












Operation	Keystroke	Function	Note
<b>Smartparens Mode</b> • <a href="#">Smartparens manual</a>  See also: <a href="#">Smartparens</a>	Simplify insertion of matching pairs with the <a href="#">smartparens</a> minor mode. PEL binds a set of keys, described below, to toggle activation of that mode.  This uses the <a href="#">smartparens</a> external package.  PEL activates it when <a href="#">pel-use-smartparens</a> is set to <b>t</b> . <ul style="list-style-type: none"> <li>Mode line lighter: <ul style="list-style-type: none"> <li>smartparens-mode: SP</li> <li>smartparens-strict-mode: SP/s</li> </ul> </li> </ul>		
Help on smartparens	<f11> ( ?	(sp-cheat-sheet &optional ARG)	Generate a cheat sheet of all the smartparens interactive functions. Shows inside Emacs buffer. <ul style="list-style-type: none"> <li>Without a prefix argument, print only the short documentation and examples.</li> <li>With non-nil prefix argument ARG, show the full documentation for each function.</li> <li>You can follow the links to the function or variable help page. <ul style="list-style-type: none"> <li>To get back to the full list, use M-x help-go-back.</li> </ul> </li> <li>You can use ‘beginning-of-defun’ and ‘end-of-defun’ to jump to the previous/next entry.</li> <li>Examples are fontified using the ‘font-lock-string-face’ for better orientation.</li> </ul>
Describe user system	<f11> ( M-?	(sp-describe-system STARTERKIT)	Describe user’s system. Prompt for starter kit: Evil, Spacemacs, Vanilla. <ul style="list-style-type: none"> <li>The output of this function can be used in bug reports.</li> </ul>
Toggle smartparens mode	<f11> ( (	(smartparens-mode &optional ARG)	Toggle smartparens mode.
Toggle smartparens-strict mode	<f11> ( )	(smartparens-strict-mode &optional ARG)	Toggle the strict smartparens mode. <ul style="list-style-type: none"> <li>When strict mode is active, ‘delete-char’, ‘kill-word’ and their backward variants will skip over the pair delimiters in order to keep the structure always valid (the same way as ‘paredit-mode’ does). This is accomplished by remapping them to ‘sp-delete-char’ and ‘sp-kill-word’. There is also function ‘sp-kill-symbol’ that deletes symbols instead of words, otherwise working exactly the same (it is not bound to any key by default).</li> <li>When strict mode is active, this is indicated with “/s” after the smartparens indicator in the mode list</li> </ul>
Toggle smartparens mode	<f11> ( M-(	(smartparens-global-mode &optional ARG)	Toggle Smartparens mode in all buffers. <ul style="list-style-type: none"> <li>With prefix ARG, enable Smartparens-Global mode if ARG is positive; otherwise, disable it.</li> <li>Smartparens mode is enabled in all buffers where ‘turn-on-smartparens-mode’ would do it.</li> </ul>
Toggle smartparens-strict mode	<f11> ( M-)	(smartparens-global-strict-mode &optional ARG)	Toggle Smartparens-Strict mode in all buffers. <ul style="list-style-type: none"> <li>With prefix ARG, enable Smartparens-Global-Strict mode if ARG is positive; otherwise, disable it.</li> <li>Smartparens-Strict mode is enabled in all buffers where ‘turn-on-smartparens-strict-mode’ would do it.</li> </ul>
<b>Smart-shift</b>  See also: <a href="#">Indentation</a>	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. <ul style="list-style-type: none"> <li>It is implemented as a minor or global minor mode that must be enabled first. You can identify the smart-shift-mode inside one of the pel-&lt;mode&gt;-activates-minor-modes user-options to activate it automatically. 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: <ul style="list-style-type: none"> <li> 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 t).</li> <li> PEL also provides the <a href="#">pel-smart-shift-keybinding</a> user-option that allows you to select additional alternative key bindings for the smart-shift commands that shift line(s). By default the key bindings are using <b>C-c</b> as a key prefix. With PEL you can also use a control key for the cursor or change the prefix key to use the <b>&lt;f9&gt;</b> key. The 3 possible key bindings are shown below but only one of them will be available at any given time. The one available is the one selected by the user-option value.</li> </ul> </li> </ul>		
Toggle smart-shift mode in current buffer	<f11> <tab> s	(smart-shift-mode &optional ARG)	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 <b>pel-smart-shift-keybinding</b> user-option.</li> </ul> </li> </ul>
Toggle smart-shift mode globally	<f11> <tab> S	(global-smart-shift-mode &optional ARG)	<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>
Shift line or region right	<ul style="list-style-type: none"> <li>C-c &lt;right&gt;</li> <li>C-c C-&lt;right&gt;</li> <li>&lt;f9&gt; &lt;right&gt;</li> </ul>	(smart-shift-right &optional ARG)	Shift the line or region to the ARG times to the right.  With PEL <b>one</b> of the extra key bindings can be enabled via the <b>pel-smart-shift-keybinding</b> user-option. So unlike other cells only one of the last 2 key bindings is available in the smart-shift minor mode.
Shift line or region left	<ul style="list-style-type: none"> <li>C-c &lt;left&gt;</li> <li>C-c C-&lt;left&gt;</li> <li>&lt;f9&gt; &lt;left&gt;</li> </ul>	(smart-shift-left &optional ARG)	Shift the line or region to the ARG times to the left.  With PEL <b>one</b> of the extra key bindings can be enabled via the <b>pel-smart-shift-keybinding</b> user-option. So unlike other cells only one of the last 2 key bindings is available in the smart-shift minor mode.
Shift line or region up	<ul style="list-style-type: none"> <li>C-c &lt;up&gt;</li> <li>C-c C-&lt;up&gt;</li> <li>&lt;f9&gt; &lt;up&gt;</li> </ul>	(smart-shift-up &optional ARG)	Shift the line or region to the ARG times to the upwards.  With PEL <b>one</b> of the extra key bindings can be enabled via the <b>pel-smart-shift-keybinding</b> user-option. So unlike other cells only one of the last 2 key bindings is available in the smart-shift minor mode.
Shift line or region down	<ul style="list-style-type: none"> <li>C-c &lt;down&gt;</li> <li>C-c C-&lt;down&gt;</li> <li>&lt;f9&gt; &lt;down&gt;</li> </ul>	(smart-shift-down &optional ARG)	Shift the line or region to the ARG times to the downwards  With PEL <b>one</b> of the extra key bindings can be enabled via the <b>pel-smart-shift-keybinding</b> user-option. So unlike other cells only one of the last 2 key bindings is available in the smart-shift minor mode.

## YAML & Emacs – References

Description & URL	Notes
<b>YAML</b>	
<b>YAML @ Wikipedia</b>	Overview, syntax, criticisms
<b>YAML official home page</b>	Links to YAML specification, links to various resources and projects. <ul style="list-style-type: none"> <li><a href="#">YAML 1.2 Specs</a></li> <li><a href="#">YAML 1.1 Specs</a></li> <li><a href="#">YAML 1.0 Specs</a></li> </ul>

Description & URL	Notes
YAML Resource sites	<ul style="list-style-type: none"> <li>• <a href="#">Learn YAML in Y Minutes</a></li> <li>• Online YAML validator (runs <a href="#">yamllint.py</a>) ⚠️ No link as the site is not using https. Instead install <a href="#">yamllint.py</a> locally and use it on the command line or via Emacs.</li> </ul>
StrictYAML	A stricter, type-safe YAML
StrictYAML @ Github	
StrictYAML @ hitchdev (Python libraries)	
RAML	RESTful API Modeling Language : RAML files have the .raml file extension.
	<ul style="list-style-type: none"> <li>• <a href="#">RAML @ Wikipedia</a></li> <li>• <a href="#">RAML.org</a></li> <li>• <a href="#">RAML Spec @ GitHub</a></li> </ul>
Common Workflow Language	Common Workflow Language (CWL) uses a <a href="#">subset of YAML</a> and provides YAML supporting tools.
See also: <a href="#">M CWL</a>	<ul style="list-style-type: none"> <li>• <a href="#">CWL home page</a> <ul style="list-style-type: none"> <li>• <a href="#">CWL User Guide</a></li> <li>• <a href="#">CWL YAML Guide</a></li> </ul> </li> </ul>
Emacs support for YAML	
yaml-mode (major mode for YAML)	<ul style="list-style-type: none"> <li>• <a href="#">yaml-mode @ GitHub</a></li> <li>• <a href="#">Yaml Mode @ Emacs Wiki</a></li> </ul>
indent-tools	<ul style="list-style-type: none"> <li>• <a href="#">indent-tools @ GitLab</a></li> <li>• <a href="#">indent-tools @ Melpa</a></li> </ul>
smartparens	<p>The smartparens mode can help deal with data that is within matching pair of characters.</p> <ul style="list-style-type: none"> <li>• <a href="#">smartparens @ GitHub</a></li> <li>• <a href="#">smartparens documentation</a></li> </ul>
Emacs/YAML Support Articles	
Blogs about YAML editing on Emacs	<ul style="list-style-type: none"> <li>• <a href="#">The best ways to work with yaml files in Emacs</a>, from Chmouel Boudjnah's blog, 2016-09-07</li> <li>• <a href="#">Editing ansible files in Emacs</a>, from Enis Özgen, 2017-12-29</li> </ul>
General blogs about YAML	<ul style="list-style-type: none"> <li>• <a href="#">10 YAML tips for people who hate YAML</a> <ul style="list-style-type: none"> <li>• BTW, the last tip is: use something else... well... S-expressions are very flexible and powerful.</li> </ul> </li> </ul>