




# Indenting & Tab

Description	Keystroke	Function	Note
Indentation under Emacs	 Emacs controls indentation via major modes and the tab key may take different and surprising behaviour for people just starting with Emacs. This table is in early stage of development, more to come with documentation of the behaviour for different modes.		
Insert Literal Tab	<b>C-q &lt;tab&gt;</b>	(quoted-insert ARG) <tab>	Inserts a hard tab inside the file but moves the cursor to the column that represents the next multiple of <i>tab-width</i> .
Behaviour of Tab	In Emacs the behaviour of the <tab> key depends on the major mode of the current buffer. This key is rebound by several major mode.  By default, in text modes, tabs are set to 8 spaces, inserting hard tabs. However, if there is text in the above lines, the tab moves to the spot under the word above. Note that if a line is full of text (without any space), then the tab stops controlled by the ruler take effect again.		
Indent current line (or region)	<b>&lt;tab&gt;</b>	(indent-for-tab-command &optional ARG)	Indent the current line or region, or insert a tab, as appropriate. <ul style="list-style-type: none"><li>This function either inserts a tab, or indents the current line, or performs symbol completion, depending on ‘tab-always-indent’. The function called to actually indent the line or insert a tab is given by the variable ‘<b>indent-line-function</b>’.</li><li>If a prefix argument is given, after this function indents the current line or inserts a tab, it also rigidly indents the entire balanced expression which starts at the beginning of the current line, to reflect the current line’s indentation.</li><li>In most major modes, if point was in the current line’s indentation, it is moved to the first non-whitespace character after indenting; otherwise it stays at the same position relative to the text.</li><li>If ‘transient-mark-mode’ is turned on and the region is active, this function instead calls ‘indent-region’. In this case, any prefix argument is ignored.</li></ul>  The behaviour of the tab key vastly differ between major modes. This ranges from not moving the cursor at all if the indentation is identified as correct for the current context, to cycling through various potential positions to just what someone new to Emacs would expect. Much more has to be documented on the behaviour of that key and how it can be controlled and customized. It’s quite possible that the best way to document its behaviour would be to place a description inside the table of each major mode.
		indent-for-tab-command &optional ARG)	In <b>Lisp</b> related modes. <ul style="list-style-type: none"><li>indent-line-function = indent-relative.</li><li>tab-always-indent = t</li></ul>  The above values that I got in Emacs via inspection do not explain tab behaviour in the Emacs Lisp mode (which is to indent the code according to Emacs Lisp semantics, a <b>very</b> useful feature when writing Lisp code). In this mode tab corrects the indentation of the code at the current line, which may be to indent, de-indent or do nothing.
		(c-indent-line-or-region &optional ARG REGION)	In C related modes.  Indent active region, current line, or block starting on this line. <ul style="list-style-type: none"><li>In Transient Mark mode, when the region is active, reindent the region.</li><li>Otherwise, with a prefix argument, rigidly reindent the expression starting on the current line.</li><li>Otherwise reindent just the current line.</li></ul>
Indent lines of list after point  (CLBC s3.lisp)	<b>C-M-q</b>	(indent-sexp &optional ENDPOS)	Indent each line of the list starting just after point. <ul style="list-style-type: none"><li>If optional arg ENDPOS is given, indent each line, stopping when ENDPOS is encountered.</li></ul>
Insert spaces or tabs to next defined tab-stop column	<b>M-i</b>	(tab-to-tab-stop)	Works in following modes: <ul style="list-style-type: none"><li>*.c: C-mode : every 4, inserts spaces</li><li>C++ mode: every 4</li><li>Erlang mode: every 4</li><li>*.py: Python mode: every 8, inserts spaces</li><li>Emacs-Lisp: every 4: (4 8 12 16)</li><li>Haskell: every 8: (8 16 24 32)</li><li>text: every 4: (4 8 12 16)</li><li>reStructuredText: every 4</li><li>org mode: every 4</li><li>Fundamental mode: every 4</li><li>makefile, *.mk: BSDmakefile mode: shows every 4, inserts tabs. &lt;tab&gt; inserts tabs and moves cursor.</li></ul>
Insert an indented line below current line	<ul style="list-style-type: none"><li><b>M-&lt;ret&gt;</b></li><li><b>&lt;f11&gt; &lt;tab&gt; &lt;ret&gt;</b></li></ul>	(pel-newline-and-indent-below)	Insert an indented line just below current line.
Change the tab stops	<b>M-x edit-tab-stops</b>		Opens a <b>*Tab Stops* buffer</b> . Identify the tab stops in the first line with colons. Use C-c C-c to activate and exit the buffer. Again, the tab stop take effect at the top of the buffer,
Change the tab width	<b>M-: (setq tab-width N)</b>		The variable <i>tab-width</i> normally defaults to 8 in emacs. It can be set locally inside a buffer with (setq tab-width N) or globally with (setq-default tab-width N). The M-: keystroke allows evaluating a lisp expression interactively (as the above two).  Note that any literal tab in the buffer impact the location of the column where the next character shows. When changing the tab-width, the layout shown in the window that contains literal tabs will be modified according to the new tab-width value.
Make <tab> insert space/tab	<b>M-: (setq indent-tabs-mode nil/t)</b>		By default, pressing <tab> insert literal (hard) tabs inside the file. The indent-tabs-mode variable controls that: set to t it inserts tabs, set to nil it inserts spaces.
Next line, indented	<b>C-j</b>	(electric-newline-and-maybe-indent)	Add new line and indent next line. Indentation is controlled by the variable <b>left-margin</b> . Pressing <b>Tab</b> anywhere on the line also indents the line properly.
Indent Region	<b>C-M-\</b>	(indent-region START END &optional COLUMN)	Indent each nonblank line in the region. <ul style="list-style-type: none"><li>A numeric prefix argument specifies a column: indent each line to that column.</li><li>With no prefix argument, the command chooses one of these methods and indents all the lines with it:<ol style="list-style-type: none"><li>If ‘fill-prefix’ is non-nil, insert ‘fill-prefix’ at the beginning of each line in the region that does not already begin with it.</li><li>If ‘indent-region-function’ is non-nil, call that function to indent the region.</li><li>Indent each line via ‘indent-according-to-mode’.</li></ol></li></ul>



Indentation — References

Title & URL	Description
<b><u>Understanding GNU Emacs and Tabs</u></b>	Overview description of how Emacs handle the Tab key, often used for strict indentation in many editors. In Emacs it can do much more.
<b><u>GNU Emacs Manual - Indentation</u></b>	
<b><u>GNU Emacs Manual - Indentation for Programs</u></b>	
<b>Indentation Styles</b>	
<b><u>Indentation Styles @ Wikipedia</u></b>	
<b><u>StackOverflow - Emacs BSD/Allman Style with 4 Space Tabs?</u></b>	
<b><u>GNU Emacs Manual - Styles</u></b>	
<b><u>Emacs BSD/Allman Style with 4 Space Tabs?</u></b>	
<b><u>Emacs: Linux Kernel Style but with Allman/BSD Style Braces?</u></b>	
<b><u>Emacs Wiki - Indenting C</u></b>	
<b><u>Indent preprocessor directives as C code in emacs</u></b>	Does not fully address the way I want to have multi-indentations for pre-processor
<b><u>elisp code - ppindent.el</u></b>	Implements pre-processor indentation with the # always in the first column. Not yet exactly what I want.
<b><u>Demystify C++ Metaprograms using Emacs</u></b>	
<b><u>Programming in C++, Rules and Recommendations</u></b>	ellemtel style