rst-mode: reStructuredText Mode

	131	-mode. restruc	Turcured text Wode
<u>Description</u>	<u>Keystroke</u>	Function	<u>Note</u>
reStructuredText Emacs Support for reSTructuredText Basic Intro to rst reStructuredText markup reStructuredText Directives Quick reference to rst rst-cheatsheet (pdf) Sphinx & rst syntax guide	This page describes Emacs support for reStructuredText (abbreviated sometimes as 'rst' and sometimes as 'reST'). • The reSructuredText files are supported by Emacs rst-mode from rst.el which is available in standard Emacs distribution. • Supported file extensions: .rst, .rest, .stxt and .rst.txt. The .rst.txt extension allows rendering by tools supporting .txt files. • To activate it under PEL, you must set the PEL pel-use-rst-mode customization variable to t. • pel-rst-tab-width: The width of a tab used for reStructuredText files. Defaults to 2. • This concept differs from indentation: you can have an indentation of 3 and tab width of 8: M-i will move point to columns that are multiple of 8 <tab></tab> • multiple of 8 <tab></tab> • Indentation.		
See also: Speedbar	 pel-rst-use-tabs: whether hard tabs are used for indentation. Defaults to nil (use space characters for all indentation). Speedbar Support: PEL activates Speedbar support for reStructuredText when the pel-use-speedbar user-option is turned on (set to t). Use the Speedbar to see the sections of the reStructuredText document and navigate to them. 		
Open this PDF file. See also: <u>Nelp/Info</u>	<f11> SPC M-r <f1> <f12> <f1></f1></f12></f1></f11>	(pel-help-pdf &optional OPEN-WEB-PAGE)	Open the <u>M reStructuredText</u> local PDF. If the prefix argument (like C-u or M) is used, then it opens the remote GitHub hosted raw PDF instead. If the pel-flip-help-pdf-arg user-option is set it's the other way around.
∑ Customize PEL reStructuredText support	<f11> SPC M-r <f2> <f12> <f2></f2></f12></f2></f11>	(pel-customize-pel &optional OTHER-WINDOW)	Customize PEL reStructuredText support. • If OTHER-WINDOW is non-nil (use C-u), display in another window.
<u>∑ Customize</u> Emacs reStructuredText support	<f11> SPC M-r <f3> <f12> <f3></f3></f12></f3></f11>	(pel-customize-library &optional OTHER-WINDOW)	Customize Emacs reStructuredText support. • If OTHER-WINDOW is non-nil (use C-u), display in another window.
<u>rst-mode</u>	Emacs provides the <u>rst-mode</u> The following file extensions		ode: .rst, .rest. PEL adds the .stxt extension.
Activate reStructuredText mode	M-x rst-mode	(rst-mode)	Toggle the rst-mode used to edit reStructuredText markup. • Automatically invoked when visiting .rst, .rest files (and .stxt files with PEL).
Get version of rst-mode	C-h v rst-version		Shows the content of the variable rst-version. • Only works once the rst-mode is loaded.
Syntax Control • superword-mode	reStructuredText files often benefit from the <u>superword-mode</u> and treating underscores as part of words. This helps searching for symbols and opening file that have names that include underscore characters. Force activation of <u>superword-mode</u> for reStructuredText by adding the <u>superword-mode</u> to the <u>pel-rst-activates-minor-modes</u> user-option.		
Control syntax of the underscore character	<f12> _</f12>	(pel-rst-set-underscore- syntax &optional ACTION)	Control syntax of underscore to punctuation or symbol when the superword-mode is active.
Requires superword-mode on.	If superword-mode is off the function issues a user error. Otherwise the function operates according to the value of the optional argument: If the argument is not specified or nil, the function toggles the syntax of the underscore character between punctuation (the default) and symbol. If the argument is positive, it sets the syntax of underscore to symbol. If the argument is 0 or negative it sets the syntax of the underscore back to punctuation. To force use of symbol syntax, add pel-rst-set-underscore-syntax to the pel-rst-activates-minor-modes user-option after superword-mode.		
Editing Content	The following generic comma	ands are useful when editing reS	tructuredText content.
• Text filling See also: • ∑ Filling/Justification	Although text filling will be handled for the generated rendering, you may decide to fill the reStructuredText file itself, after all you're using a markup that's made to allow reading the original text. • You can turn the auto fill mode on and identify the fill column. Force the auto-fill-mode when a reStructuredText file is visited by adding the auto-fill-mode to the pel-rst-activates-minor-modes user-option.		
Toggle auto-fill mode	• <f11> t f a • <f11> RET</f11></f11>	(auto-fill-mode &optional ARG)	Toggle automatic line breaking (Auto Fill mode). With a prefix argument, enable Auto Fill mode if the prefix argument is positive, and disable it otherwise. When Auto Fill mode is enabled, inserting a space at a column beyond 'current-fill-column' automatically breaks the line at a previous space.
Set Fill Column	• C-x f • <f11> t f c</f11>	(set-fill-column ARG)	When no prefix value: prompts for column unless a prefix argument was used. If with C-u prefix: use current column. If with prefix value: use that value.
Fill current paragraph	• M-q • <f11> t f p</f11>	(fill-paragraph &optional JUSTIFY REGION)	To justify as well: C-u M-q • In refill mode this is done automatically. In auto fill mode the filling is done at the end of the line.
Align a set of lines on some text	<f11> t w a</f11>	(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. Interactively, this function prompts for the regular expression REGEXP to align with.
	The PEL package creates to	the ar alias for align-regexp , so	le, to align assignment of variables over the equal sign use = as the regexp. it's also possible to invoke it with M-x ar RET nk lines and then issue the command, specifying http as the regexp to line them vertically.
Text Emphasis	The PEL commands emphasi	ize the current word or marked r	egion, then move point to the character right after the emphasized text.
Bold	<f12> b <f11> SPC M-r b</f11></f12>	(pel-rst-bold)	Mark current word or marked region bold. • Leave point after to the next character.
Italic	<f11> SPC M-r i</f11>	(pel-rst-italic)	Mark current word or marked region italic. Leave point after to the next character.
Literal	<f11> SPC M-r i <f12> 1</f12></f11>	(pel-rst-literal)	Mark current word or marked region with the literal markup. • Leave point after to the next character.
Interpreted	<f11> SPC M-r 1 <f12> `</f12></f11>	(pel-rst-interpreted)	Mark current word or marked region with the interpreted markup. • Leave point after to the next character.
	<f11> SPC M-r `</f11>		200.0 point and to the floot origination.
Indent list item See also: ∑ Indentation	<tab></tab>	(indent-for-tab-command &optional ARG)	When point is anywhere on a list item line (a line that starts with one if the supported bullet characters), this cycles the indentation through the possible indentations of the item.
Comment See also: <u>▼ Comments</u>	M-;	(comment-dwim ARG)	Comment line or region. ! Uncommenting does not work.
Itemize all previous lines same indention level	• <f12> M • <m-f12> M</m-f12></f12>	(pel-itemize-lines &optional ITEM-PREFIX-STRING)	Prepend each of the previous lines with a ITEM-PREFIX-STRING that is "- " by default. When writing a list of items, instead of manually typing the "- " prefix on each line, type each line without them and then use this command to itemize each of the lines above the current one. It indents all lines above the current line that are at the same indentation level as the current position.

<u>Description</u>	<u>Keystroke</u>	Function	<u>Note</u>	
File's Table of Content			xt tools automatically generate a table of contents for your file.	
	 You can also insert an explicit table of content with the rst-toc-insert command. There are several ways to view the files sections: 			
See also: <u>∑ Speedbar</u>	 with C-c C-t to invoke the rst-doc command: it opens a *table of Content* buffer, moves point inside it, move to the section title, hit RET to select that section inside the original reStructuredText buffer. using the Speedbar to open a buffer that lists the sections. See <u>Speedbar</u>. 			
Insert a table content at point	C-c C-t TAB	(rst-toc-insert &optional MAX-LEVEL)	Insert the table of contents of the current section at the current column. • By default the top level is ignored if there is only one, because we assume that the	
		IVIAX-LEVEL)	document will have a single title.	
			A numeric prefix argument MAX-LEVEL overrides 'rst-toc-insert-max-level'. Text in the line beyond column is deleted.	
			Sou may want to use the contents markup directive instead.	
Display table of content	C-c C-t C-t	(rst-doc)	Display a table of contents for current buffer inside the *Table of Contents* buffer. • Displays all section titles found in the current buffer in a hierarchical list.	
Navigate to specific section			 Select the section of interest in the *Table of Contents* buffer by navigating to it, then hit RET on that section to move back to the section in the original reStructuredTex document and close the Table of Contents* buffer window. 	
Moving across sections	You can also use the followin	g commands to move to the nex	t or previous section.	
Move to previous section title	• <f12> p</f12>	(rst-backward-section OFFSET)	Jump backward OFFSET section titles ending up at the start of the title line. • OFFSET defaults to 1 and may be negative to move backward. • An OFFSET of 0 does not move unless point is inside a title.	
	• <f12> <up></up></f12>		Go to end or beginning of buffer if no more section titles in the desired direction.	
	• <f11> SPC M-r p • <f11> SPC M-r <up></up></f11></f11>			
Move to next section title	• C-M-e	(rst-forward-section OFFSET)	Jump forward OFFSET section titles ending up at the start of the title line. • OFFSET defaults to 1 and may be negative to move backward.	
	• <f12> n • <f12> <down></down></f12></f12>	OTTOLI)	An OFFSET of 0 does not move unless point is inside a title.	
	• <f11> SPC M-r n</f11>		 Go to end or beginning of buffer if no more section titles in the desired direction. 	
	• <f11> SPC M-r <down></down></f11>			
Mark complete current section	С-М-Һ	(rst-mark-section &optional COUNT ALLOW-EXTEND)	Select COUNT sections around point. Mark following sections for positive COUNT or preceding sections for negative COUNT.	
Tempo skeletons for	PEL provides support for flex	 ible text template insertion throu	gh the Emacs built-in tempo skeleton mechanism.	
reStructuredText		·	oported major modes, using the same key prefix sequence for each mode: <f12> s (such as file header block) as much as possible.</f12>	
See also: <u>▼ Inserting Text</u>			about tempo skeleton and yasnippet template-based text insertion).	
Insert a file header	<f12> <f12> h</f12></f12>	(pel-rst-large-header)	Insert a large header includes all normal header fields plus separators.	
			 Prompts for title and insert title, automatically updated timestamp, attributes for home page and license, markup for table of contents using the tempo skeleton mechanism. 	
			 Automatically activates the PEL tempo skeleton mode so you can move to the target points where extra text must be entered to complete the template. 	
Toggle pel-tempo-mode	<f12> <f12> SPC</f12></f12>	(pel-tempo-mode &optional ARG)	Toggle PEL tempo mode on/off. PEL tempo mode activates C-c . and C-c , as well as to C-c C and C-c C-,	
		7	key bindings to navigate across tempo mark hot-spots. When pel-tempo-mode is active the pel-tempo-mode lighter (‡) is shown on the status bar. The second set are	
			only available when Emacs runs in graphics mode.	
			When a skeleton is inserted via the execution of one of the pel-rst commands, the pel-tempo-mode is automatically activated.	
Jump to next tempo mark	• C-c M-f	(tempo-forward-mark)	Jump to the next mark in 'tempo-back-mark-list': the location where code must be	
	• C-c . • C-c C		updated inside the inserted skeleton. • These key key bindings are only available when pel-tempo-mode is active.	
Jump to previous tempo	• C-c M-b	(tempo-backward-mark)	Jump to the previous mark in 'tempo-back-mark-list': the location where code must be	
mark	• C-c , • C-c C-,		updated inside the inserted skeleton. • These key binding are only available when pel-tempo-mode is active.	
Tempo Template Tag Insertion	<f12> <f12> <f12></f12></f12></f12>	(tempo-complete-tag &optional SILENT)	Look for a tag and expand it.	
	Instead of using the <f12> <f12> key bindings above, you can type the template name (shown in the title column like "if", "case", etc)</f12></f12>			
	completely or partially and then hit <f12> <f12> <f12>. A completion buffer opens up if the template name is incomplete (or empty in which case the buffer lists all available template names). Select the template name and hit RET. Emacs expands the template. • All the tags in the tag lists in 'tempo-local-tags' (this includes 'tempo-tags') are searched for a match for the text before the point. The way the string to match for is determined can be altered with the variable 'tempo-match-finder'. If 'tempo-match-finder' returns nil, then the results are the same as no match at all. • If a single match is found, the corresponding template is expanded in place of the matching string.</f12></f12></f12>			
	If a partial completion or no	o match at all is found, and SILE	NT is non-nil, the function will give a signal.	
	1 '	 If a partial completion is found and 'tempo-show-completion-buffer' is non-nil, a buffer containing possible completions is displayed. Since only one template is available in rst-mode, the usefulness of this command is limited for reStructuredText. 		
Select Section Title Adornment Styles	The underlying character used for section line adornment is customizable. The number of available levels and whether the line is indented, has a line over and under the title line is selected by the adornment style. PEL supports 3 styles. The following commands can be used to select a style.			
Select default adornment style	<f12> A d</f12>	(pel-rst-adorn-default)	Set the default section adornment style. This is Emacs rst-mode default: a title with 7 levels.	
	<f11> SPC M-r A d</f11>			
Select Sphinx-Python adornment style	<f12> A S</f12>	(pel-rst-adorn-Sphinx- Python)	Set the Sphinx-Python section adornment style. This is what Sphinx supports: 6 levels:	
	<f11> SPC M-r A S</f11>		parts, chapters.	
			sections, subsections,	
			subsections, subsubsections, paragraphs.	
Select CRiSPer adornment	<f12> A C</f12>	(pel-rst-adorn-CRiSPer)	Set the CRiSPer section adornment style.	
style	<f11> SPC M-r A C</f11>		A title level with another 12 levels. Use <f12> + to create those levels.</f12>	

<u>Description</u>	<u>Keystroke</u>	Function	<u>Note</u>	
Section Title level			section adornment of the current line. ately sometimes fails when market is used and not expected by its code.	
adornmentcommands that insert section titles	 This command tries to infer the level required and unfortunately sometimes fails when market is used and not expected by its code. PEL provides a set of very simple commands that use multiple key bindings to adorn the current line to a fixed section level: title level and up to 10 other levels, from 1 to 9 and then 0 for 10. It also provides commands to adorn a line to the same level as the previous section or a lower or higher level. And then to increase or decrease. 			
	 It also provides commands to adorn a line to the same level as the previous section or a lower or higher level. And then to increase or decrease the section level of the adornment of the current line. PEL provides 3 style of section adornments: default, Sphinx-Python and CRiSPer, which can be selected with commands. 			
	<u>w</u> PEL remembers the preferred style inside the customizable variable: pel-rst-adornment-style .			
	 The rest.el provides the rst-preferred-adornment user option to select the adornment characters for the various sections. PEL code selects the value according to the adornment style you select. See section "Select Adornment Styles" above. 			
Adjust section level	• C-=	(rst-adjust PFXARG)	Auto-adjust the adornment around point.	
	• C-c C-= • C-c C-a C-a		 Adjust/rotate the section adornment for the section title around point or promote/ demote the adornments inside the region, depending on whether the region is active. This function is meant to be invoked possibly multiple times, and can vary its behavior with a positive PFXARG (toggle style), or with a negative PFXARG (alternate behavior). This function is a bit of a swiss knife. It is meant to adjust the adornments of a section title in reStructuredText. It tries to deal with all the possible cases gracefully and to do "the right thing" in all cases. 	
Adorn line at title level	<f12> t</f12>	(pel-rst-adorn-title)	Adorn current line with level-0 (title) reStructuredText section adornment. If done at the top of the file, the first adorn line is placed on the first line of the file, a mark is left at the end of the title line and point is moved 2 lines below. To return to the end of the title line, type M-` or <f6><f6>.</f6></f6>	
	<f11> SPC M-r t</f11>			
Adorn to specific level	• <f12> 1</f12>	• (pel-rst-adorn-1)	Adorn current line with level [1 to 10] reStructuredText section adornment.	
From level 1 to level 10	 • <f12> 9</f12>	• (pel-rst-adorn-2) • (pel-rst-adorn-3)	►The <f11> SPC M-r 1 to <f11> SPC M-r 0 key sequences can be used</f11></f11>	
	• <f12> 0</f12>	• (pel-rst-adorn-4) • (pel-rst-adorn-5)	inside any buffer. The <f12> keys can only be used in inside the buffers in rst-mode.</f12>	
	• <f11> SPC M-r 1</f11>	• (pel-rst-adorn-6)		
	• <f11> SPC M-r 9 • <f11> SPC M-r 0</f11></f11>	(pel-rst-adorn-7)(pel-rst-adorn-8)(pel-rst-adorn-9)(pel-rst-adorn-0)		
Adorn current line: same	• <f12> =</f12>	(pel-rst-adorn-same-level)	Adorn current line with the same level as the previous section.	
section level as previous section	<f11> SPC M-r =</f11>	,	If the line is already adorned, update the adornment: adjust to previous section level.	
Adorn to higher section level	<f12> +</f12>	(pel-rst-adorn-increase-	Adorn current line at a higher-level that current if already adorned.	
·	<f11> SPC M-r +</f11>	level)	If the line is not already adorned, adorn it with a level higher than previous section.	
Adorn to lower section level	<f12> -</f12>	(pel-rst-adorn-decrease-	Adorn current line at a lower-level than current if already adorned.	
	<f11> SPC M-r -</f11>	level)	If the line not already adorned, adorn it with a level lower than previous section.	
Refresh current line	<f12> r</f12>	(pel-rst-adorn-refresh)	Refresh the adornment of the current line, adjusting the underlining to the current length	
adornment	<f11> SPC M-r r</f11>		of the line. • This can be useful when changing the text on the line.	
Creating and Using Hyperlinks	The following 3 PEL commands help write hyperlink of various forms: • the embedded form where the URL is stored inside the text between angle brackets and • the full named format where the link is located elsewhere in the file on its own line. When editing a buffer using the rst-mode, type the <f12> • keystroke to create a hyperlink. • It uses the selected region if one is highlighted or the word at point otherwise as the title for the link and creates the link entry on a line identified by a dedicated bookmark: that bookmark is created by the <f12> s keystroke. That helps identify an area inside the file where the next (or several) hyperlinks will be located. • With PEL, the <f12> key prefix is mode sensitive. If you want to use the same commands inside another mode, you can use the longer key chord that uses the <f11> SPC M-r prefix (assuming that pel-use-rst-mode user-option is set).</f11></f12></f12></f12>			
Set location of hyperlinks	<f12> s</f12>	(pel-rst-set-ref-bookmark)	Set the reference bookmark for the currently edited file at point.	
	<f11> SPC M-r s</f11>		 Used to identify the location where the next invocation of M-x pel-rst-mekelink inserts fully expanded links. Ensures the bookmark is at the beginning of an empty line which is followed by another empty line, by inserting 2 lines and placing the point at the beginning of the first of the 2 lines. 	
Add an hyperlink for text at	<f12> .</f12>	(pel-rst-makelink &optional	Create a reStructuredText hyperlink prefix for the word at point or region's text. If region active, use text of the region for the link, otherwise use the word at point. If an argument (which can be a C-u) is specified, use the embedded URI format. If no argument is specified, use the named hyperlink format: if the region is a single word, just append an underscore to make the link if the region is several words, surround it with the "" and the "" strings. The named link is placed in the location of bookmark named "RST" if it exists and points to same file, otherwise the link is placed at the beginning of the next empty line The cursor is placed where the URL is to be written. Command pushes the mark on mark ring, type M- or <f6><f6> to move back to previous location.</f6></f6>	
point	<f11> SPC M-r .</f11>	ARG)		
Go to hyperlink location	<f12> g</f12>	(pel-rst-goto-ref-bookmark)	Move point to the reference bookmark. Useful to see where the bookmark for storing the hyperlink are currently located or add empty lines for future references.	
	<f11> SPC M-r g</f11>		 Command pushes the mark on mark ring, type M-\^ or <f6><f6> to move back to previous location.</f6></f6> 	
Activating URLs to browse and open files See also: •	Once the mode is active t C-c RET or the mous If the URL is an ema If the URL is a web c C-c C-n: move point C-c C-p: move point	he following key sequences are a se to click on the button. il address a buffer to write an em or FTP address the system brows t to the end of the next URL in th t to to the previous URL in the bu the file identified by the URL into	er is invoked to open the address. e buffer. Iffer. o a local temporary file and visit the file. See (pel-open-url-at-point) above.	
		to address Mostly control the	regey for LIRL and the face used	
Toggle goto-address-mode	Customization group: go	(goto-address-mode &optional ARG)	Minor mode to buttonize URLs and e-mail addresses in the current buffer. With a prefix argument ARG, enable the mode if ARG is positive, and disable it	
Toggle goto-address-mode Toggle goto-addrress-prog-	Customization group: go	(goto-address-mode	Minor mode to buttonize URLs and e-mail addresses in the current buffer.	

<u>Description</u>	<u>Keystroke</u>	Function	Note
Open the URL (email or web page)	C-c RET	(goto-address-at-point &optional EVENT)	Open the URL at point: If URL is a web page: open it in a browser If URL is a mail address: Send mail to address at point: Find e-mail address around or before point. Then search backwards to beginning of line for the start of an e-mail address. If no email address is found there, then load the URL at or before point.
Move to end of next URL in buffer See also: Navigation	C-c C-n <f6> C-n</f6>	(pel-goto-next-url)	Move point forward to the end of the next URL located in the current buffer. • The global <f6> C-n key binding activates the goto-address-mode if it is not already active.</f6>
Move to beginning of	C-c C-p	(pel-goto-previous-url)	Move point backward to the beginning of the previous URL located in the current buffer.
previous URL in buffer See also: Navigation	<f11> C-p</f11>	(per gote protions and	The global <f6> C-p key binding activates the goto-address-mode if it is not already active.</f6>
Copy URL at point in temporary file and visit the file See also: File mngt	<f11> f M-u</f11>	(pel-open-url-at-point)	Copy the URL at point to a local temporary file and visit that file. • A The download copy of the file does not have the same name and may not open with the proper mode because it won't have an extension. The HTML formatted files will be recognized by Emacs but most of the files won't be. • Save the file somewhere else using the C-x C-w key sequence and identify the proper extension to activate the required major mode.
	C-c C-f		Solution Solution
Open file or web-page whose name or markup link is at point	• C-^ • <f11> f . • <m-f11> M-f M</m-f11></f11>	(pel-open-at-point &optional N)	Open the file, library or the URL, named at point, with potential line and column #s. With PEL, the <u>6y</u> key-chord is available if pel-use-key-chord is non-nil.
	• <u>6y</u>		Command prefixes are supported with the key-chord. See ★ Key-Chords Key-Chords Command prefixes are supported with the key-chord. See ★ Key-Chords Command prefixes are supported with the key-chord. See ★ Key-Chords Command prefixes are supported with the key-chord. See ★ Key-Chords Command prefixes are supported with the key-chord. See ★ Key-Chords Command prefixes are supported with the key-chord. See ★ Key-Chords Command prefixes are supported with the key-chord. See ★ Key-Chords Command prefixes are supported with the key-chord. See ★ Key-Chords Command prefixes are supported with the key-chord. See ★ Key-Chords Command prefixes are supported with the key-chords. See Texture Command prefixes are supported with the key-chords. See Texture Command prefixes are supported with the key-chords. See Texture Command prefixes are supported with the key-chords. Texture Command prefixes are supported with the key-chords. See Texture Command prefixes are supported with the key-chords. See Texture Command prefixes are supported with the key-chords. See Texture Command prefixes are supported with the key-chords. See Texture Command prefixes are supported with the key-chords. See Texture Command prefixes are supported with the key-chords. See Texture Command prefixes are supported with the key-chords. See Texture Command prefixes are supported with the key-chords. See Texture Command prefixes are supported with the key-chords. See Texture Command prefixes are supported with the key-chords. See Texture Command prefixes are supported with the key-chords. See Texture Command prefixes are supported with the key-chords. See Texture Command prefixes are supported with the key-chords. See
**	This command extracts the fi below.	le name to search from text at po	oint. The file name is either surrounded by white space characters or the delimiters listed
See also: •	 If embedded space(s) are allowed in the filename, then point must be located at the first of the 2 delimiter characters. These delimiter character can be any of the following: tab, newline and: "`' ()[]{}<> ''"" 「」() 〈〉 《〉 【】 []] «» ‹ › ◇ O (]) ·。 If embedded space in the file name is not allowed, then the file name must also be enclosed in the above delimiters, the space acts as an extra delimiter, and point can be positioned anywhere between the delimiters. If the string identifies a URL, the function opens the page in the systems' default browser. The file name extracted from the file may include glob characters. A relative file name uses the visited file's parent directory or the buffer's current working directory by default. You can change this behaviour for each buffer by executing the pet-set-open-at-point-dir command (<f11> f;) in the buffer.</f11> Otherwise the command attempts to open the file name with the specified name. If that file does not exists it then proceed to search for it. If the file name is followed by line and column numbers the point is moved to that position. Inside a rst-mode buffer, when the point is over a restructuredText external hyperlink target reference, the command locates the reference and opens the file identified by the reference: If the reference is a Wb URL, it opens the identified web using the system browser. If the reference is a HTML file name that corresponds to the rendering of a local restructuredText file, it opens that reStructuredText file. If the reference is another type of file it opens that file. If the reference URL identifies a # URl fragment that identifies the name of a target file section, the command moves point to the section If the hyperlink refers to a title inside the document move point to that title instead of trying to open a file. When several file names are found, the command lists them and prompts using th		
in system's web browser or reStructuredText file identified by the external hyperlink target Jump to section identified by # URI fragment or jump to section in current reStructuredText file.			
Colcot prompt meaned "A"	 With ivy selected PEL will automatically set pel-use-ivy to t particular by mode will be installed automatically when you restart Emacs. Note that the command shows all files found by the specified search method, it does not only use the first one found. This allows you to detect potential duplication in header file names in large include paths. It Prompts for incomplete file names, allowing editing the find file (with completion), search for libraries files (type 1) according to current file type. 		
Select target window	 Select target window: Without argument: If file is already opened in a window, move point to that window and to the line column coordinates if specified following the file name at point. If no window holds that file, select the target window based on the number of editable windows in frame: if 1, split that window and use the new window, if 2: use the other window, if 3 or more, use the current window. With numeric argument N: N < 0: create a new window. 		
N>20 : open the directory See function docstring for	 (abs N) > 20: then open the directory instead of the file. Interpret the window position from the N value adjusted: N-20 (or N+20 if N is negative) N = 0: use the 'other' (the next) window. N = 1, 3, 7or above (excluding 8, 9 and 10): select the target window based on the number of editable windows in frame: if 1 window: split that window and use the new window, if 2 windows: use the other window, if 3 or more windows: use the current window. 		
more info.	open a direct N is 10: open the URI	in the system's browser (with	

rst-mode - References

Description & URL	Notes
Emacs Support for reStructuredText	
How to get the table of content with section numbers?	
reStructuredText	Main page for all reStructuredText documents.
reStructuredText markup Specifications	Formal markup specifications.
Sphinx Python Documentation Generator	
Sphinx - Documentation Contents	
Sphinx - Documentation - Sections	