
















YAML Data Serialization Support

Operation	Keystroke	Function	Note																											
Editing <u>YAML</u> files See also: <u>M</u> <u>CWL</u>	Long <u>YAML</u> files are notoriously difficult to edit properly. Use the external package <u>yaml-mode</u> , a major mode for <u>YAML</u> files. Also use a couple of minor-modes and commands listed in this page to help. <ul style="list-style-type: none">Aside from the first 3 key bindings listed to access help and customization buffers for <u>YAML</u>, the key bindings listed in this page and their related commands are also described in other <u>PEL</u> PDF pages. The links to these pages are on the first column. <div> The <u>yaml-mode</u> external package provides a major mode support for <u>YAML</u>. <u>PEL</u> provides access to it when the <u>pel-use-yaml</u> user-option is turned on (set to <u>t</u>).</div> <ul style="list-style-type: none"><u>PEL</u> associates the following file extensions with <u>yaml-mode</u>: <u>.yaml</u>, <u>.yaml</u>, <u>.eyaml</u>, <u>.raml</u>.																													
Last updated on:	2025-12-16																													
Open this PDF file. See also: <u>Z</u> <u>Help/Info</u>	<div><f11> SPC M-y <f1></div> <div><f12> <f1></div>	(<u>pel-help-pdf</u> &optional OPEN-WEB-PAGE)	Open the <u>D</u> <u>YAML</u> local PDF. If the prefix argument (like <u>C-u</u> or <u>M--</u>) is used, then it opens the remote GitHub hosted raw PDF instead. If the <u>pel-flip-help-pdf-arg</u> user-option is set it's the other way around.																											
<u>Z</u> <u>Customize</u> <u>PEL</u> <u>YAML</u> control	<div><f11> SPC M-y <f2></div> <div><f12> <f2></div>	(<u>pel-customize-pel</u> &optional OTHER-WINDOW)	Customize <u>PEL</u> <u>YAML</u> support. <ul style="list-style-type: none">If <u>OTHER-WINDOW</u> is non-nil (use <u>C-u</u>) , display in other window.																											
<u>Z</u> <u>Customize</u> Emacs <u>YAML</u> control	<div><f11> SPC M-y <f3></div> <div><f12> <f3></div>	(<u>pel-customize-library</u> &optional OTHER-WINDOW)	Customize Emacs <u>YAML</u> support groups: <u>yaml</u> , <u>fly check</u> , <u>indent-tools</u> and <u>smartparens</u> <ul style="list-style-type: none">If <u>OTHER-WINDOW</u> is non-nil (use <u>C-u</u>) , display in other window.																											
<u>Flycheck</u> See also: <u>Z</u> <u>SyntaxCheck</u>	<u>Flycheck</u> is a minor mode for on-the-fly syntax checking. <div> The <u>flycheck</u> external package is activated by <u>PEL</u> when the <u>pel-use-flycheck</u> user-option is turned on or another activated <u>PEL</u> user-option requires it.</div> <div> Aside from the following 2 key bindings that <u>PEL</u> provides to toggle the <u>flycheck</u> mode, <u>flycheck</u> key prefix is <u>C-c</u> ! as set by its <u>flycheck-keymap-prefix</u> user-option. You can change it for a different key prefix.</div>																													
Toggle flycheck mode for current buffer	<f11> ! !	(<u>flycheck-mode</u> &optional ARG)	Toggle flycheck minor-mode for the current buffer.																											
Toggle flycheck mode for all buffers	<f11> ! M-!	(<u>global-flycheck-mode</u> &optional ARG)	Toggle Flycheck mode in all buffers. <ul style="list-style-type: none">Flycheck mode is enabled in all buffers where ‘flycheck-mode-on-safe’ would do it.																											
<ul style="list-style-type: none"><u>Flycheck</u> buffer/file																														
Syntax Check current buffer	C-c ! c	(<u>flycheck-buffer</u>)	Start checking syntax in the current buffer. <ul style="list-style-type: none">Get a syntax checker for the current buffer with ‘flycheck-get-checker-for-buffer’, and start it.																											
Check syntax of current file	C-c ! C-c	(<u>flycheck-compile</u> CHECKER)	Run CHECKER via ‘compile’. <ul style="list-style-type: none">CHECKER must be a valid syntax checker. Interactively, prompt for a syntax checker to run.Instead of highlighting errors in the buffer, this command pops up a separate buffer with the entire output of the syntax checker tool, just like ‘compile’.																											
Highlight current column <u>Toggle Vline Mode</u> See also: <ul style="list-style-type: none"><u>Z</u> <u>Highlight</u><u>Z</u> <u>Hide/Show</u>	<ul style="list-style-type: none">• <f11> h • <f11> 9	(<u>vline-mode</u> &optional ARG)	Toggle the display of a vertical line spanning the entire window at the cursor column. <div> Requires: <u>vline.el</u> <u>PEL</u> activates it when <u>pel-use-vline</u> user option is <u>t</u>.</div>																											
Indented Text Folding Toggle hiding lines more indented than current line See also: <u>Z</u> <u>Hide/Show</u>	The following command folds (hide or show) all lines that are indented more than the current line. <ul style="list-style-type: none">You can also use the <u>f</u> key inside the indent-tools Hydra, shown below, to fold indented sections.																													
Toggle hiding lines more indented than current line See also: <u>Z</u> <u>Hide/Show</u>	<f11> H I	(<u>pel-toggle-hide-indent</u>)	Toggle hiding lines more indented than current line. <ul style="list-style-type: none">Affects the entire buffer. Not syntax sensitive. Can be used anywhere. <div> Do not modify the buffer while lines are hidden, it's allowed but its using selective display and you don't see what you change.</div>																											
<u>Indent-tools</u> The indent-tools hydra provide keys you can use to navigate across the indented <u>YAML</u> elements. See also: <u>Z</u> <u>Hide/Show</u>	<ul style="list-style-type: none">• <f11> <tab> <f7>• <f7> <tab>• C-c >	(<u>indent-tools-hydra/body</u>)	Activate the body in the "indent-tools-hydra" hydra. <div> With <u>PEL</u>, this key binding is only available when:<ul style="list-style-type: none">globally, when <u>pel-indent-tools-key-bound</u> is set to <u>globally</u>,in python-mode only when <u>pel-indent-tools-key-bound</u> is set to <u>python</u>.The actual key is selected by indent-tools <u>indent-tools-keymap-prefix</u> user-option, the default is <u>C-c ></u></div>																											
The heads for the associated hydra are: <div>>: ‘indent-tools-indent’, <: ‘indent-tools-demote’, B: ‘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</div>		<table><tr><th>Indent</th><th>Navigation</th><th>Actions</th></tr><tr><td colspan="3">+-----+-----+-----</td></tr><tr><td>> indent</td><td>j v</td><td>K kill</td></tr><tr><td>< de-indent</td><td>k ^</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> <div> The <u>f</u> key toggles the element folding. Press once to hide the sub-tree, press-again to display it back.</div>		Indent	Navigation	Actions	+-----+-----+-----			> indent	j v	K kill	< de-indent	k ^	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
Indent	Navigation	Actions																												
+-----+-----+-----																														
> indent	j v	K kill																												
< de-indent	k ^	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																												

Operation	Keystroke	Function	Note
Smartparens Mode • Smartparens manual See also: Smartparens	Simplify insertion of matching pairs with the smartparens minor mode. PEL binds a set of keys, described below, to toggle activation of that mode.  This uses the smartparens external package.  PEL activates it when pel-use-smartparens is set to t . • Mode line lighter: • <code>smartparens-mode:</code> SP • <code>smartparens-strict-mode:</code> SP/s		
Help on smartparens	<code><f11> (?</code>	(sp-cheat-sheet &optional ARG)	Generate a cheat sheet of all the smartparens interactive functions. Shows inside Emacs buffer. • Without a prefix argument, print only the short documentation and examples. • With non-nil prefix argument ARG, show the full documentation for each function. • You can follow the links to the function or variable help page. • To get back to the full list, use M-x help-go-back. • You can use ‘beginning-of-defun’ and ‘end-of-defun’ to jump to the previous/next entry. • Examples are fontified using the ‘font-lock-string-face’ for better orientation.
Describe user system	<code><f11> (M-?</code>	(sp-describe-system STARTERKIT)	Describe user’s system. Prompt for starter kit: Evil, Spacemacs, Vanilla. • The output of this function can be used in bug reports.
Toggle smartparens mode	<code><f11> ((</code>	(smartparens-mode &optional ARG)	Toggle smartparens mode.
Toggle smartparens-strict mode	<code><f11> ()</code>	(smartparens-strict-mode &optional ARG)	Toggle the strict smartparens mode. • 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). • When strict mode is active, this is indicated with “/s” after the smartparens indicator in the mode list
Toggle smartparens mode	<code><f11> (M-(</code>	(smartparens-global-mode &optional ARG)	Toggle Smartparens mode in all buffers. • With prefix ARG, enable Smartparens-Global mode if ARG is positive; otherwise, disable it. • Smartparens mode is enabled in all buffers where ‘turn-on-smartparens-mode’ would do it.
Toggle smartparens-strict mode	<code><f11> (M-)</code>	(smartparens-global-strict-mode &optional ARG)	Toggle Smartparens-Strict mode in all buffers. • With prefix ARG, enable Smartparens-Global-Strict mode if ARG is positive; otherwise, disable it. • Smartparens-Strict mode is enabled in all buffers where ‘turn-on-smartparens-strict-mode’ would do it.
Smart-shift See also: Indentation	The smart-shift external package simplifies shifting a complete line or region of lines right or left but also up or down. • It is implemented as a minor or global minor mode that must be enabled first. You can identify the <code>smart-shift-mode</code> inside one of the <code>pel-<mode>-activates-minor-modes</code> user-options to activate it automatically. You can also use the commands manually or through the key bindings provided by PEL to activate the <code>smart-shift-mode</code> in the current buffer or globally for all buffers. • PEL controls it through customization user-options:  The smart-shift external package  PEL activates it when the <code>pel-use-smart-shift</code> user-option is turned on (set to <code>t</code>).  PEL also provides the pel-smart-shift-keybinding 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 <code>C-c</code> as a key prefix. With PEL you can also use a control key for the cursor or change the prefix key to use the <code><f9></code> 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.		
Toggle smart-shift mode in current buffer	<code><f11> <tab> s</code>	(smart-shift-mode &optional ARG)	Activate/de-activate the smart-shift mode in the current buffer. • Activate the line-shift key bindings listed below, in the current buffer. • With PEL, the actual key binding selected for the line shift commands depend on the value of the pel-smart-shift-keybinding user-option.
Toggle smart-shift mode globally	<code><f11> <tab> S</code>	(global-smart-shift-mode &optional ARG)	• Toggle Smart-Shift mode in all buffers. • With prefix ARG, enable Global Smart-Shift mode if ARG is positive; otherwise, disable it. • Smart-Shift mode is enabled in all buffers where ‘smart-shift-mode-on’ would do it.
Shift line or region right	• <code>C-c <right></code> • <code>C-c C-<right></code> • <code><f9> <right></code>	(smart-shift-right &optional ARG)	Shift the line or region to the ARG times to the right. 👉 With PEL one of the extra key bindings can be enabled via the pel-smart-shift-keybinding 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	• <code>C-c <left></code> • <code>C-c C-<left></code> • <code><f9> <left></code>	(smart-shift-left &optional ARG)	Shift the line or region to the ARG times to the left. 👉 With PEL one of the extra key bindings can be enabled via the pel-smart-shift-keybinding 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	• <code>C-c <up></code> • <code>C-c C-<up></code> • <code><f9> <up></code>	(smart-shift-up &optional ARG)	Shift the line or region to the ARG times to the upwards. 👉 With PEL one of the extra key bindings can be enabled via the pel-smart-shift-keybinding 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	• <code>C-c <down></code> • <code>C-c C-<down></code> • <code><f9> <down></code>	(smart-shift-down &optional ARG)	Shift the line or region to the ARG times to the downwards 👉 With PEL one of the extra key bindings can be enabled via the pel-smart-shift-keybinding 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
YAML	
YAML @ Wikipedia	Overview, syntax, criticisms
YAML official home page	Links to YAML specification, links to various resources and projects. • YAML 1.2 Specs • YAML 1.1 Specs • YAML 1.0 Specs

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