Aligning Text Vertically

Operation	Keystroke	Function	Note

Align text vertically	Emacs provides several ways to align text vertically.
	• One set of commands are provided by the built-in **align.el** library provides the following commands, all based on alignment regular expression rules.
	  • These can be very useful for source code and some style of comments.
	  • The commands are listed in the rows below.
	• There are **other ways** to align code.
	  • in Emacs the TAB key is used to indent semantically, so it can be used to align the left side of the code. See ∑ **Indentation**
	  • It's also possible to use the multiple-cursor mode and delete or add whitespace. See ∑ **Cursor**.
	  • The artist and picture modes allows you to move the cursor in "void" space, after the end of a line, that can also help. See ∑ **Drawing**.
	• Automatic vertical alignment of text can be done when the M-RET key sequence is typed if that behaviour is enabled in the current buffer. You can either use the <**f11> M-RET** command to toggle this behaviour or have it automatically enabled for a set of major modes.
	Activate automatic text alignment on M-RET for a specified major mode by adding the name of the mode in the **pel-mode-activating-align-on-M-RET** user option. You can open the specific customization buffer by typing **<f11> t a <f2>**		

Open local copy of this Align Text PDF file. See also: ∑ **Help/Info**	**<f11> t a <f1>**	(**pel-help-pdf**)	Open the PEL PDF file(s) for the current context. It opens the local copy of this file.
Open PEL align customization group. See also:∑ **Customize**	**<f11> t a <f2>**	(**pel-customize-pel** &optional OTHER-WINDOW)	Open the PEL customize group(s) for the current context. Use this to open to change PEL user option variables the activate and control the various abbreviations features. • When a prefix argument (like **C-u**) opens the buffer inside another window.
Customize Emacs built-in align support See also:∑ **Customize**	**<f11> t a <f3>**	(**pel-customize-library** &optional OTHER-WINDOW)	Customize Emacs **abbrev** group which includes: abbrev-mode, dynamic-abbreviations,expand,hippie-expand, quickurl. • When a prefix argument (like **C-u**) opens the buffer inside another window.
Align text based on alignment rules	**<f11> t a a**	(**align** BEG END &optional SEPARATE RULES EXCLUDE-RULES)	Attempt to align a region based on a set of alignment rules. • BEG and END mark the region. If BEG and END are specifically set to nil (this can only be done programmatically), the beginning and end of the current alignment section will be calculated based on the location of point, and the value of 'align-region-separate' (or possibly each rule's 'separate' attribute). • If SEPARATE is non-nil, it overrides the value of 'align-region-separate' for all rules, except those that have their 'separate' attribute set. • RULES and EXCLUDE-RULES, if either is non-nil, will replace the default rule lists defined in 'align-rules-list' and 'align-exclude-rules-list'. See 'align-rules-list' for more details on the format of these lists.

Align contiguous lines of text using pre-defined aligning rules.
 	
Default rules include variable assignment alignment, comment alignment, etc…
	<f11> t a c	(**align-current** &optional RULES EXCLUDE-RULES)	Call 'align' on the current alignment section.
			• This function assumes you want to align only the current section, and so saves you from having to specify the region. If RULES or EXCLUDE-RULES is set to a list of rules (see '**align-rules-list**'), it can be used to override the default alignment rules that would have been used to align that section.
			• This command aligns a set of assignment statements using programming language specific rules. For example it would align the '=' of several C variable assignment statements on contiguous lines. Lines separated by spaces are not considered inside the same group. Use **align-entire** with a region for that.
			  • With point over a location anywhere on the following lines of C code in a buffer in c-mode:
			
			`count_lines=0;  // first comment`
			`idx=&lines[0];  // second comment`
			`completed=FALSE;  // last comment`
			
			Running this command transforms the text to the following:
			
			`count_lines = 0;  // first comment`
			`idx  = &lines[0];  // second comment`
			`completed  = FALSE;  // last comment`
			
			• The command uses a set of pre-defined alignment rules made of regular expressions and applies all rules that match. The rules are major-mode specific.
			• It is possible to add more rules and to customize them. The rules are stored in the **align-rules-list** user option.

Align all lines of text in the current region	**<f11> t a e**	(**align-entire** BEG END &optional RULES EXCLUDE-RULES)	Align the selected region as if it were one alignment section. • BEG and END mark the extent of the region. If RULES or EXCLUDE-RULES is set to a list of rules (see 'align-rules-list'), it can be used to override the default alignment rules that would have been used to align that section. • The command uses a set of pre-defined alignment rules made of regular expressions and applies all rules that match. The rules are major-mode specific. • It is possible to add more rules and to customize them. The rules are stored in the **align-rules-list** user option.
Align statement with above lines and add new line and indent.	**<f11> t a l**	(**align-newline-and-indent**)	A replacement function for 'newline-and-indent', aligning as it goes. • The alignment is done by calling 'align' on the region that was indented. • Use **<f11> M-RET** to toggle the behaviour of the M-RET key to make it align as well as inserting new lines and indent.
Align a set of lines on some text	**<f11> t a r**	(**align-regexp** BEG END REGEXP &optional GROUP SPACING REPEAT)	Align the current region using an ad-hoc rule read from the minibuffer. BEG and END mark the limits of the region. Prompts for the regular expression REGEXP to align with. • First select a region, then issue the command. For example, to align assignment of variables over the equal sign use **=** as the *regexp*. • The PEL package creates the **ar** alias for **align-regexp**, so it's also possible to invoke it with **M-x ar RET**
Highlight whitespace that a specific rule would modify	**<f11> t a h**	(**align-highlight-rule** BEG END TITLE &optional RULES EXCLUDE-RULES)	Highlight the whitespace which a given rule would have modified. • BEG and END mark the extent of the region. TITLE identifies the rule that should be highlighted. If RULES or EXCLUDE-RULES is set to a list of rules (see 'align-rules-list'), it can be used to override the default alignment rules that would have been used to identify the text to be colored.
Remove highlighting created by above command	**<f11> t a H**	(**align-unhighlight-rule**)	Remove any highlighting that was added by 'align-highlight-rule'.
Show state of pel-newline-and-indent-below See also: ∑ **Indentation**	**<f11> t a ?**	(**pel-show-if-newline-aligns**)	Display the behaviour of M-RET in the current buffer: show if that command aligns text or not. Print the information in the echo area.
Toggle text alignment on pel-newline-and-indent-below See also: ∑ **Indentation**	**<f11> M-RET**	(**pel-toggle-newline-indent-align**)	Toggle variable 'pel-newline-does-align' for the local buffer. This toggles the way function 'pel-newline-and-indent-below' operates.

Operation	Keystroke	Function	Note
Insert an indented line below current line See also: **Indentation**	• `M-RET` • `<f11> <tab> RET`	(pel-newline-and-indent-below)	Insert an indented line just below current line. • The command can also align text vertically if this special mode was activated for the buffer with the `<f11> M-RET`. • Use `<f11> t a ?` to display whether the `M-RET` command aligns text or not.

Aligning Text — Reference

Topic/URL	Comment
ErgoEmacs — Emacs: Align Text	Xah Lee's page provides a good and simple description on how to align (and sort) tables of text using the align-regex command.
EmacsWiki — Align Command	