













Description	Keystroke	Function	Note																							
Indenting and un-indenting rigidly	The following commands provide non-semantic indentation of the current line or marked region. <ul style="list-style-type: none"><li>The first command allows you to use further keystrokes to fine-tune the indentation back and forth using cursor keys. That’s probably all you ever need to use.</li><li>Currently, PEL also provides the last 2 commands that indent or un-indent the current line or marked region. Once used, the region remains marked to allow further use of the command.</li></ul>																									
Indent/Unindent rigidly  See also: <a href="#">🔗 Key-Chords</a>	<ul style="list-style-type: none"><li><b>C-x &lt;tab&gt;</b></li><li><b>&lt;f11&gt; &lt;tab&gt; &lt;tab&gt;</b></li><li><b>&lt;tab&gt;q</b></li></ul>	<b>(pel-indent-rigidly &amp;optional N)</b>  ----- PEL uses the above instead of the standard: <b>(indent-rigidly START END ARG &amp;optional INTERACTIVE)</b>	Enter a mode to indent rigidly the marked region or current line N times. <ul style="list-style-type: none"><li><b>If a region is marked</b>, it uses ‘indent-rigidly’ and provides the same prompts to control indentation changes.</li><li><b>If no region is marked</b>, it operates on current line(s) identified by the numeric argument N (or if not specified N=1):<ul style="list-style-type: none"><li>N = [-1, 0, 1] : operate on current line</li><li>N &gt; 1 : operate on the current line and N-1 lines below.</li><li>N &lt; -1 : operate on the current line and (abs N) -1 lines above.</li></ul></li></ul> Once the command is issued use the keys listed below to indent or un-indent by indent-width step or by single column.  ----- Indent all lines starting in the region. <ul style="list-style-type: none"><li>If called interactively with no prefix argument, activate a transient mode in which the indentation can be adjusted interactively by typing <b>&lt;left&gt;</b>, <b>&lt;right&gt;</b>, <b>S-&lt;left&gt;</b>, or <b>S-&lt;right&gt;</b>.</li></ul>																							
PEL rebinds this key, but extends the functionality: <b>pel-indent-rigidly</b> uses indent-rigidly, described below the dashed line.  See also: <ul style="list-style-type: none"><li><a href="#">🔗 I - C</a></li><li><a href="#">🔗 I - C++</a></li><li><a href="#">🔗 I - D</a></li><li><a href="#">🔗 reStructuredText</a></li></ul>	Both of these commands activate a transient mode where Emacs prompts for extra keys to control how to indent or un-indent. The capabilities are controlled by the variable <i>indent-rigidly-map</i> with by default provides: <table><tr><td><ul style="list-style-type: none"><li><b>S-&lt;right&gt;</b> indent-rigidly-right-to-tab-stop</li><li><b>&lt;right&gt;</b> indent-rigidly-right</li></ul></td><td><b>S-&lt;left&gt;</b> indent-rigidly-left-to-tab-stop</td><td><b>&lt;left&gt;</b> indent-rigidly-left</td></tr></table> Typing any other key deactivates the transient mode. <ul style="list-style-type: none"><li>The <b>S-&lt;right&gt;</b> and <b>S-&lt;left&gt;</b> keys indent/de-indent to the next tab-stop position, which is controlled by the <b>tab-width</b> user option.<ul style="list-style-type: none"><li>With PEL, for several major modes, the indentation is controlled by a mode-specific user option variable . For example, for buffers in c-mode, the value of <b>pel-c-tab-width</b> is automatically stored into tab-width when the buffer is opened.</li></ul></li></ul> If you use the cua-mode: the cua-mode uses <b>C-x</b> , to invoke this command when cua-mode is active, type it really fast or type <b>C-x C-x &lt;tab&gt;</b> (or use the PEL binding <b>&lt;f11&gt; &lt;tab&gt; &lt;tab&gt;</b> ).  With PEL, the <b>&lt;tab&gt;q</b> key-chord is available when <b>pel-use-key-chord</b> is non-nil. See <a href="#">🔗 Key-Chords</a> . Command numeric prefix <b>is available</b> with the key-chord binding.			<ul style="list-style-type: none"><li><b>S-&lt;right&gt;</b> indent-rigidly-right-to-tab-stop</li><li><b>&lt;right&gt;</b> indent-rigidly-right</li></ul>	<b>S-&lt;left&gt;</b> indent-rigidly-left-to-tab-stop	<b>&lt;left&gt;</b> indent-rigidly-left																				
<ul style="list-style-type: none"><li><b>S-&lt;right&gt;</b> indent-rigidly-right-to-tab-stop</li><li><b>&lt;right&gt;</b> indent-rigidly-right</li></ul>	<b>S-&lt;left&gt;</b> indent-rigidly-left-to-tab-stop	<b>&lt;left&gt;</b> indent-rigidly-left																								
Indent line(s) rigidly	<ul style="list-style-type: none"><li><b>&lt;f6&gt; &lt;tab&gt;</b></li><li><b>&lt;f11&gt; &lt;tab&gt; c</b></li></ul>	<b>(pel-indent-lines &amp;optional N)</b>	Indent current or marked lines by N indentation levels																							
Un-indent line(s) rigidly	<ul style="list-style-type: none"><li><b>&lt;backtab&gt;</b></li><li><b>&lt;f6&gt; &lt;backtab&gt;</b></li><li><b>&lt;f11&gt; &lt;tab&gt; C</b></li></ul>		<ul style="list-style-type: none"><li>Un-indent current line or marked lines by N indentation levels.</li></ul>																							
	<ul style="list-style-type: none"><li>Works with point anywhere on the line.</li><li>All lines touched by the region are indented.</li><li>A special argument N can specify more than one indentation level. It defaults to 1.</li><li>If a negative number is specified, ‘pel-unindent-lines’ is used.</li><li>If a region is marked, the function does not deactivate it to allow repeated execution of the command. It also modifies the region to include all characters in all affected lines.</li><li>Use <b>C-g</b> to de-activate the region.</li><li>Handles presence of hard tabs:<ul style="list-style-type: none"><li>If indent-tabs-mode is non-nil the indentation is created with a mix of hard-tabs and space characters.</li><li>If indent-tabs-mode is nil, any hard tab in the indentation of the marked lines is replaced by the proper number of spaces. Hard tabs after first non-whitespace character on the line are left.</li></ul></li></ul>																									
<a href="#">Indent-tools</a>	The <a href="#">indent-tools</a> external package provides several commands to indent, un-indent and navigate across indented text levels. <ul style="list-style-type: none"><li>It provides a minor mode and a key <a href="#">hydra</a> that provides all of these commands.</li></ul> The <a href="#">indent-tools</a> external package  PEL activates it when the <b>pel-use-indent-tools</b> user-option is turned on (set to <b>t</b> ). <ul style="list-style-type: none"><li>This also automatically activates the <a href="#">hydra</a> external package.</li></ul> PEL provide a global key binding to its key <a href="#">hydra</a> and provides the ability to activate the proposed key binding globally and for python mode: <ul style="list-style-type: none"><li><b>pel-indent-tools-key-bound</b> : activates the <b>C-c &gt;</b> key binding either globally or for python-mode only.</li></ul>																									
Open the indent-tools hydra  See also: <a href="#">🔗 I - Python</a>	<ul style="list-style-type: none"><li><b>&lt;f11&gt; &lt;tab&gt; &lt;f7&gt;</b></li><li><b>&lt;f7&gt; &lt;tab&gt;</b></li><li><b>C-c &gt;</b></li></ul>	<b>(indent-tools-hydra/body)</b>	Activate the "indent-tools-hydra" hydra.  With PEL, this key binding is only available when: <ul style="list-style-type: none"><li>globally, when <b>pel-indent-tools-key-bound</b> is set to <b>globally</b>,</li><li>in python-mode only when <b>pel-indent-tools-key-bound</b> is set to <b>python</b>.</li><li>The actual key is selected by indent-tools <b>indent-tools-keymap-prefix</b> user-option, the default is <b>C-c &gt;</b></li></ul>																							
See also: <a href="#">🔗 Hide/Show</a>	The heads for the associated hydra are: <pre>&gt;: 'indent-tools-indent', &lt;: 'indent-tools-demote', E: 'indent-tools-indent-end-of-defun', c: 'indent-tools-comment', U: 'indent-tools-uncomment', P: 'indent-tools-indent-paragraph', l: 'indent-tools-indent-end-of-level', K: 'indent-tools-kill-tree', C: 'indent-tools-copy-hydra/body', s: 'indent-tools-select', e: 'indent-tools-goto-end-of-tree', u: 'indent-tools-goto-parent', d: 'indent-tools-goto-child', S: 'indent-tools-select-end-of-tree', n: 'indent-tools-goto-next-sibling', p: 'indent-tools-goto-previous-sibling', i: 'helm-imenu', j: 'forward-line', k: 'previous-line', SPC: 'indent-tools-indent-space', _: 'undo-tree-undo', L: 'recenter-top-bottom', f: 'yafolding-toggle-element', q: exit</pre>																									
	<table><tr><th>Indent</th><th>Navigation</th><th>Actions</th></tr><tr><td>&gt; indent</td><td>j v</td><td>K kill</td></tr><tr><td>&lt; de-indent</td><td>k A</td><td>i imenu</td></tr><tr><td>l end of level</td><td>n next sibling</td><td>C Copy...</td></tr><tr><td>E end of fn</td><td>p previous sibling</td><td>c comment</td></tr><tr><td>P paragraph</td><td>u up parent</td><td>U uncomment (paragraph)</td></tr><tr><td>SPC space</td><td>d down child</td><td>f fold</td></tr><tr><td>_ undo</td><td>e end of tree</td><td>q quit</td></tr></table> The <b>f</b> key toggles the element folding. Press once to hide the sub-tree, press-again to display it back.			Indent	Navigation	Actions	> indent	j v	K kill	< de-indent	k A	i imenu	l end of level	n next sibling	C Copy...	E end of fn	p previous sibling	c comment	P paragraph	u up parent	U uncomment (paragraph)	SPC space	d down child	f fold	_ undo	e end of tree
Indent	Navigation	Actions																								
> indent	j v	K kill																								
< de-indent	k A	i imenu																								
l end of level	n next sibling	C Copy...																								
E end of fn	p previous sibling	c comment																								
P paragraph	u up parent	U uncomment (paragraph)																								
SPC space	d down child	f fold																								
_ undo	e end of tree	q quit																								



Description	Keystroke	Function	Note
<a href="#">indent-bars</a>	<ul style="list-style-type: none"> <li>A minor-mode that displays vertical bars over each indentation level. The style of the bars is customizable.</li> </ul> <div>  The <a href="#">indent-bars</a> external package            PEL activates it when the <a href="#">pel-use-indent-bars</a> user-option is turned on (set to <code>t</code>).         </div>		
Toggle indent-bars	<code>&lt;f11&gt; &lt;tab&gt; b b</code>	<code>(indent-bars-mode &amp;optional ARG)</code>	Toggle showing bars that indicate indentation. A minor mode.
Reset indent bar config	<code>&lt;f11&gt; &lt;tab&gt; b r</code>	<code>(indent-bars-reset &amp;rest R)</code>	Reset indent-bars config.
Reset style of indent bars	<code>&lt;f11&gt; &lt;tab&gt; b s</code>	<code>(indent-bars-reset-styles &amp;rest R)</code>	Reset all styles' colors and faces. <ul style="list-style-type: none"> <li>Useful for calling after theme changes.</li> </ul>
<a href="#">Smart-shift</a>	<p>The <a href="#">smart-shift</a> external package simplifies shifting a complete line or region of lines right or left but also up or down.</p> <ul style="list-style-type: none"> <li>It is implemented as a minor or global minor mode that must be enabled first.</li> </ul> <ul style="list-style-type: none"> <li>Automatically activate the smart-shift-mode in specified major mode by customizing the <code>pel-&lt;mode&gt;-activates-minor-modes</code> user-options.</li> <li>You can also use the commands manually or through the key bindings provided by PEL to activate the smart-shift-mode in the current buffer or globally for all buffers.</li> <li>PEL controls it through customization user-options:               <div>  The <a href="#">smart-shift</a> external package                  PEL activates it when the <a href="#">pel-use-smart-shift</a> user-option is turned on (set to <code>t</code>).               </div> </li> </ul> <div>  PEL also provides the <code>pel-smart-shift-keybinding</code> user-option that allows you to select whether the shift keys used by smart-shift mode is the default provided keys only or whether you also want to activate another set.           <ul style="list-style-type: none"> <li>The default are always available when smart-shift mode is active: <code>C-c &lt;right&gt;</code>, <code>C-c &lt;left&gt;</code>, <code>C-c &lt;up&gt;</code> and <code>C-c &lt;down&gt;</code>.</li> <li>PEL can also activate <b>one</b> of the following extra key binding sets:               <ul style="list-style-type: none"> <li>Using the control cursor key : <code>C-c C-&lt;right&gt;</code>, <code>C-c C-&lt;left&gt;</code>, <code>C-c C-&lt;up&gt;</code> and <code>C-c C-&lt;down&gt;</code>.</li> <li>Using the <code>&lt;f9&gt;</code> key as prefix: <code>&lt;f9&gt; &lt;right&gt;</code>, <code>&lt;f9&gt; &lt;left&gt;</code>, <code>&lt;f9&gt; &lt;up&gt;</code> and <code>&lt;f9&gt; &lt;down&gt;</code></li> </ul> </li> </ul> </div>		
ⓘ <b>Customize with:</b> <code>&lt;f11&gt; &lt;tab&gt; s &lt;f2&gt;</code>			
Toggle smart-shift mode in current buffer	<code>&lt;f11&gt; &lt;tab&gt; s</code>	<code>(smart-shift-mode &amp;optional ARG)</code>	Activate/de-activate the smart-shift mode in the current buffer. <ul style="list-style-type: none"> <li>Activate the line-shift key bindings listed below, in the current buffer.               <ul style="list-style-type: none"> <li>With PEL, the actual key binding selected for the line shift commands depend on the value of the <code>pel-smart-shift-keybinding</code> user-option.</li> </ul> </li> </ul>
Toggle smart-shift mode globally	<code>&lt;f11&gt; &lt;tab&gt; S</code>	<code>(global-smart-shift-mode &amp;optional ARG)</code>	<ul style="list-style-type: none"> <li>Toggle Smart-Shift mode in all buffers.</li> <li>With prefix ARG, enable Global Smart-Shift mode if ARG is positive; otherwise, disable it.</li> <li>Smart-Shift mode is enabled in all buffers where 'smart-shift-mode-on' would do it.</li> </ul>
When smart-shift mode is active:	<div>  As described above, with PEL only <b>one</b> of the extra key bindings provided by PEL can be enabled via the <code>pel-smart-shift-keybinding</code> user-option.           </div> So unlike other key binding description cells in this and other tables, only one of the last 2 key bindings is available in the smart-shift minor mode.		
Shift line or region right	<ul style="list-style-type: none"> <li><code>C-c &lt;right&gt;</code></li> <li><code>C-c C-&lt;right&gt;</code></li> <li><code>&lt;f9&gt; &lt;right&gt;</code></li> </ul>	<code>(smart-shift-right &amp;optional ARG)</code>	Shift the line or region to the ARG times to the right.
Shift line or region left	<ul style="list-style-type: none"> <li><code>C-c &lt;left&gt;</code></li> <li><code>C-c C-&lt;left&gt;</code></li> <li><code>&lt;f9&gt; &lt;left&gt;</code></li> </ul>	<code>(smart-shift-left &amp;optional ARG)</code>	Shift the line or region to the ARG times to the left.
Shift line or region up	<ul style="list-style-type: none"> <li><code>C-c &lt;up&gt;</code></li> <li><code>C-c C-&lt;up&gt;</code></li> <li><code>&lt;f9&gt; &lt;up&gt;</code></li> </ul>	<code>(smart-shift-up &amp;optional ARG)</code>	Shift the line or region to the ARG times to the upwards.
Shift line or region down	<ul style="list-style-type: none"> <li><code>C-c &lt;down&gt;</code></li> <li><code>C-c C-&lt;down&gt;</code></li> <li><code>&lt;f9&gt; &lt;down&gt;</code></li> </ul>	<code>(smart-shift-down &amp;optional ARG)</code>	Shift the line or region to the ARG times to the downwards
<a href="#">smart-tabs</a>	<div>  The <a href="#">smart-tabs</a> external package            PEL activates it when the <a href="#">pel-use-smart-tabs</a> user-option is turned on (set to <code>t</code>).         </div>		
Toggle smart-tabs mode	<code>&lt;f11&gt; &lt;tab&gt; M-s</code>	<code>(smart-tabs-mode &amp;optional ARG)</code>	Toggle smart-tabs minor mode. <ul style="list-style-type: none"> <li>Intelligently indent with tabs, align with spaces!</li> </ul>

## Indentation – References

Title & URL	Description
<a href="#">Understanding GNU Emacs and Tabs</a>	Overview description of how Emacs handle the Tab key, often used for strict indentation in many editors. In Emacs it can do much more.
<a href="#">GNU Emacs Manual - Indentation</a>	
<a href="#">GNU Emacs Manual - Indentation for Programs</a>	
<a href="#">Indentation Basic Concepts Tutorial @ XEmacs</a>	A tutorial on indentation written by <a href="#">KaiGrossjohann</a>
<b>Tabs or space for indentation??</b> There are several views on the use of hard-tab and space characters for indenting source code. They are: <ol style="list-style-type: none"> <li>Use only hard-tab for indentation. Uncontrolled use of tabs or spaces for alignment.</li> <li>Use only space characters for indentation. Popular in C like languages. Also popular in Python.</li> <li>Use hard tabs for indentation, and space character for alignment.</li> </ol> <ul style="list-style-type: none"> <li>Method 1 was popular originally since it reduces file size when hard tab size was always the same. But soon it became possible to identify a different number of character positions to render a hard tab. And then it became impossible to guarantee the rendering of code indentation and alignment when the number of hard-tabs did not match the indentation level of a line of source code.</li> <li>A reaction to this problem is to use Method 2 where hard-tabs are banned. The rendering is therefore always the same no matter what the <i>size</i> of a hard tab is since you don't use any. This however increases the size of files. Not a problem for storage today you'd say, but perhaps a problem for data transfer and/or power consumption.</li> <li>Method 3 is used by some programming environments. The Go programming language imposes the use of hard-tabs for indentation. And if you want to align text at the right of the indentation level, you use spaces.               <ul style="list-style-type: none"> <li>To use this method in other programming languages, you can use the smart-tabs-mode explained in the <a href="#">Smart-Tabs Emacs Wiki page</a>.</li> </ul> </li> </ul> <p>Emacs support all modes. It has 2 different buffer local variables that are important and control the rendering of hard-tabs and the indentation:</p> <ul style="list-style-type: none"> <li><b>tab-width:</b> How many columns a hard-tab occupies, the distance between tab-stops.</li> <li>indentation offset variable: a variable for each major mode, like <b>c-basic-offset</b> for CC modes (C, C++, Java, etc...), that identifies the number of columns per indentation level.</li> </ul> <p>PEL provides access to the smarttabs package but it's not yet fully configured for programming modes nor tested. 🐛 For CC modes it provides PEL user-options that control the indentation using method 2.</p> <p>Using method 3 requires a better understanding from all developers working on the source code with all their editors being able to handle the mix of hard tab and space characters correctly.</p>	
<a href="#">Smarttabs @ GitHub</a>	Starttabs source code repository.
<a href="#">Indentation Styles for Curly Bracket Languages</a>	
<a href="#">Indentation Styles @ Wikipedia</a>	
<a href="#">StackOverflow - Emacs BSD/Allman Style with 4 Space Tabs?</a>	
<a href="#">GNU Emacs Manual - Styles</a>	
<a href="#">Emacs BSD/Allman Style with 4 Space Tabs?</a>	
<a href="#">Emacs: Linux Kernel Style but with Allman/BSD Style Braces?</a>	

Title & URL	Description
<b>Emacs Wiki - Indenting C</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