**		
corfu **CO**mpletion in **R**egion **FU**nction. • For Emacs >= 29.1	📦 Requires the **corfu** external package activated by **pel-use-corfu.** A very powerful, popular and extensible auto-completion engine. • Available for Emacs >= 29.1 only. The completion done by **corfu** pops a drop-down menu as shown at right. The menu pop up on: • automatic, by timeout • manual by typing: • **C-M-i** • **<Esc> <tab>** • **M-<tab>** • **<f11> c c** • **M-l**		
Using marginalia	📦 Requires **marginalia** activated by **pel-use-marginalia** automatically at start when set to **use-from-start** or left to start manually when set to t.		
Toggle marginalia-mode	**<f11> c m m**	(**marginalia-mode** &optional ARG)	Toggle marginalia mode on/off. This minor mode annotate completion candidates with richer information.
Show description of marginalia annotations	**<f11> c m s**	(**pel-marginalia-symbols**)	Open a help buffer describing the marginalia annotation symbols. • Use this command when marginalia-mode is active.

Auto-completion — References

Document	Note
Basic Auto Completion	**GNU Emacs Manual - Completion for Symbol Names**
Auto Completion with Auto-Complete	
Auto Complete @ MELPA	You can get auto-complete from MELPA. An interesting point of this page lists the other packages that need auto-complete. There's over 45 packages that use it for various programming languages and environments.
Auto Complete @ GitHub	Auto complete source code
Auto Complete Manual @ Github	Covers installation, check, features, concepts, configuration, advanced usage. Reading required for users.
Using Emacs: 8 - Auto-complete @ Youtube	Mike Zamansky video that covers abbreviation and auto-complete. Duration: 5 minutes.
Using Emacs: 45- Company or Autocomplete @ Youtube	Another video from Mike Zamansky that covers both auto-complete and company-mode. Duration: 13 minutes.
Auto Completion with Company-mode	
company-mode ; Modular in-buffer completion framework for Emacs	Text completion framework for emacs
Using digits to select company-mode candidates @ (or emacs irrelevant)	