NAME

xvi − multi-file text editor

SYNOPSIS

xvi { **−R** } { **−c** *command* } { **−s** *parameter-assignment* } [**−t** *tag* | **+***number* | **+/***pattern*] { *filename* ... }

DESCRIPTION

Xvi (pronounced *ecks-vee-eye*) is a free, portable, multi-window implementation of the popular **vi**(1) editor. It has some useful enhancements, although, as described below, not all of **vi**'s features have been implemented yet, and some things work differently from **vi**.

This manual page describes the differences between **xvi** and POSIX, classic **vi** and other popular **vi** clones. For a full reference manual, see the manual page for classic vi under http://ex-vi.sourceforge.net and the POSIX specifications for ex and vi. For a tutorial, see *An Introduction to Display Editing with Vi*.

OPTIONS

The following command-line options are available:

−R Start the editor in read-only mode.

−c *command*
Execute *command* as an ex command in the first buffer after reading input files or going to a tag. Multiple **−c** flags are allowed.

−s *parameter-assignment*
Set the value of the specified parameter at startup. The assignment has the same form as when given as an editor command, i.e:

*name***=***string*
for string parameters

*name***=***number*
for numeric parameters

name to turn a Boolean parameter on

no*name*
to turn a Boolean parameter off

Parameters set with **−s** take their new values before opening input files or going to a tag.

−t *tag* Edit the file containing the definition specified as *tag*, at the start of the definition (as per **vi**).

+*number*
Go to the specified line number of the file being edited.

+$ goes to the last line,

+-*number*
goes to the *n*th line before the last line and

+- goes to the penultimate line.

+/*pattern*
Go to the first occurrence of the specified *pattern* within the file being edited.

The **−l**, **−L**, **−r**, **−V** command line options are not supported and **−w***n* is ignored.

ENHANCEMENTS

Parameter handling

Xvi supports 5 types of parameter: as well as **vi**'s *numeric*, *string* and *boolean*, it also has *enumerated* and *list* types. The former is used for **format**, **infoupdate**, **jumpscroll**, **preserve** and **regextype**, while the latter is currently only used for **tags**.

The advantage of the *enumerated* type is that if you try to set an illegal value, the set of correct values will be displayed, which is very useful if you have forgotten what the values may be. (Try **:set preserve** to see an example of this.)

Like the **:** commands and parameter names, an enumerated value can also be set by giving enough of its initial characters to uniquely identify it.

Multiple buffers and windows

Xvi supports multiple *buffers* and *windows*. A *buffer* is the internal object which holds a file in memory, while a *window* is an area of the screen which shows part of a buffer. Every window references a buffer, even if no file is being edited.

The following commands are available for operating on buffers and windows:

:buffer create a new buffer in a new window; can be followed by a filename, which will be edited in the new buffer.

:close close the current window; will also close the buffer if this is the last window on to it.

:equalise
make all windows as nearly the same size as possible.

:split create a new window on to the current buffer by splitting the current window in half. The two resulting windows are similar to *viewports* on to a single editing buffer, in that changes made in one window are reflected in the other one.

:x / ZZ close only the current window. If the window is the only one on to the buffer, the buffer will be closed as well, writing it first if it is modified.

g move to the next window.

^W increase the size of the current window (may be given a numeric prefix, default is one line).

^T decrease the size of the current window (may be given a numeric prefix, default is one line).

^O make the current window as large as possible.

^] as for **vi**, but create a new buffer window if appropriate (and if **autosplit** allows).

Note that the **:quit** command quits out of the editor, not out of a window. The **:close** command is thus the equivalent of **:quit** for windows. There is no equivalent of **:x** or **ZZ** for the whole editor; these have been hijacked for operations on windows.

The boolean **autosplit** parameter specifies whether of buffer windows that will be created automatically whenever you either edit more than one file, or use tags to edit a different file.

The **minrows** parameter specifies the minimum number of rows to which a window may be shrunk, including the status line. The default value is 2; 0 and 1 may also be useful.

Undo works per buffer, as do marks; yank/put and redo (the **.** command) work over all buffers, i.e. you can delete from one buffer and put the text into a different buffer.

Named buffers

As well as the normal named (conjugate) buffers, and the default one named @, several extra buffers named **:**, **/**, **?**, **!** and **<** contain the last command lines entered for each of the four command types and for the last thing inserted. So, for instance, **@:** will re-execute the last colon command, or you can insert it into your buffer, edit it and then re-execute it (e.g. with **dd@@**).

Function keys

For keyboards with function keys, F1 displays a page of help, while F2 to F10 produce **#2** to **#9** and **#0**, which you can **:map!** or **:map** to phrases that you type often or to useful command sequences.

File preservation

Rather than use **vi**'s Unix-specific method for preservation, **xvi** does periodic preservation of all files currently being edited into temporary files in the same directory. It tries to do this when you aren't typing, so that you won't notice the short delay when the temporary file is written out. Obviously, only changed files are preserved in this way, and the temporary file is removed once the real file has been successfully written.

As an additional safety measure, when a file is explicitly saved and it appears not to have been preserved

recently, it is normally preserved first. This ensures that, even if the operating system crashes while the real file is being created, there should always be at least one recent copy of it in the filesystem. The **:preserve** command is available as in **vi** to preserve a specific buffer manually.

The level of safety provided by the preservation facility may be configured by changing the values of the **preserve** and **preservetime** parameters. The following values are available for **preserve**:

unsafe Never preserve any buffer before an explicit save. This can be useful on old, slow, floppy-only systems, but is not generally recommended.

standard
The default value. Only preserve a buffer before an explicit save if it appears not to have been preserved recently.

safe Always preserve buffers before they are written.

paranoid
As for **safe**, but the preserve file is never removed, even after the file has been successfully written.

In all cases, all modified buffers are preserved automatically after no user events have been received for **preservetime** seconds, if a minimum number of events (currently 60) have been received since the last automatic preservation. This behaviour can be more or less disabled by setting **preservetime** to a very high value.

The names given to preserve files are system-dependent, but are generally of the form "*filename*.tmp", or "*filename*.001" to "*filename*.999". If a preserve file already exists, it will not be overwritten; instead, a new filename will be generated.

8-bit character support

Characters with the top bit set may be displayed, although it is not yet possible to have null ('\0') bytes in a file buffer. How the characters are displayed varies between systems; on UNIX, they will be shown as an octal escape sequence, while on MS-DOS, OS/2 and QNX they will be shown as the actual character in the PC character set. This can be controlled by setting the **cchars** and **mchars** variables; if these parameters are set, control- and meta-characters (respectively) are shown directly, otherwise they are shown as some sequence of printable characters.

Tabs are normally displayed as a series of spaces of the appropriate length (according to the **tabstops** parameter); setting **list** mode will cause them to be displayed as control characters, as will unsetting the **tabs** parameter. How the tab character is displayed is then under the control of the **cchars** parameter.

You can use the ^_ (control-underscore) command to flip the top bit of the character the cursor is on. This may be useful on systems where it is otherwise impossible to enter 8-bit characters.

File formats

Xvi can read and write text files in non-Unix formats. The current format is given by the value of the **format** parameter, which may be set to "**unix**", "**msdos**", and so on. This means you can edit MS-DOS files under UNIX, etc. To see a list of available formats, type

:se fmt=?

If the new boolean **autodetect** parameter is set, xvi sniffs files before reading them to determine their newline style and sets the default file-saving newline style to that of the file read.

It would be better if the file format were remembered separately for each file but instead it is a single editor-wide option that applies to all open buffers. The same could be said for the "readonly" option.

Extended regular expressions

vi's **magic** parameter is superseded by the **regextype** parameter, which can take the following values:

tags only ^ and **$** are significant (used for tags)

grep like **grep**(1), but with **\<** and **\>** added

egrep like **egrep**(1), but with **\<** and **\>** added

The default is **grep**.

Note that it is still possible to set or unset **magic** as in **vi**; this will simply result in **regextype** being set as appropriate.

The **sections** and **paragraphs** parameters define **egrep**-style patterns to search for, rather than **vi**'s simplistic (and **troff**-dependent) character pairs.

A similar parameter, **sentences**, defines a pattern for the **(** and **)** motions.

Improved replace mode

The **R** command acts more intelligently when you press return — it leaves the rest of the current line alone, and just starts replacing text on the next line, starting at the screen column where you first typed **R**.

Command line editing and filename completion

While entering a ':' command or a '/' search string, as well as the usual keys, **Backspace** to cancel the previous character, **^W** to cancel the previous word and **^U** to cancel the line, **xvi** also lets you move back and forth in the line with the arrow keys to correct typing errors.

For file-oriented commands, the **Tab** key performs filename completion on the last word of the line, which can be the first part of a file's name or a filename regular expressions containing special characters **?**, ***** and maybe others, depending on your operating system.

Command re-execution

As well as the normal named (conjugate) buffers, and the default one (named @), there exist several extra ones named **:**, **/**, **?** and **!**, which contain the last command lines typed to each of the given commands. So for instance, **@:** will re-execute the last **ex** command, or you can insert it into your buffer, edit it and then re-execute it (e.g. with **dd@@**).

Scrolling

When multiple windows are used, **xvi** normally has to be able to scroll individual windows without scrolling the whole screen. This can be very inefficient on terminals without scrolling regions, so the **jumpscroll** parameter is provided to control the editor's scrolling behaviour. It can be set to one of:

off When the cursor moves outside a window's boundaries, and the new position is near enough, the window will scroll to the new position.

on When the cursor moves outside a window's boundaries, the window will always jump to the new position.

auto A window will scroll only if it can do so efficiently; otherwise it will jump.

The default value is **auto**.

On ISA-type systems which have memory-mapped displays, hardware character generators and reasonably fast processors, **jumpscroll** should generally be set to **off**; however, on LCD screens or other displays with a long image persistence, this may actually make the text more difficult to read, and many users may be more comfortable with it turned **on**.

Explicit scroll commands (e.g. **^D** and **^E**) are not affected by the **jumpscroll** parameter.

Colour

There are four new parameters to control screen colours:

colour colour used for text

statuscolour
colour used for status lines

roscolour
as **statuscolour**, but for read-only files

systemcolour
colour used for system mode (i.e. subshells and after termination)

These parameters are numeric, and the value means different things on different operating systems. On Unix, it is an index into the **termcap**(5) entries "**c0**" to "**c9**", which are assumed to be colour-setting escape sequences if they are present. If they are not present, "**so**" (begin standout mode) and "**se**" (end standout

mode) are used instead. Values of 0 and 1 give normal text, whereas 2 and above give standout mode.

The default colour for the **roscolour** parameter will generally involve red if colours are available; this is intended to provide a warning to the user that writing the file may not be possible.

On-line help

A primitive help facility is available; the **:help** command simply creates a new buffer window on to a standard help file. The name of the file which is edited is given by the **helpfile** string parameter; the default on Unix versions is **"/usr/lib/xvi.help"**.

Note that the help file buffer will be marked "not editable" when it is created, which prevents accidental overwriting of the help file even when the file permissions would allow it.

Mouse support

Some mouse support is available for micro-based systems and workstations Clicking the mouse button on:

any line outside current window
: changes current window to the one indicated by the mouse (can be used instead of **g**).

top line of any window
: scrolls window downwards (same as **^Y**).

bottom line of any window
: scrolls window upwards (same as **^E**).

status line of any window
: shows current file and lines (same as **^G**).

any text line of any window
: moves text cursor as near as possible to mouse cursor.

Also, windows can be resized by dragging the appropriate status line up or down with the mouse.

Miscellaneous

The command **:wn** (write file and edit next) is provided, as in **PC-vi**.

In insert and replace modes, **^A** has the same meaning as ^@ in vi, except that it works at any time, not just for the first character. Also, typing **^B***x* where *x* is the name of a conjugate buffer, inserts the contents of that buffer into the input stream at that point.

A new parameter **infoupdate** (**iu** for short), when set to **continuous** shows the current line number on the status line as you move in the file. Its default value of **terse** just shows the filename.

When the boolean parameter **tabindent** is clear (**:set notabindent***), automatic indentation does not use tab characters and is done with spaces.*

The **posix** parameter, set automatically if environment variable **POSIXLY_CORRECT** is set, tweaks xvi's behaviour to conform to the POSIX specification instead of to what classic vi does. At present it:

- makes the execution of a character-mode buffer using **:@** or **:*** *not* terminate the last line of the buffer with a newline, leaving it on the command line waiting for you to press Enter;
- makes the **$** command or target on an empty line fail instead of succeeding.

LIMITATIONS

Ex mode

The main area in which **xvi** is lacking is **vi**'s **ex** mode, which is not implemented at all (and neither are **edit**, **e**, or **open** modes). However, many of the **ex** commands are available in **vi** mode as colon commands; the colon commands that have not been implemented are mostly those which offer the same functionality as other commands in **vi** mode.

In particular, **abbreviate**, **append**, **change**, **ex**, **insert**, **open**, **recover**, **unabbreviate**, **z** and **|** have not been implemented as colon commands yet.

Vi mode

The **Q** command is inappropriate in the context of **xvi**, since there is no **ex** mode.

Parameters

The following parameters have not been implemented, and probably won't be:

ada(vim), **adapath**(vim), **autoprint**, **directory**, **edcompatible**, **hardtabs**, **lisp**, **mesg**, **modelines**, **open**, **optimize**, **prompt**, **redraw**, **slowopen**, **sourceany**, **term**, **terse**, **ttytype**, **window**

The command **:se all** gives a complete list, with current values, of those that have been.

Miscellaneous

It is not possible to interrupt the editor while it is performing certain operations. If you start off a big global command, you have to wait for it to finish.

Flags and counts after **ex** mode commands are not supported.

The **:substitute** command does not support splitting of lines.

Regular expressions, although implemented (see above), do not support the ~ metacharacter on the left hand side.

The **:global** command only supports the commands [**lps&~d**].

OTHER DIFFERENCES FROM VI

If the **XVINIT** environment variable is set, it is read instead of **EXINIT**.

The **tags** parameter can be used to specify multiple tags files; these can be separated by either "\ " (backslash space) or "," (comma).

Alternate files are handled slightly differently, owing to the presence of buffer and window handling. Essentially, when you close a buffer, its filename is remembered as the alternate file; when you invoke the ^^ or **:e #** commands, this file is re-edited. Note that ^^ edits the alternate file in a new buffer window, if **autosplit** allows.

FILES

/usr/lib/xvi.help Default help file.

SEE ALSO

ex(1), **vi**(1), **termcap**(5).

BUGS

See the BUGS and ISSUES files in the source code.

- **Undo** does not work properly when applied to macros (either @ or **:map**); it should undo all the changes made by the macro, but in fact only the last command within the macro is undone.
- Most **termcap**(5) terminal descriptions are only tested with **vi**(1) (and possibly **rogue**(6)). Since **xvi** is, in some ways, more demanding than **vi** in its use of **termcap** capabilities, it sometimes exposes bugs or inadequacies in **termcap** entries. This applies especially to scrolling regions.

AVAILABILITY

Xvi has been ported to MS-DOS, OS/2, QNX, Atari ST, Amiga and many different versions of Unix. Downloads are available from http://martinwguy.net/xvi and from http://xvi.sf.net and the source code is maintained at http://gitlab.com/martinwguy/xvi

AUTHORS

Chris and John Downey.

Derived from STEVIE, written by Tim Thompson and Tony Andrews.