## Contents for SpexEdit Lite Help

SpexEdit Lite by Little Wing is a shareware version of SpexEdit, a heavy duty professional text editor for the programming professional with a feature set based upon the Unix VI editor.

This help file includes all the information for SpexEdit although many of the features are unavailable in SpexEdit Lite. You can see here all of the features you are missing out on by not ordering the commercial version of SpexEdit from Little Wing.

This Lite version serves three purposes. 1) It gives people access to a nice replacement for notepad with many advanced features for the low price of $20. 2) It helps make people aware of the advanced features of the SpexEdit product available from Little Wing for $99. 3) It provides editing/viewing support for reference products like Windows Grep by Huw Millington (CIS: 100016,3452) which is available in the WINSHARE forum on CompuServe. You haven't used a file search tool until you've used Windows Grep and SpexEdit together. Try it!

For more information on registering your copy of SpexEdit Lite or ordering SpexEdit see Shareware Information.

## Menu Commands

## VI Mode Reference

## VI Mode Overview

## File Menu Commands

| | |
|---|---|
| **New** | Open a window for a new file. |
| **Open...** | Open a file and a new window. |
| **Close** | Close the current window and file. |
| **Save** | Save the contents of the current window. |
| **Save As...** | Save the contents of the current window to the specified file. |
| **Save All** | Save the contents of all modified windows. |
| **Refresh** | Forget any changes and reload current file from disk. |
| **Info...** | Display number of bytes and lines in current window. |
| **Print...** | Send the contents of the current window to the printer. |
| **Exit** | End the SpexEdit session. |
| **Most Recent Files** | Open a window for the specified file. |

## Edit Menu Commands

| | |
|---|---|
| **Undo Line** | Restore the current line to its initial state before changes. |
| **Undo** | Undo the last edit. |
| **Cut** | Cut the current selection to the clipboard. |
| **Copy** | Copy the current selection to the clipboard. |
| **Paste** | Paste the contents of the clipboard to the current insert point. |
| **Delete** | Delete the current selection. |

## Search Menu Commands

**Find...** Displays the find dialog. See also Contextual Movement.

**Find Next** Searches for the next occurrence of the current search pattern.

**Find Previous** Searches for the previous occurrence of the current search pattern.

**Replace...** Displays the replace dialog.

**Match Delim** Locates the delimiter which matches the one to the right of the cursor. If there is no delimiter to the right, the delimiter to the left is used. If a matching delimiter does not exist SpexEdit will beep. Delimiters are: **(**, **)**, **[**, **]**, **{**, **}**, **<**, and **>**.

## Preferences Menu Commands

**Auto Indent** Causes new lines to be indented with the same leading white space as the preceding line (or the following line if the **O** command is used).

**Allow Duplicates** Allows multiples windows to be opened for the same file.

**Confirm On Refresh** Causes a confirmation dialog to be displayed before a refresh.

**Set Tab Width...** Sets the number of columns between tab stops.

**Font...** Sets the font.

**VI Mode Default** Causes all newly opened windows to be in VI mode.

**VI Mode Never** Inhibits VI mode.

**Mouse Click Exits VI** If this is checked, clicking the left mouse button will always leave SpexEdit in insert mode.

## Window Menu Commands

| | |
|---|---|
| **Tile As Rows** | Tiles all windows from top to bottom in a single column. |
| **Tile As Columns** | Tiles all windows from left to right in a single row. |
| **Cascade** | Cascades all open windows. |
| **Display Window** | Causes the specified window to be displayed as the top window. |

## Help Menu Commands

**About SpexEdit...** Display information about SpexEdit.

## VI Fundamentals

The SpexEdit VI mode borrows heavily and faithfully from the standard Unix editor. It is a common experience among Unix programmers to be initially intimidated by what seems to be a bewildering array of terse commands that turn the keyboard into a minefield; however, the fifteen years that VI has remained the standard among Unix professionals serves as testimony that these fears give way to a satisfied mastery of what is actually a very efficient, effective, and even elegant set of commands.

The design of the VI command set is narrowly focused on the programmer who is a touch typist and who spends as much time manipulating text as actually typing it in. A central design goal is that the fingers should not have to stray from the home keys to do any text manipulation or cursor movement. With this in mind, we can group VI commands into basically three categories: scrolling commands, movement commands, and manipulation commands. Commands from these categories are combined to produce complete command sequences that can easily carry out virtually any sort of edit that can be done with a mouse, but without moving the hands from the home keys.

There are only five manipulation commands: **d** for delete text; **c** for change text; **y** for yank text; and **>** & **<** for shift text left and right. To carry out an edit in VI mode you simply type a manipulation command and follow it with a movement command. The affected text is that between the initial cursor position and the position after the move. For example, **w** is a movement command that moves the cursor to the beginning of the next word. So to delete from the current position to the beginning of the next word, you simply type **dw**. The **L** movement command moves the cursor to the last line in the window, so to delete from the current position to the end of the window, type **dL**.

There are only five manipulators but there are (depending on how you count them) over 20 one-stroke movement commands. Plus all movement commands may be combined with a count to repeat them an arbitrary number of times, e.g. **d5w** will delete the next 5 words. Also, any manipulator can be followed by itself to imply line mode, e.g. **dd** will delete the current line; **yy** will copy the current line to the clipboard; and **<<** will shift the current line left one tab stop. These are some of the simplest examples, but you will find that once you master only a few movement commands you will begin to abandon your mouse in favor of a quick edit at the keyboard.

Finally, SpexEdit adds to the VI mode the feature most sorely lacking and probably most criticized about VI, a status bar. SpexEdit continually displays whether you are in VI mode and the state of the current command sequence. You may cancel any VI command sequence at any time by pressing **Escape**.

Where other professional editors give you a sophisticated macro capability and a large set of cumbersome commands that cannot be combined, SpexEdit gives you the simple syntax of **{count}<manipulator>{count}<movement>** and a concise set of instructions that alleviate the need to program your editor.

## Entering and Leaving VI Mode

Pressing the **Escape** key is the only way to enter VI mode. If SpexEdit is in insert mode and the **VI Never** option in the **Preferences** menu is not checked, pressing the **Escape** key will put SpexEdit into VI mode with no state and cause the VI indicator to be displayed in the status bar. If SpexEdit is already in VI mode, pressing the **Escape** key causes the VI state to be cleared.

The simplest way to exit VI mode is to simply click the mouse anywhere in an edit window. This causes SpexEdit to flush all VI state and enter insert mode which behaves exactly like any other standard Windows editor.

The insert, append, and open commands provide alternative methods to exit VI mode. The standard insert command **i** behaves exactly like clicking the mouse at the current insert point; The **I** command moves the cursor to the first non-space character of the current line and then enters insert mode. The append command **a** moves the cursor right one space and enters insert mode; **A** moves the cursor to the end of the line and enters insert mode. The open command **o** inserts a blank line after the current line, autoindenting if the **Autoindent** option is checked, positions the cursor at the end of the new line and enters insert mode; the **O** command does the same thing except the new line is inserted above the current line. The auto indent will always be comprised of the leading white space from the current line.

Additionally the replace command **r** will enter insert mode for a single keystroke to replace the character to the right of the cursor with the next character typed.

All of these commands ignore any count.

The change manipulator **c** also causes SpexEdit to enter insert mode. After the change command sequence, the text to be changed will be selected and SpexEdit will enter insert mode. The first character typed will cause the selected text to be replaced with the character; further typing will be inserted as usual. For example, typing **2cw** in VI mode will cause the next two words to be selected; the next characters typed will replace the words with the characters typed.

## Scrolling

SpexEdit provides a number of ways to scroll the window in either insert mode or VI mode. All work in either mode. First you may use the scroll bars which behave like any Windows program. Second you may use the **Page Up** and **Page Down** keys to scroll the window backward and forward one full screen. **Ctrl-End** positions the cursor at the end of the file. **Ctrl-Home** first positions the cursor at the upper left corner of the window; a second press positions the cursor at the end of the file.

In addition to these there are six standard VI commands for scrolling the window. **Ctrl-E** scrolls the window up one line causing a new line to appear at the bottom of the window; **Ctrl-Y** scrolls the window down one line. **Ctrl-F** scrolls the window forward by one screen minus two lines; **Ctrl-B** scrolls the window backward by one screen minus two lines. **Ctrl-U** scrolls the window up by half a screen; **Ctrl-D** scrolls the window down by half a screen.

All of these commands recognize a count in VI mode. For all except **Ctrl-U** and **Ctrl-D**, these simply cause the command to be repeated the number of times given. For **Ctrl-U** and **Ctrl-D** a count specifies the number of lines to scroll. By default this amount is half the lines that fit in a window. After a count has been given with **Ctrl-U** or **Ctrl-D**, it becomes the default until another count is given.

While all of these commands work in either insert or VI mode, a count can only be given in VI mode.

The last scroll command is **z** which must be followed by **.**, **-**, **Return**, or **Ctrl-M**. **z.** scrolls the current line to the middle of the window; **z-** scrolls the current line to the bottom of the window; and both **z Return** and **z Ctrl-M** scroll the current line to the top of the window. If a count is given, the line specified becomes the current line before the scroll takes place.

All of the keyboard commands ensure that the cursor remains in the window after the scroll; the scroll bar commands do not reposition the cursor.

## Absolute Movement

SpexEdit supports the arrow keys in both VI mode and insert mode for absolute movement. In VI mode several additional keystrokes may be used that don't require movement away from the home keys. The commands **j**, **Ctrl-N**, and **Ctrl-J** all have the same effect as the down arrow; the commands **k** and **Ctrl-P** are the same as the up arrow; the commands **h**, **Ctrl-H**, and **Backspace** behave the same as the left arrow; and the commands **l** and **Spacebar** are the same as the right arrow.

The commands **+** and **Ctrl-M** moves the cursor to the first non-space character on the next line; the **-** command moves the cursor to the first non-space character on the previous line.

The **|** command moves the cursor to the column specified by the count. The **^** command moves the cursor to the first non-space character on the current line. The **0** command moves the cursor to the beginning of the line. The **$** command moves the cursor to the end of the line.

The **G** command moves to the line specified by the count. For example, **200G** moves the cursor to line 200. If no count is specified, **G** moves the cursor to the last line of the file.

The **m** command marks the current position. Follow the command with a single lower case letter to identify the mark. For example **ma** records the current position as mark **a**. The ` (back quote) command jumps to the mark specified after it. For example **`a** would jump to the position recorded under mark **a**.The **'** (single quote) character also jumps to the specified mark, but positions the cursor at the first non-space character on the line. Either jump command may be followed by itself, such as ``, to move the cursor to the previous context, which is defined to be the last position from which you executed a non-relative move. A non-relative move is basically any move for which there is no simple sequence to get back; for example **100G** may move you from line 237 to line 100. To return to line 237 simply type **''**. The one caveat to be aware of with marks is that they are implemented as a (line, column) pair. Any edits that add or remove lines also fix up the mark positions; however, edits that alter line lengths do NOT fix up the mark positions. A mark may become invalid in such instances.

## Contextual Movement

The first group of contextual movement commands move the cursor by words. The three lower-case commands **w**, **e**, and **b** move the cursor to the beginning of the next word, the end of the current word, and the beginning of the current word respectively. The beginning of a word is defined to be the first non-space character following a sequence of valid C-symbol characters or a sequence of non-valid C-symbol characters. A C-symbol character is any character that may appear in a symbol as defined by the **C** language. These characters are **A-Z**, **a-z**, **0-9**, and **_**. Experiment with the **w** command to get a feel for what a word is. The three related upper case commands **W**, **E**, and **B** work the same but with a different definition of word. They define the beginning of a word to be the first non-space character following the first space character that follows the current position. **Ctrl-Right Arrow** and **Ctrl-Left Arrow** carry out the same functions as **W** and **B** respectively. These definitions, while precise, may not give you a good feel for the commands. The best way to understand them is to experiment a bit. You will find them to be quite comprehensible in no time.

The two commands **[** and **]** move the cursor to the previous or next section respectively. A section is defined to be a line that begins with a **{**. Both of these commands will use a count if given. The **%** will find the character that matches the one near the cursor; characters that may be matched are **[{(<**and **]})>**.

The commands **H**, **M**, and **L** move the cursor to the first non-space character of the top line, the middle line, and the last line in the window respectively.

The commands **f** and **F** move the cursor to the next occurrence of the given character on the line. For example, **fa** will move the cursor forward to the next **a** on the line. **F** works the same way but searches backward. Both commands leave the cursor just beyond the character in the direction of the search, i.e. **f** leaves the cursor to the right of the hit and **F** leaves the cursor to the left of the hit. The commands **t** and **T** work similarly but leave the cursor just before the hit, i.e. **t** leaves the cursor to the left of the hit and **T** leaves the cursor to the right of the hit. Finally, the commands **;** and **,** repeat the previous find or to command, **;** in the same direction, **,** in the opposite direction. All of these commands may be used with a count. None of them will move cursor from the current line. These commands are particularly useful in combination with the **c** or change manipulator.

The last group invokes the regular expression search function. The **/** command causes a small dialog to appear into which you may type a regular expression or use the combo box to select an old search string; you may use the arrow keys to scroll through old search strings without dropping the combo box. Upon pressing **Return** SpexEdit will search forward in the file for the first match of the pattern. The **?** command does the same but searches backward. The **n** command repeats the previous search in the same direction. The **N** command repeats the previous search in the opposite direction. None of these commands use a count.

## Manipulators

A manipulator is a command that works with a block of text. A manipulator must always be followed by a movement command. The block of text is defined to be from the cursor position where the command was begun to the position after the movement command. For example **d** is the delete manipulator, and **$** moves the cursor to the end of the line; so **d$** will delete the text from the current cursor position to the end of the line. Any movement command may follow a manipulator. If the movement command uses a count, it may be given either before or after the manipulator; for example, **100yG** and **y100G** will both yank (copy) the text from the current cursor position to line 100 into the clipboard. Most movement commands work exactly as they would if you were not using a manipulator although a few are treated as special cases to make their behavior more "reasonable" with a manipulator. For the most part, you probably won't even notice these special cases because they exist only to make SpexEdit "do the right thing". Also a few movement commands cause the manipulator to work in "line mode", i.e. complete lines will be manipulated rather than just the block of text. These line mode movement commands are: **j**, **+**, **k**, **-**, **G**, **'**, **`**, **[**, **]**, **H**, **M**, **L**, and any synonyms for these.

The five manipulators are **y**, **d**, **c**, **<**, and **>**, which correspond to yank, delete, change, left shift, and right shift. Yank copies the selected text to the clipboard. Delete cuts the selected text to the clipboard. Change selects the text and replaces it with the next characters typed. The shift commands move the selected text left or right by one tab stop; these two always work in line mode.

Instead of following a manipulator with a movement command, you may simply repeat it to imply line mode. For example **yy** will yank the current line to the clipboard and **dd** will delete the current line. The behavior may be modified with a count. For example **5yy** will yank the current line and the next 4 lines to the clipboard.

By default, the yank and delete manipulators place the affected text in the clipboard; however, SpexEdit maintains 26 named buffers which may also receive text from these manipulators. You may specify a buffer by using the " command followed by a single lower case letter. The current buffer is always displayed in the status bar. The Windows clipboard may be selected by entering "". If you specifiy a buffer with an upper case letter, the selected text will be appended to the current contents of the buffer instead of replacing them. Text may be retrieved from a buffer by using the **p** or paste command.

In addition to these buffers, SpexEdit also maintains nine automatic delete buffers. Any type of deleted text is placed in buffer 1. The next time a delete occurs, the contents of each buffer are shifted to the next higher numbered buffer. When text is shifted from buffer 9 it is permanently lost. These buffers can only be specified for a paste operation.

## Manipulator Abbreviations

There are a few abbreviations for manipulators with an implicit block to make common edits easier. All of them may be modified with a count. They are **C** and **D** which change or delete to the end of the line respectively; they are the same as **c$** and **d$**; **Y** is the same as **yy** and will yank the current line; **s** is the same as **cl** which will replace the current character with the next characters typed; **S** is the same as **cc** which will replace the entire line with the next characters typed; **x** is the same as **dl** and will delete the character to the right of the cursor; and **X** is the same as **dh** and will delete the character to the left of the cursor.

The **x** and **s** commands behave subtly different from their non-abbreviated forms. If the cursor is at the end of the line, these two commands will affect the character to the left of the cursor rather than simply beeping at you.

## Undo, Redo, Paste, Join, and Toggle Case

The two paste commands **p** and **P** insert text from the current buffer at the cursor. If the text in the buffer was put there by a line mode manipulator it will be pasted in line mode; **p** will insert below the current line and **P** will insert above the current line. If the text in the buffer was put there by a non-line mode manipulator, it will be inserted exactly at the insertion point; in this case the only difference between **p** and **P** is the final position of the cursor.

The two undo commands **u** and **U** undo the last edit and undo all edits to the current line respectively. The last edit is defined to be the last manipulator command (not including yank), the last join, toggle case, or replace, or the last continuous piece of text typed. If the **u** command is repeated it undoes the last undo.

The redo command **.** is perhaps the most useful command in the entire SpexEdit command set. It causes the last edit as defined above to be repeated at the current cursor position. This can be very useful when combined with the change manipulator. A count may be specified with redo; if no count is given the count originally given is used.

As a special case, if the paste command is repeated with redo and the specified buffer was a delete buffer, the buffer number will be incremented for each redo.

The toggle case command **~** simply changes the case of the character to the right of the cursor if it is a letter and advances the cursor. It may be used with a count.

The join command **J** by default concatenates the current line and the following line. All white space between the two lines is removed and a single space is inserted. If a count n is given, it causes n lines to be joined.

## Manipulator Menu Commands

The manipulator menu provides all of the functionality of the SpexEdit manipulators from the mouse instead of from the keyboard. Pressing the right mouse button causes it to appear.

| | |
|---|---|
| **Close** | Closes the manipulator menu. |
| **Delete** | Deletes the selected text and places it in the current buffer. A line mode delete may be specified by pressing the right mouse button during the selection. |
| **Yank** | Yanks the selected text to the current buffer. A line mode yank may be specified by pressing the right mouse button during the selection. |
| **Put** | Pastes the text from the current buffer. Equivalent to the **p** command. |
| **Undo** | Undoes the the last edit. Equivalent to the **u** command. |
| **Shift <<** | Shifts the selected text left one tab stop. Equivalent to the **<** manipulator. |
| **Shift >>** | Shifts the selected text right one tab stop. Equivalent to the **>** manipulator. |
| **Set Buffer** | Displays the buffer selection submenu which allows the current buffer to be specified. |

## Scrolling Reference

**Ctrl-E** Scroll <count> lines onto the bottom of the window.
**Ctrl-Y** Scroll <count> lines onto the top of the window.

**Ctrl-D** Scroll <count> lines onto the bottom of the window.
**Ctrl-U** Scroll <count> lines onto the top of the window.
If no count is given, the count last given is used. If no count has been given, half the window is scrolled.

**Ctrl-F** Scroll <count> pages forward.
**Ctrl-B** Scroll <count> pages backward.
A page is one window minus two lines.

**z** Must be followed by **Return**, **Ctrl-M**, **.**, or **-**. **Return** and **Ctrl-M** scroll the current line to the top of the window. **.** scrolls the current line to the middle of the window. **-** scrolls the current line to the bottom of the window. A count jumps to the specified absolute line and then positions as above.

See also Scrolling Overview.

## Insert Reference

| | |
|---|---|
| **a** | Move cursor right and enter insert mode. |
| **A** | Move cursor to end of line and enter insert mode. |
| **i** | Enter insert mode. |
| **I** | Move cursor to the first non-space character and enter insert mode. |
| **o** | Open a new line below the current line. |
| **O** | Open a new line above the current line. |
| **r** | Replace the character to the right of the cursor with the next one typed. |

See also Entering and Leaving VI Mode.

## Absolute Movement Reference

**j, Down Arrow, Ctrl-N, Ctrl-J**
Move the cursor <count> lines down.
**k, Up Arrow, Ctrl-P**
Move the cursor <count> lines up.
**h, Left Arrow, Ctrl-H, Backspace**
Move the cursor <count> characters left.
**l, Right Arrow, Spacebar**
Move the cursor <count> characters right.

**+, Ctrl-M** Move the cursor <count> lines down to the first non-space character on the line.
**-** Move the cursor <count> lines up to the first non-space character on the line.

**|** Move the cursor to the column specified by the count.
**^** Move the cursor to the first non-space character on the current line.
**0** Move the cursor to the first character on the current line.
**$** Move the cursor to the end of the line.

**G** Move the cursor to the absolute line specified by the count. If no count is given, move the cursor to the last line of the file.

**m** Record the current position under the mark letter following the **m** command.
**`** Jump to the position specified by the mark letter following the ` command.
**'** Jump to the line specified by the mark letter following the ' command. If no mark is specified, jump to the previous context.

See also Absolute Movement Overview.

## Contextual Movement Reference

**w** Move the cursor to the beginning of the next word.
**e** Move the cursor forward to the end of a word.
**b** Move the cursor backward to the beginning of the previous word.
The beginning of a word is defined to be the first non-space character following a sequence of valid C-symbol characters or a sequence of non-valid C-symbol characters. A C-symbol character is any character that may appear in a symbol as defined by the **C** language. These characters are **A-Z**, **a-z**, **0-9**, and **_**.

**W** Move the cursor to the beginning of the next full word.
**E** Move the cursor forward to the end of a full word.
**B** Move the cursor backward to the beginning of the previous full word.
The beginning of a full word is defined to be the first non-space character following the first non-space character.

**[** Move the cursor to the previous section.
**]** Move the cursor to the next section.
A section is marked by a line that begins with a { character.

**%** Move the cursor to the character that matches the delimiter near the cursor.
Delimiters are [{(< and ]})>.

**H** Move the cursor to the first non-space character of the top line in the window.
**M** Move the cursor to the first non-space character of the middle line in the window.
**L** Move the cursor to the first non-space character of the bottom line in the window.

**f***c* Move the cursor right past the <count>'th occurrence of *c*.
**F***c* Move the cursor left past the <count>'th occurrence of *c*.
**t***c* Move the cursor right to the <count>'th occurrence of *c*.
**T***c* Move the cursor left to the <count>'th occurrence of *c*.
**;** Repeat the last **f**, **F**, **t**, or **T** command.
**,** Repeat the last **f**, **F**, **t**, or **T** command in the opposite direction.

*/regexp* Search forward for the given regular expression.
**?***regexp* Search backward for the given regular expression.
**n** Repeat the last **/**, or **?** command.
**N** Repeat the last **/**, or **?** command in the opposite direction.

See also Contextual Movement Overview.

## Manipulator Reference

| | |
|---|---|
| **d** | Delete the specified text. |
| **c** | Change the specified text. |
| **y** | Yank the specified text. |
| **<** | Shift the specified text left. |
| **>** | Shift the specified text right. |

See also Manipulator Overview.

## Manipulator Abbreviation Reference

| | |
|---|---|
| **C** | Change to the end of line - **c$**. |
| **D** | Delete to the end of line - **d$**. |
| **s** | Change next character - **cl**. |
| **S** | Change line - **cc**. |
| **x** | Delete next character - **dl**. |
| **X** | Delete previous character - **dh**. |
| **Y** | Yank line - **yy**. |

See also Manipulator Abbreviation Overview.

## Miscellaneous Reference

| | |
|---|---|
| **p** | Put the contents of the current buffer after the current position. |
| **P** | Put the contents of the current buffer before the current position. |
| "*c* | Select buffer *c* as the current buffer. *c* may be a-z, A-Z, or 1-9. A-Z causes text put in buffer to be appended to the current contents. |
| **u** | Undo the last edit. |
| **U** | Undo all edits to current line. |
| **.** | Redo the last edit. |
| **~** | Toggle the case of the character to the right of the cursor. |
| **J** | Join <count> lines together. |

See also Undo, Redo, Paste, Join, and Toggle Case Overview.

## Commands Outside VI Mode

**Ctrl-Left Arrow**
Move left one word.
**Ctrl-Right Arrow**
Move right one word.

**Ctrl-Up Arrow**
Scroll window one line up.
**Ctrl-Down Arrow**
Scroll window one line down.

**Home** Moves to first non-white space character on line. Second press moves to column 0.
**Ctrl-Home** Moves to upper left corner of window. Second press moves to first character in file.

**End** Moves to end of line.
**Ctrl-End** Moves to last character in file.

**Keypad -** Deletes current line. Can be used with a count in VI mode.
**Keypad *** Copies current line. Can be used with a count in VI mode.
**Keypad +** Pastes from current buffer. Same as VI **p**.

## Regular Expression Reference

Single-character regular expressions may be constructed as follows:

1) Any character not a special character matches itself.
2) A **** followed by a character matches the literal character.
3) Special characters are **+ * ? . [ ] ^ $ **.
4) The special character **.** matches any single character.
5) A set of characters in square brackets **[ ]** matches any single character from the set. The **-** character can be used to denote a range of characters, e.g. **[a-z]** matches any lower-case letter. If the first character in the brackets is **^** then the expression matches any single character **not** contained in the set.

Multi-character regular expressions may be constructed as follows:

1) A single-character regular expression followed by * matches zero or more instances of the expression. **a*b** matches **b**, **ab**, and **aab**.
2) A single-character regular expression followed by **+** matches one or more instances of the expression. **a+b** matches **ab** and **aab**, but not **b**.
3) A single-character regular expression followed by **?** matches zero or one occurrence of the expression.
4) Any two regular expressions may be concatenated. The expression **[a-zA-Z_][a-zA-Z0-9_]*** matches any C language symbol.
5) Any regular expression can be anchored to the beginning of a line by prepending **^** to it, or the end of a line by appending **$** to it. No regular expression will match a string across a new line character.
6) The following escape sequences may be used to match control characters: **\t** - tab, **\xddd** - the literal hex character 0xddd, and **\^***c* - the literal control character *c*.

## Configuration And Command Line

If the SpexEdit command line is empty it automatically starts in session mode and opens the files being edited the last time it was shut down in session mode. If any files are specified on the command line, SpexEdit does not use command mode and does not save any session information when shut down. Putting "-d" on the command line defeats session mode.

All configuration of SpexEdit should be done through the **Preferences** menu with one exception. The **File Open** dialog filters may be changed by adding to lwspex.ini in the [SpexEdit] section an entry for **FileOpenFilter**. The format for this entry consists of pairs of strings. In each pair, the first string will be displayed in the "List Files of Type:" listbox, and the second string will be used as the filter for the "File Name:" edit control. Every string, including the last, must be followed by a **|** character. For example, the default filter string entry would look as follows:

```
[SpexEdit]
FileOpenFilter=C++ Files (*.cpp;*.h)|*.cpp;*.h|C Files (*.c;*.h)|*.c;*.h|All files (*.*)|*.*|
```

If you primarily edit, say, C and Pascal files you may wish to use:

```
[SpexEdit]
FileOpenFilter=C Files (*.c;*.h)|*.c;*.h|Pascal Files (*.pas)|*.pas|All Files (*.*)|*.*|
```

The only other configuration detail of note is the use of disk caching. SpexEdit stores all text in temporary files in the directory used by Windows, typically specified by the TEMP environment variable. Disk cache software such as SmartDrive can dramatically improve SpexEdit performance. The use of a RamDrive utility for your TEMP directory is also highly recommended.

## Differences From VI

Several elements of the VI command set and behavior have been altered or removed for the SpexEdit implementation usually due to the inherent differences between the text-based interface of VI and the GUI interface of SpexEdit.

**[[** and **]]** have been changed to **[** and **]**.

The find and to set of commands, **f** and **t**, et al., behave slightly differently to accommodate a cursor that rests between characters rather than under them. The functional meaning has been preserved.

Buffer selection with " is more or less permanent whereas in VI it only affects the current manipulator. The unnamed buffer is the clipboard; choosing Cut or Paste from the Edit menu always uses it. The command "" also resets to it.

Shifting works with tab width not shift width.

**R**, the replace command, is not implemented.

In non-linemode **p** and **P** do essentially the same thing; they differ only in where they leave the cursor. This is because of the logical difference between a cursor that is under a char and a caret between two chars. Alas, **xp** no longer twiddles chars, although this may be added as a special case in a future release...

**J**, the join command, always replaces white space with 1 space.

No EX commands have been implemented.

The commands **ZZ**, **Ctrl-L**, **Ctrl-^**, **Ctrl-Z** have been ommitted.

No special keystrokes except accelerators and cursor movement have been implemented in insert mode.

The regular expression syntax is considerably less robust than that of VI. Also, there is no support for parameterized replace expressions. Both of these will be corrected in the next release.

Macro support has not been implemented. This decision stems mainly from the lack of support for the EX command set for which the macro system was primarily designed. Some sort of macro support will probably be introduced in a future release.

## Tips For The New User

Spend some time memorizing the **hjkl** movement keys. You will be surprised at how natural these become after you have them memorized, and you won't have to move to the arrow keys to get around.

Similarly, the **Ctrl-F**, **Ctrl-B**, and **Ctrl-Y**, **Ctrl-E** keys can become very useful with a bit of practice. Note that all of these keystrokes allow you to plant the little finger of your left hand on the **Ctrl** key and then to use remaining fingers to do the scrolling without moving from the home keys. These commands work in either insert mode or VI mode.

Use the **.** redo command. It automatically turns any command sequence into a single-stroke macro. For the experienced user this can become the most used SpexEdit command. When using the change manipulator **c** it is a good idea to always press **Escape** after typing the changed text so the complete change will be available for the redo command.

Use **Ctrl-Tab** to cycle through non-iconized windows.

Quick block moves and copies can be accomplished using the **y** and **d** manipulators. If you can count the number of lines you need to copy or move, say 5 or fewer, you can type **5yy** or **5dd** to yank or delete the lines, move to where you want them, and type **p** to put them. If you have too many lines to easily count in a glance, use a mark and jump sequence. At one end of the block type **ma**; move to the other end of the block and type **y'a** or **d'a** to get the block; move to where you want it, and type **p** to put it.

Use the **''** command (two single-quotes) to jump back to the previous context. Any time you execute a move command for which there is no simple reciprocal command to return, the starting position is remembered as the "previous context". Typing the jump command **'** twice rather than specifying a mark letter jumps the caret to the previous context.

## Technical Support

Technical support may be obtained from Little Wing by calling us at (501) 771-2408 Monday-Friday between the hours 9 AM and 5 PM Central Time or by sending e-mail to 71532,403 on CompuServe or 71532.403@compuserve.com on Internet.

## Shareware Information

SpexEdit Lite is distributed as shareware. It is not free or public domain. You may copy and distribute SpexEdit Lite freely but if you use it for more than 30 days you are legally and morally obligated to pay a license fee of **$20** or the equivalent in local currency (not currently excepting rubles...).

If you have an account on CompuServe it's very easy to register SpexEdit Lite on-line. Simply GO SWREG and follow the instructions. When prompted enter the registration code for SpexEdit Lite. You will be notified by Little Wing of you registration number usually within 24 hours via CompuServe mail.

You may also register SpexEdit Lite by mail. Just send your name, company name (if applicable), address, and a check for $20 to

Little Wing
4618 JFK Blvd., Suite 176
N. Little Rock, AR 72116

Also feel free to include any suggestions or comments about SpexEdit Lite.

If you would like information on SpexEdit or would like to order please call Little Wing at (501) 771-2408 from 9AM-5PM Central weekdays.