

Buffers

Operation	Keystroke	Function	Notes
Manage Buffers	The following commands support buffer management. There is also the <b>C-⌘F10</b> key that pops up the Emacs Menu dialog (see the Menus table).		
Toggle read-only status of buffer	<ul style="list-style-type: none"><li>C-x C-q</li><li>&lt;f11&gt; b r</li></ul>	(read-only-mode &optional ARG)	<p>When the buffer is in read-only mode the <u>mode line</u> shows ‘%%’ on the left side, in the ‘ch’ area of “cs:ch-fr buf pos line (major minor)”.</p> <p>The <u>manual</u> states: “For a read-only buffer, it shows ‘%%’ if the buffer is modified, and ‘%’ otherwise.”</p> <p>➡ <b>See also:</b> the <b>View Mode</b> activating commands toward the end of this table.</p> <ul style="list-style-type: none"><li>A buffer in View Mode cannot be modified.</li><li>The View Mode may be used to ensure that no modifications are made to a buffer (visiting a file or not).</li></ul>
Switch to next buffer	<ul style="list-style-type: none"><li>C-x &lt;right&gt;</li><li>C-x C-&lt;right&gt;</li><li>&lt;f11&gt; b n</li></ul>	(next-buffer)	Switch to the next buffer displayed in the current window.
Switch to previous buffer	<ul style="list-style-type: none"><li>C-x &lt;left&gt;</li><li>C-x C-&lt;left&gt;</li><li>&lt;f11&gt; b p</li></ul>	(previous-buffer)	Switch to the previous buffer displayed in the current window.
Show name of previous buffer in window	<f11> b P	(pel-show-window-previous-buffer)	Show the name of previous buffer used in the current window.
Switch to previous buffer in window	<f11> b l	(pel-switch-to-last-used—buffer)	Switch buffer in current window to the buffer previously seen in this window. Used twice returns to the same buffer.
Switch to buffer	C-x b	(switch-to-buffer BUFFER-OR-NAME &optional NORECORD FORCE-SAME-WINDOW)	<p>Switch window to display the previous, or another buffer (entered at prompt).</p> <p>👉 The invisible buffers have a name that start with a space. To see them type space and tab and a list of those buffers will appear before the list of visible buffers.</p>
List all buffers	C-x C-b	<ul style="list-style-type: none"><li>(list-buffers &amp;optional ARG)</li><li>(ibuffer &amp;optional OTHER-WINDOW-P NAME QUALIFIERS NOSELECT SHRINK FILTER-GROUPS FORMATS)</li></ul>	<p>Display a list of existing buffers in a buffer named “*Buffer List*”, the buffer displays information about all buffers and enters the <b>Buffer Menu Mode</b>. See the keystrokes for the Buffer Menu Mode below.</p> <p>➡ The PEL package the ‘ibuffer’ function instead, which provides more functionality, working like dired.</p>
Clone buffer	<f11> b c	(clone-buffer &optional NEWNAME DISPLAY-FLAG)	<p>Create and return a twin copy of the current buffer.</p> <p>Unlike an indirect buffer, the new buffer can be edited independently of the old one (if it is not read-only). NEWNAME is the name of the new buffer. It may be modified by adding or incrementing &lt;N&gt; at the end as necessary to create a unique buffer name.</p> <p>For example if buffer *Help* is opened it opens another one named *Help*&lt;2&gt; (or *Help*&lt;3&gt; if *Help*&lt;2&gt; already exists, etc…)</p>
Toggle buffer between normal and hexadecimal display	<f11> b x	(nhexl-mode &optional ARG)	<p>Toggle minor mode to edit files via hex-dump format.</p> <p>➡📦 Requires the <a href="#">nhexl-mode</a> package.</p> <p>➡🔌 PEL activates this when the <a href="#">pel-use-nhexl</a> customize variable is set to <b>t</b>.</p>
Activate Hex nibble editing mode	<f11> b X	(nhexl-nibble-edit-mode &optional ARG)	<p>Minor mode to edit the hex nibbles in ‘nhexl-mode’.</p> <p>⚠️ <b>Note:</b> only works after nhexl-mode has been activated once.</p> <p>➡📦 Requires the <a href="#">nhexl-mode</a> package.</p> <p>➡🔌 PEL activates this when the <a href="#">pel-use-nhexl</a> customize variable is set to <b>t</b>.</p>
Rename a buffer	<f11> b R	(rename-buffer NEWNAME &optional UNIQUE)	If UNIQUE argument is non-nil via C-u M-x rename-buffer, the name is auto generated to be unique.
Rename buffer - use unique name	<f11> b U	(rename-uniquely)	<p>Rename the current buffer by adding ‘&lt;number&gt;’ to the end. Use this if you want multiple “Buffer” or “Info” buffers for example.</p> <p>Example: <a href="#">StackExchange: How can I have multiple help buffer with different content</a></p>
Kill current buffer	<ul style="list-style-type: none"><li>&lt;f11&gt; b k</li><li>⌘-k</li><li>⌘-Ⓜ</li></ul>	(kill-current-buffer)	Kill (close) the current buffer. Does not prompt if there is no change in the buffer.
Kill buffer	C-x k	(kill-buffer &optional BUFFER-OR-NAME)	<p>Kill (close) the current buffer.</p> <ul style="list-style-type: none"><li>Always prompt to identify a buffer, current is identified. Press enter to kill the buffer.</li></ul>
Kill some buffer		(kill-some-buffers &optional LIST)	Kill some buffers. Asks the user whether to kill each one of them.
Delete all windows of a specific buffer		(delete-windows-on &optional BUFFER-OR-NAME FRAME)	Deletes all windows showing BUFFER-OR-NAME, by calling ‘delete-window’ on those windows.
Accumulating Text	Emacs provides the following commands to insert text in buffer from various sources.		
Append region to specified buffer	<f11> b M-a	(append-to-buffer BUFFER START END)	<p>Append to specified BUFFER the text of the region.</p> <ul style="list-style-type: none"><li>The text is inserted into that buffer before its point.</li><li>BUFFER can be a buffer or the name of a buffer; this function will create BUFFER if it doesn’t already exist.</li></ul>
Prepend region to specified buffer	<f11> b M-p	(prepend-to-buffer BUFFER START END)	<p>Prepend to specified BUFFER the text of the region.</p> <ul style="list-style-type: none"><li>The text is inserted into that buffer after its point.</li><li>BUFFER can be a buffer or the name of a buffer; this function will create BUFFER if it doesn’t already exist.</li></ul>
Copy region to specified buffer (replacing old content)	<f11> b C-c	(copy-to-buffer BUFFER START END)	<p>Copy to specified BUFFER the text of the region.</p> <ul style="list-style-type: none"><li>The text is inserted into that buffer, replacing existing text there.</li><li>BUFFER can be a buffer or the name of a buffer; this function will create BUFFER if it doesn’t already exist.</li></ul>
Insert content of specified buffer at point	<f11> b i	(insert-buffer BUFFER)	<p>Insert after point the contents of BUFFER.</p> <ul style="list-style-type: none"><li>Puts mark after the inserted text.</li><li>BUFFER may be a buffer or a buffer name.</li></ul>
Append region’s text to specified file	<f11> b f	(append-to-file START END FILENAME)	<p>Append the contents of the region to the end of file FILENAME.</p> <ul style="list-style-type: none"><li>This does character code conversion and applies annotations like ‘write-region’ does.</li></ul>
Indirect Buffers	<p>As described in <a href="#">Emacs Indirect Buffer section</a>, “an indirect buffer shares the text of some other buffer, called the base buffer of the indirect buffer. In some ways it is a buffer analogue of a symbolic link between files.”</p> <p>The section also states: “One way to utilize indirect buffers is to display multiple views of an outline” (such as Org-Mode files).</p> <p>The following commands are available to manage indirect buffers.</p>		
Create indirect buffer explicitly	<f11> b I m	(make-indirect-buffer BASE-BUFFER NAME &optional CLONE)	<p>Create and return an indirect buffer for buffer BASE-BUFFER, named NAME.</p> <ul style="list-style-type: none"><li>BASE-BUFFER should be a live buffer, or the name of an existing buffer.</li><li>NAME should be a string which is not the name of an existing buffer.</li><li>Optional argument CLONE non-nil means preserve BASE-BUFFER’s state, such as major and minor modes, in the indirect buffer.</li><li>CLONE nil means the indirect buffer’s state is reset to default values.</li></ul>

Operation	Keystroke	Function	Notes
Create indirect buffer of current buffer	<f11> b I c	(clone-indirect-buffer NEWNAME DISPLAY-FLAG &optional NORECORD)	Create an indirect buffer that is a twin copy of the current buffer. <ul style="list-style-type: none"> <li>Give the indirect buffer name NEWNAME. Interactively, read NEWNAME from the minibuffer when invoked with a prefix arg. If NEWNAME is nil or if not called with a prefix arg, NEWNAME defaults to the current buffer's name. The name is modified by adding a '&lt;N&gt;' suffix to it or by incrementing the N in an existing suffix. Trying to clone a buffer whose major mode symbol has a non-nil 'no-clone-indirect' property results in an error.</li> <li>DISPLAY-FLAG non-nil means show the new buffer with 'pop-to-buffer'. This is always done when called interactively.</li> <li>Optional third arg NORECORD non-nil means do not put this buffer at the front of the list of recently selected ones.</li> </ul>
Create indirect buffer of current buffer in another window	<ul style="list-style-type: none"> <li>C-x 4 c</li> <li>&lt;f11&gt; b I w</li> </ul>	(clone-indirect-buffer-other-window NEWNAME DISPLAY-FLAG &optional NORECORD)	Like 'clone-indirect-buffer' but display in another window.
Buffer View Mode	Several commands (view-buffer, etc..., see at top of this table) activate the View Mode for a buffer where the buffer is essentially read-only and special commands are available.		
View buffer - no modification allowed	<f11> b v	(view-buffer BUFFER &optional EXIT-ACTION)	View BUFFER in View mode, returning to previous buffer when done. <ul style="list-style-type: none"> <li>Emacs commands editing the buffer contents are not available; instead, a special set of commands (mostly letters and punctuation) are defined for moving around in the buffer.</li> <li>Space scrolls forward, Delete scrolls backward.</li> <li>For a list of all View commands, type H or h while viewing. See the View Mode command list below.</li> </ul>
View Mode commands	<p>H, h, ? This message.</p> <p>Digits provide prefix arguments.</p> <p>- negative prefix argument.</p> <p>&lt; move to the beginning of buffer.</p> <p>&gt; move to the end of buffer.</p> <p>o scroll so that buffer end is at last line of window.</p> <p>SPC scroll forward "page size" lines.</p> <p>With prefix scroll forward prefix lines.</p> <p>DEL, S-SPC scroll backward "page size" lines.</p> <p>With prefix scroll backward prefix lines.</p> <p>z like SPC but with prefix sets "page size" to prefix.</p> <p>w like DEL but with prefix sets "page size" to prefix.</p> <p>d scroll forward "half page size" lines. With prefix, sets "half page size" to prefix lines and scrolls forward that much.</p> <p>u scroll backward "half page size" lines. With prefix, sets "half page size" to prefix lines and scrolls backward that much.</p> <p>RET, LFD scroll forward one line. With prefix scroll forward prefix line(s).</p> <p>y scroll backward one line. With prefix scroll backward prefix line(s).</p> <p>F revert-buffer if necessary and scroll forward.</p> <p>Use this to view a changing file.</p> <p>= prints the current line number.</p> <p>% goes prefix argument (default 100) percent into buffer.</p> <p>g goes to line given by prefix argument (default first line).</p> <p>. set the mark.</p> <p>x exchanges point and mark.</p> <p>@ return to mark and pops mark ring.</p> <p>Mark ring is pushed at start of every successful search and when jump to line occurs. The mark is set on jump to buffer start or end.</p> <p>m save current position in character register.</p> <p>' go to position saved in character register.</p> <p>s do forward incremental search.</p> <p>r do reverse incremental search.</p> <p>/ searches forward for regular expression, starting after current page.</p> <p>! and @ have a special meaning at the beginning of the regexp.</p> <p>! means search for a line with no match for regexp. @ means start search at beginning (end for backward search) of buffer.</p> <p>\ searches backward for regular expression, starting before current page.</p> <p>n searches forward for last regular expression.</p> <p>p searches backward for last regular expression.</p> <p>q quit View mode, restoring this window and buffer to previous state.</p> <p>q is the normal way to leave view mode.</p> <p>e exit View mode but stay in current buffer. Use this if you started viewing a buffer (file) and find out you want to edit it.</p> <p>This command restores the previous read-only status of the buffer.</p> <p>E exit View mode, and make the current buffer editable even if it was not editable before entry to View mode.</p> <p>Q quit View mode, restoring all windows to previous state.</p> <p>c quit View mode and maybe switch buffers, but don't kill this buffer.</p> <p>C quit View mode, kill current buffer and go back to other buffer.</p> <p>The effect of c, q and C depends on how view-mode was entered. If it was entered by <b>view-file</b>, <b>view-file-other-window</b>, <b>view-file-other-frame</b>, or M-x <b>dired-view-file</b> (M-x view-file, M-x view-file-other-window, M-x view-file-other-frame, or the Dired mode v command), then q will try to kill the current buffer.</p> <p>If view-mode was entered from another buffer, by &lt;f11&gt; b v, M-x view-buffer-other-window, M-x view-buffer-other frame, M-x view-file, M-x view-file-other-window, or M-x view-file-other-frame, then c, q and C will return to that buffer.</p>		
Buffer Menu Mode	The list of buffers is shown inside its own buffer, "Buffer List" when (list-buffer) is executed. This buffer support the following commands.		
	<p>🔑The full list of key bindings is available via the &lt;f1&gt; m key.</p> <p>🔑⚠️Note that PEL uses (ibuffer) for the C-x C-b key binding, so the list of commands and key bindings that are available differ.</p>		
Buffer Menu Mode keys	<ul style="list-style-type: none"> <li>? : Get help : immediately</li> <li>g : Update buffer list : immediately</li> <li>C-n : next buffer in list : immediately</li> <li>SPC : next buffer in list : immediately</li> <li>n : next buffer in list : immediately</li> <li>C-p : previous buffer in list : immediately</li> <li>p : previous buffer in list : immediately</li> <li>C-d : mark buffer for deletion : deleted when pressing x</li> <li>d : mark buffer for deletion : deleted when pressing x</li> <li>k : mark buffer for deletion : deleted when pressing x</li> <li>s : save buffer : saved when pressing x</li> <li>&lt;DEL&gt; : Move to previous line, remove all marks on buffer : immediately if just after marking</li> <li>M-&lt;DEL&gt; : Remove a specific mark from all buffers : immediately if just after marking</li> <li>u : unmark all marks on buffer : immediately</li> <li>x : execute marked commands (delete buffers marked for deletion) : immediately</li> <li>- : mark buffer as un-modifiable : immediately</li> <li>% : toggle read-only : immediately</li> <li>1 : display emacs in full emacs screen : immediately</li> <li>2 : Display this buffer &amp; next in horizontal window : immediately</li> <li>o : replace other (next) window with this buffer : immediately</li> <li>m : mark buffer to be displayed in windows : when pressing v</li> <li>v : display buffers marked with in as many windows as required : immediately</li> <li>q : quit buffer list : immediately</li> </ul>		
🔑🔑🔑 Complete the list			

Operation	Keystroke	Function	Notes
<b>IBuffer Mode command (1)</b>		'S' - Save the marked buffers. 'A' - View the marked buffers in the selected frame. 'H' - View the marked buffers in another frame. 'V' - Revert the marked buffers. 'T' - Toggle read-only state of marked buffers. 'L' - Toggle lock state of marked buffers. 'D' - Kill the marked buffers. 'M-s a C-s' - Do incremental search in the marked buffers. 'M-s a C-M-s' - lsearch for regexp in the marked buffers. 'r' - Replace by regexp in each of the marked buffers. 'Q' - Query replace in each of the marked buffers. 'I' - As above, with a regular expression. 'P' - Print the marked buffers. 'O' - List lines in all marked buffers which match a given regexp (like the function 'occur'). 'X' - Pipe the contents of the marked buffers to a shell command. 'N' - Replace the contents of the marked buffers with the output of a shell command. '!' - Run a shell command with the buffer's file as an argument. 'E' - Evaluate a form in each of the marked buffers. This is a very flexible command. For example, if you want to make all of the marked buffers read-only, try using (read-only-mode 1) as the input form. 'W' - As above, but view each buffer while the form is evaluated. 'k' - Remove the marked lines from the "Ibuffer" buffer, but don't kill the associated buffer. 'x' - Kill all buffers marked for deletion.	
<b>IBuffer Mode command (2)</b>		<b>Marking commands:</b> 'm' - Mark the buffer at point. 't' - Unmark all currently marked buffers, and mark all unmarked buffers. '* c' - Change the mark used on marked buffers. 'u' - Unmark the buffer at point. 'DEL' - Unmark the previous buffer. 'M-DEL' - Unmark buffers marked with MARK. 'U' - Unmark all marked buffers. '* M' - Mark buffers by major mode. '* u' - Mark all "unsaved" buffers. This means that the buffer is modified, and has an associated file. '* m' - Mark all modified buffers, regardless of whether they have an associated file. '* s' - Mark all buffers whose name begins and ends with '*'. '* e' - Mark all buffers which have an associated file, but that file doesn't currently exist. '* r' - Mark all read-only buffers. '* /' - Mark buffers in 'dired-mode'. '* h' - Mark buffers in 'help-mode', 'apropos-mode', etc. '.' - Mark buffers older than 'ibuffer-old-time'. 'd' - Mark the buffer at point for deletion. '% n' - Mark buffers by their name, using a regexp. '% m' - Mark buffers by their major mode, using a regexp. '% f' - Mark buffers by their filename, using a regexp. '% g' - Mark buffers by their content, using a regexp. '% L' - Mark all locked buffers.	
<b>IBuffer Mode command (3)</b>		<b>Filtering commands:</b> 'M-x ibuffer-filter-chosen-by-completion' - Select and apply filter chosen by completion. '/ RET' - Add a filter by any major mode. '/ m' - Add a filter by a major mode now in use. '/ M' - Add a filter by derived mode. '/ n' - Add a filter by buffer name. '/ c' - Add a filter by buffer content. '/ b' - Add a filter by basename. 'M-x ibuffer-filter-by-directory' - Add a filter by directory name. '/ f' - Add a filter by filename. '/ .' - Add a filter by file extension. '/ i' - Add a filter by modified buffers. '/ e' - Add a filter by an arbitrary Lisp predicate. '/ >' - Add a filter by buffer size. '/ <' - Add a filter by buffer size. '/ **' - Add a filter by special buffers. '/ v' - Add a filter by buffers visiting files. '/ s' - Save the current filters with a name. '/ r' - Switch to previously saved filters. '/ a' - Add saved filters to current filters. '/ &' - Replace the top two filters with their logical AND. '/  ' - Replace the top two filters with their logical OR. '/ p' - Remove the top filter. '/ !' - Invert the logical sense of the top filter. '/ d' - Break down the topmost filter. '/ /' - Remove all filtering currently in effect.	
<b>IBuffer Mode command (4)</b>		<b>Filter group commands:</b> '/ g' - Create filter group from filters. '/ P' - Remove top filter group. 'TAB' - Move to the next filter group. 'M-p' - Move to the previous filter group. '/ \' - Remove all active filter groups. '/ S' - Save the current groups with a name. '/ R' - Restore previously saved groups. '/ X' - Delete previously saved groups.	
<b>IBuffer Mode command (5)</b>		<b>Sorting commands:</b> ',' - Rotate between the various sorting modes. 's i' - Reverse the current sorting order. 's a' - Sort the buffers lexicographically. 's f' - Sort the buffers by the file name. 's v' - Sort the buffers by last viewing time. 's s' - Sort the buffers by size. 's m' - Sort the buffers by major mode.	
<b>IBuffer Mode command (6)</b>		<b>Other commands:</b> 'g' - Regenerate the list of all buffers. Prefix arg means to toggle whether buffers that match 'ibuffer-maybe-show-predicates' should be displayed. '^' - Change the current display format. 'SPC' - Move point to the next line. 'C-p' - Move point to the previous line. 'h' - This help. '=' - View the differences between this buffer and its associated file. 'RET' - View the buffer on this line. 'o' - As above, but in another window. 'C-o' - As both above, but don't select the new window. 'b' - Bury (not kill!) the buffer on this line.	