Outline Mode

Description	Keystroke	Function	Note
Emacs Outline Mode See also: ∑ **Hide/Show** ☝ The outline minor mode is a **killer feature** that is under-utilized by several major modes. 🚧 PEL will eventually support the ability to define the variables and dynamically change the outlining hiding.	Emacs provides the Outline Mode that provides ability to manage outline of indented text. There are two different modes: • outline major mode • outline minor mode, which can be used in any other major mode, allowing outline operation on its content. • ☝ This mode is very useful to work on large code files in hierarchical manner. PEL provides key bindings to enter and leave the outline major and minor mode. • The outline **minor** mode is used more often, allowing outlining features inside source code files. Activate the outline minor mode inside any other major mode with **`<f11> M-l`**. PEL adds key bindings under the **`<f2>`** prefix key while the outline minor mode is active to help using it as well as some other convenience keys such as access to this help file via the **`<f1>`** key. ⌨ To support outline minor mode for a major mode the mode must identify values for the following variables: • **outline-regexp:** Regular expression to match the beginning of a heading. Any line whose beginning matches this regexp is considered to start a heading. Note that Outline mode only checks this regexp at the start of a line, so the regexp need not (and usually does not) start with '^'. The recommended way to set this is with a Local Variables: list in the file it applies to. See also 'outline-heading-end-regexp' • **outline-level :** Function of no args to compute a header's nesting level in an outline. It can assume point is at the beginning of a header line and that the match data reflects the 'outline-regexp'. • More variables are controlling it. Use the **pel-outline-print-vars** command to print their current buffer values. See below.		
Open this PDF file. See also: ∑ **Help/Info**	**`<f11> SPC M-l <f1>`** **`<f12> <f1>`**	**(pel-help-pdf** &optional OPEN-WEB-PAGE)	Open the ∑ **Outline** 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 Outline support	**`<f11> SPC M-l <f2>`** **`<f12> <f2>`**	**(pel-customize-pel** &optional OTHER-WINDOW)	Customize PEL Org Mode support: open pel-pkg-for-outline-mode group. • If OTHER-WINDOW is non-nil (use **`C-u`**), display in another window.
∑ **Customize** Emacs Outline support	**`<f11> SPC M-l <f3>`**	**(pel-customize-library** &optional OTHER-WINDOW)	Customize Org Mode external packages support: outlines • If OTHER-WINDOW is non-nil (use **`C-u`**), display in another window.
Enter Outline mode	**`<f11> M-L`**	**(outline-mode)**	Set major mode for editing outlines with selective display. • Headings are lines which start with asterisks: one for major headings, two for subheadings, etc. Lines not starting with asterisks are body lines. • Body text or subheadings under a heading can be made temporarily invisible, or visible again. Invisible lines are attached to the end of the heading, so they move with it, if the line is killed and yanked back. A heading with text hidden under it is marked with an ellipsis (...).
Enter Outline minor mode	**`<f11> M-l`**	**(outline-minor-mode** &optional ARG)	Toggle Outline minor mode. Headings are defined by outline variables. See ⌨ above. • With prefix ARG, enable Outline minor mode if positive, and disable it otherwise.
Key Bindings See ➤**Legend**	The key bindings shown below describe both the key bindings for the outline **major mode** and the outline **minor mode**. • The key bindings for the major mode are shown in a cell with light blue background. • The minor mode key bindings are shown in cells with light red background. • The **outline-minor-mode-prefix** user-option allows the selection of the minor mode prefix key, with is **`C-c @`** by default. • PEL adds key bindings under the **`<f2>`** prefix key while the outline minor mode is active.		
Open this PDF file. See also: ∑ **Help/Info**	**`<f2> <f1>`**	(pel-help-on-outline &optional OPEN-GITHUB-PAGE)	Open the ∑ **Outline** 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.
Print value of outline controlling variables See also: ∑ **Help/Info**	**`<f2> ?`** **`<f11> ? e o`**	**(pel-outline-print-vars)**	Print the current buffer specific values of outline controlling variables. Use this to learn possible how to control the outline minor mode.
Control Visibility			
Show all	**`C-c C-a`** • **`<f2> a`** • **`C-c @ C-a`**	**(outline-show-all)**	Show all of the text in the buffer.
Show entry	**`C-c C-e`** • **`<f2> e`** • **`C-c @ C-e`**	**(outline-show-entry)**	Show the body directly following this heading. • Show the heading too, if it is currently invisible.
Show children	**`C-c C-i`** • **`<f2> i`** • **`C-c @ C-i`**	**(outline-show-children** &optional LEVEL)	Show all direct subheadings of this heading. • Prefix arg LEVEL is how many levels below the current level should be shown. • Default is enough to cause the following heading to appear.
Show branches	**`C-c C-k`** • **`<f2> k`** • **`C-c @ C-k`**	**(outline-show-branches)**	Show all subheadings of this heading, but not their bodies.
Show subtree	**`C-c C-s`** • **`<f2> s`** • **`C-c @ C-s`**	**(outline-show-subtree)**	Show everything after this heading at deeper levels.
Hide body	**`C-c C-t`** • **`<f2> t`** • **`C-c @ C-t`**	**(outline-hide-body)**	Hide all body lines in buffer, leaving all headings visible. • Note that this does not hide the lines preceding the first heading line.
Hide entry	**`C-c C-c`** • **`<f2> c`** • **`C-c @ C-c`**	**(outline-hide-entry)**	Hide the body directly following this heading.
Hide subtree	**`C-c C-d`** • **`<f2> d`** • **`C-c @ C-d`**	**(outline-hide-subtree)**	Hide everything after this heading at deeper levels.
Hide leaves	**`C-c C-l`** • **`<f2> l`** • **`C-c @ C-l`**	**(outline-hide-leaves)**	Hide the body after this heading and at deeper levels.
Hide sub-levels	**`C-c C-q`** • **`<f2> q`** • **`C-c @ C-q`**	**(outline-hide-sublevels** LEVELS)	Hide everything but the top LEVELS levels of headers, in whole buffer. • This also unhides the top heading-less body, if any. • Interactively, the prefix argument supplies the value of LEVELS. • When invoked without a prefix argument, LEVELS defaults to the level of the current heading, or to 1 if the current line is not a heading.
Hide other	**`C-c C-o`** • **`<f2> o`** • **`C-c @ C-o`**	**(outline-hide-other)**	Hide everything except current body and parent and top-level headings. • This also unhides the top heading-less body, if any.

Description	Keystroke	Function	Note
Navigate			
Backward same level	`C-c C-b` • `<f2> b` • `C-c @ C-b`	(**outline-backward-same-level** ARG)	Move backward to the ARG'th subheading at same level as this one. • Stop at the first and last subheadings of a superior heading.
Forward same level	`C-c C-f` • `<f2> f` • `C-c @ C-f`	(**outline-forward-same-level** ARG)	Move forward to the ARG'th subheading at same level as this one. • Stop at the first and last subheadings of a superior heading.
Previous visible heading	`C-c C-p` • `<f2> p` • `C-c @ C-p`	(**outline-previous-visible-heading** ARG)	Move to the previous heading line. • With ARG, repeats or can move forward if negative. • A heading line is one that starts with a '*' (or that 'outline-regexp' matches)
Next visible heading	`C-c C-n` • `<f2> n` • `C-c @ C-n`	(**outline-next-visible-heading** ARG)	Move to the next visible heading line. • With ARG, repeats or can move backward if negative. • A heading line is one that starts with a '*' (or that 'outline-regexp' matches).
Up Heading	`C-c C-u` • `<f2> u` • `C-c @ C-u`	(**outline-up-heading** ARG &optional INVISIBLE-OK)	Move to the visible heading line of which the present line is a subheading. • With argument, move up ARG levels. • If INVISIBLE-OK is non-nil, also consider invisible lines.
Mark			
Mark subtree	`C-c @` • `<f2> .` • `C-c @ @`	(**outline-mark-subtree**)	Mark the current subtree in an outlined document. • This puts point at the start of the current subtree, and mark at the end.

Insert Heading	`C-c C-m`
	
	`<f2> RET`
		(**outline-insert-heading**)	Insert a new heading at same depth at point.

Re-organize			
Move subtree down	`C-c C-v` • `<f2> v` • `C-c @ C-v`	(**outline-move-subtree-down** &optional ARG)	Move the current subtree down past ARG headlines of the same level.

Move subtree up	`C-c C-^`
	
	`<f2> ^`
		(**outline-move-subtree-up** &optional ARG)	Move the current subtree up past ARG headlines of the same level.

Promote	• `C-c C-<`
	• `C-c [`
	
	`<f2> [`
	`C-c @ C-<`
		(**outline-promote** &optional WHICH)	Promote headings higher up the tree.
			• If 'transient-mark-mode' is on, and mark is active, promote headings in the region (from a Lisp program, pass 'region' for WHICH).
			• Otherwise: without prefix argument, promote current heading and all headings in the subtree (from a Lisp program, pass 'subtree' for WHICH); with prefix argument, promote just the current heading (from a Lisp program, pass nil for WHICH, or do not pass any argument).

Demote	• `C-c C->`
	• `C-c ]`
	
	`<f2> ]`
	`C-c @ C->`
		(**outline-demote** &optional WHICH)	Demote headings lower down the tree.
			• If 'transient-mark-mode' is on, and mark is active, demote headings in the region (from a Lisp program, pass 'region' for WHICH).
			• Otherwise: without prefix argument, demote current heading and all headings in the subtree (from a Lisp program, pass 'subtree' for WHICH); with prefix argument, demote just the current heading (from a Lisp program, pass nil for WHICH, or do not pass any argument).

Emacs Outline Reference

Emacs Manual- Outline Mode	
Outline Mode @ Emacs Wiki	
Articles about Using Outline Mode and Extending It	• Managing code with Outlines
Emacs: outline-minor-mode and imenu	Interesting video hosted on Youtube. Created by Protesilaos Stavrou