














Emacs support for Ruby 🚧

Description	Keystroke	Function	Note
Ruby Editing	Emacs provides the built-in ruby-mode and ruby-ts-mode to support Ruby programming.  PEL supports it when pel-use-ruby user options is turned on. <ul style="list-style-type: none">On Emacs >= 30, PEL supports tree-sitter if pel-use-tree-sitter is set to t.<ul style="list-style-type: none">You can activate tree-sitter for Ruby by setting pel-use-ruby to 'with-tree-sitter (as long as pel-use-tree-sitter is t and Emacs >= 30). See 📖 Tree SitterThe Ruby files (.rb and several other extensions) are recognized as Ruby source files and use the ruby-mode or ruby-ts-mode according to pel-use-ruby value.Speedbar support for .ruby files listing functions and types. See 📖 Speedbar for more info about it.When it is turned on the <f11> SPC U prefix is made available. In a ruby buffer these commands are accessible via the <f12> key. It also activates the ability to activate minor modes for the ruby major mode through the PEL pel-ruby-activates-minor-modes user-option.  PEL support for Ruby is not complete. More commands should be provided and documented. Ruby support is preliminary.		
Last updated on:	2025-10-15		
Open this PDF file. See also: 📖 Help/Info	<f11> SPC U <f1>	(pel-help-pdf &optional OPEN-WEB-PAGE)	Open the 📖 I - Ruby 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 Ruby support	<f11> SPC U <f2>	(pel-customize-pel &optional OTHER-WINDOW)	Customize PEL Ruby support. <ul style="list-style-type: none">If OTHER-WINDOW is non-nil (use C-u), display in another window.
	<f12> <f2>		
📖 Customize Emacs Ruby support	<f11> SPC U <f3>	(pel-customize-library &optional OTHER-WINDOW)	Customize Emacs Ruby support: ruby. <ul style="list-style-type: none">If OTHER-WINDOW is non-nil (use C-u), display in another window.
	<f12> <f3>		
Select ruby-mode for extension-less file  The <f12> key is available only until a PEL controlled major mode is activated. Then it becomes a buffer prefix key.	<f12>	(pel-as &optional FORCE)	Inside a fundamental-mode buffer, interactively select major mode for the buffer. Re-do it with arg.  see Create extension-less executable scripts with PEL . This command is mostly used to set the major mode of a buffer in fundamental-mode', when the <f12> key binding is available for it. After being used once in a buffer the major mode is selected and the PEL key binding will not be available when PEL supports the major mode. For Ruby file, select ruby . It will insert a shebang line specified by  pel-ruby-shebang-line user option. PEL defines the (as &optional FORCE) alias unless  pel-has-alias-as user-option is set to nil. You can use M-x as to invoke it.
Show PEL setup for Ruby	<f12> ?	(pel-ruby-setup-info &optional APPEND)	Display Ruby setup information inside a *pel-ruby-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">pel-use-ruby (whether the classic or tree-sitter based major mode is used).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.
	<f11> SPC U ?		
Comments			
Toggle display of comments in buffer or active region See also: 📖 Comments	<f11> ; ;	(hide/show-comments-toggle &optional START END)	Toggle hiding/showing of comments in the active region or whole buffer. <ul style="list-style-type: none">If the region is active then toggle in the region. Otherwise, in the whole buffer.  This requires the hide-comnt.el package (see 📖 Comments).  PEL activates it when the pel-use-hide-comnt user option is t.
Generic code skeletons <ul style="list-style-type: none">tempo skeletons See also: <ul style="list-style-type: none">📖 Inserting TextT Templates	Several mechanisms have been developed to allow easy insertion of predefined text in Emacs.  PEL does not yet define skeletons for Ruby. You can use the generic one. <ul style="list-style-type: none">Emacs provides the built-in skeleton mechanism and the tempo skeletons.<ul style="list-style-type: none">PEL supports both. They are used a little bit differently. PEL provides generic tempo skeletons you can use for Ruby until PEL adds Ruby-specific skeletons.PEL provides key bindings to the tempo skeletons: the generic code templates, accessible via the <f6> prefix key, and the language-specific code templates, accessible via the <f12> key prefix.		
📖 Customize PEL Text Insertions control for Ruby code skeletons.	<f6> <f2>	(pel-customize-pel &optional OTHER-WINDOW)	Open the customization groups that control the format of the various skeletons including the generic skeleton used by the <f6> h key and the <f12><f12> h key (see below). <ul style="list-style-type: none">If OTHER-WINDOW is non-nil (use C-u), display in other window.
	<f12> <f12> <f2>	(pel-customize-generic-skels &optional OTHER-WINDOW)	
Insert generic file module header block — Language agnostic After inserting the template, navigate though areas that must be filled with: <ul style="list-style-type: none">forward: C-c .backward: C-c ,	<f6> h	(pel-generic-file-header)	Insert a file header block at the top of the file. Works only for buffer visiting a file.  The command key binding <f6> h is available only 1 second after Emacs has started.  As mentioned above PEL does not yet define Ruby-specific skeletons, this uses the generic one.
	<f12> <f12> h		
Toggle pel-tempo-mode	<f6> SPC	(pel-tempo-mode &optional ARG)	Toggle PEL tempo mode on/off.
	<f12> <f12> SPC		
PEL tempo mode activates C-c . and C-c , , as well as to C-c C-. and C-c C-, key bindings to navigate across tempo mark hot-spots. When pel-tempo-mode is active the pel-tempo-mode lighter () is shown on the status bar. The second set of keys are only available in graphics mode.  The pel-generic-file-header command inserts the text using a tempo skeleton: the PEL tempo mode is automatically activated by typing <f6> h .			
Expand any tag in template Note: PEL default skeleton does not use tags.	<f6> <f12>	(tempo-complete-tag &optional SILENT)	Look for a tag and expand it. All the tags in the tag lists in ' tempo-local-tags ' (this includes 'tempo-tags') are searched for a match for the text before the point. The way the string to match for is determined can be altered with the variable 'tempo-match-finder'. If 'tempo-match-finder' returns nil, then the results are the same as no match at all. <ul style="list-style-type: none">If a single match is found, the corresponding template is expanded in place of the matching string.If a partial completion or no match at all is found, and SILENT is non-nil, the function will give a signal.If a partial completion is found and 'tempo-show-completion-buffer' is non-nil, a buffer containing possible completions is displayed.
	<f12> <f12> <f12>		
Ruby-mode control			
Toggle string literal quoting	C-c `	(ruby-toggle-string-quotes)	Toggle string literal quoting between single and double.
Toggle block type	C-c {	(ruby-toggle-block)	Toggle block type from do-end to braces or back. <ul style="list-style-type: none">The block must begin on the current line or above it and end after the point.If the result is do-end block, it will always be multiline.
Navigation	The following navigation commands are specialized for Ruby and complement what is described in the 📖 Navigation section.		
• by block	The following commands move point through Ruby code blocks		
Move forward to end of current block	C-M-n	(ruby-end-of-block &optional ARG)	Move forward to the end of the current block. <ul style="list-style-type: none">With ARG, move out of multiple blocks.
Move backward to beginning of current block	C-M-p	(ruby-beginning-of-block &optional ARG)	Move backward to the beginning of the current block. <ul style="list-style-type: none">With ARG, move up multiple blocks.

Description	Keystroke	Function	Note
Search Support	In Python mode, the superword mode can be useful since <code>snake_case</code> is often used. Using superword-mode helps searching. PEL activates the superword mode by default in Python mode. To change this use the <code><f11> t <f2></code> to access the customize buffer.		
Toggle superword-mode See also: • Text Modes • Search/Replace	<ul style="list-style-type: none"> <code><f11> t m p</code> <code><f12> M-p</code> 	(superword-mode &optional ARG)	Toggle superword-mode: a minor mode that treats <code>snake_case</code> as one word. In Ruby <code>'_'</code> are treated as part of words. <ul style="list-style-type: none"> With a prefix argument ARG, enable superword mode if ARG is positive, and disable it otherwise. PEL provides the <code><f12> M-p</code> key for the programming language modes where <code>snake_case</code> is popular (Emacs Lisp, C, C++, Erlang, Python, Ruby, etc...)

Emacs & Ruby — References

Document	Notes
Ruby Programming Language	<ul style="list-style-type: none"> Ruby @ Wikipedia Ruby Homepage
Notes on Emacs Ruby Support	<ul style="list-style-type: none"> Ruby Mode @ Emacs Wiki Ruby On Rails @ Emacs Wiki
LSP Support	<ul style="list-style-type: none"> LSP Mode for Ruby solargraph @ GitHub
Blogs on adding support for Ruby	<ul style="list-style-type: none"> Getting Started with Emacs for Ruby , by Horace William, 07 June 2016. Ruby and Emacs Tip: Advanced Pry Integration, from Thiago Araújo Silva, Aug 27, 2018 and updated May 11, 2019
Tools for Ruby	
<ul style="list-style-type: none"> Pry - an alternative for the Ruby IRB shell 	<ul style="list-style-type: none"> Pry @ GitHub Pry Home Page