9. Operating System Notes

Elvis 2.1 currently runs under MS-DOS, MS-WindowsNT/MS-Winsdows95, OS/2, and many versions of UNIX. This chapter describes the quirks of each implementation.

In addition, elvis has been reported to work under a few other operating systems. It isn't officially supported under these yet.

9.1 MS-DOS

Elvis 2.1 was ported to MS-DOS by Steve Kirkendall, using code from Guntram Blohm and Martin Patzel's MS-DOS port elvis 1.X. It also uses Thomas Wagner's "exec" function to swap elvis out to EMS/XMS memory or disk, while running another program.

The "os" option
Under MS-DOS, the os option's value is "msdos".

The "shell" option
The default value of shell is "C:\COMMAND.COM". This can be overridden by the value of the COMSPEC environment variable.

The "lpout" option
The default value of the lpout option is "prn", which causes the :lpr command to write its output directly to the printer.

I tried to make it use the spooler program, PRINT, but failed. The first problem I ran into was the fact that PRINT can't read data from stdin; it must be given the name of a file. To work around this problem, I modified the osmsdos/osprg.c file to allow $1 in the command line to be replaced by the name of elvis' temporary file that contains the print data.

This lead to the the second problem: PRINT doesn't make a copy of the file, and elvis deletes the temporary file as soon as PRINT exits. Since PRINT exits before the file has been printed completely, this means that the file is deleted before PRINT is through reading it. The print job is truncated. I was unable to find a way around this problem, so I gave up and just set lpout=prn.

The "ccprg" option
The default value of the ccprg option is "cl /c ($1?$1:$2)", which is appropriate for MSVC++ 1.5. This way, you can run :cc without arguments to compile (but not link) the current source file. If you supply arguments, they'll be used instead of the filename.

The "makeprg" option
The default value of the makeprg option is "make". You might consider changing it to "nmake -f ($1?$1:basename($2)".mak")", would allow MSVC++ 1.5 to compile "foo.c" via the "foo.mak" makefile, unless you invoke :make with command-line arguments naming a different makefile.

The "elvispath" option

The default value of the elvispath option is " ~\elvislib;BIN\lib;BIN", where BIN is the directory where elvis.exe resides. Note that elements of the path are separated with semicolons. The ~ character is replaced by the value of the home option, which is also usually the name of the directory where elvis.exe resides.

The "blksize" option

The blksize option controls the size of elvis' data blocks. Under other operating systems this is usually 2048, but for MS-DOS is it reduced to 1024 because memory is tight.

If you're going to be editing a large file, you may want to increase the block size. The only way to do that is to invoke elvis with a "-b2048" argument. You can't change it via ":set blksize=..." because by then the session file would have already been created with the old block size. Once elvis starts, you should probably ensure that the blkcache is no more than 6 or so, or else elvis may run out of memory unexpectedly.

The "TERM" environment variable

The TERM environment variable tells elvis how to access the screen. If it is undefined, or set to "pcbios", then elvis will use BIOS calls to access the screen. This should work on all MS-DOS systems, but it isn't very fast.

If TERM is set to "dosansi" then elvis will output escape sequences which are supported by the ANSI.SYS driver. This isn't recommended; ANSI.SYS is just as slow as the BIOS, and it isn't as powerful so elvis is forced to redraw the screen instead of scrolling it, in some circumstances.

If TERM is set to "nansi" then elvis will output escape sequences which are supported by more capable drivers, such as NANSI.SYS and FANSI.SYS. These drivers usually bypass the BIOS, so they are very fast. If you're looking for a way to speed up elvis, this is probably your best bet. The Web URL for these drivers is listed in the Tips chapter.

Elvis also supports "vt220" which use DEC VT-220 escape codes, and "ansi" which uses true ANSI escape codes for both input and output. "ansi" doesn't work with ANSI.SYS!

The "termcap" user interface

Under MS-DOS, elvis normally uses the termcap user interface. It supports the :color command, and the mouse.

Missing features

Because elvis must run in the lower 640k, it is normally compiled with certain features disabled. If you want to enable these, or disable more of them, you'll need to edit the config.h file and recompile elvis. Normally the missing features are:

* **PROTOCOL_HTTP** and **PROTOCOL_FTP** - These two must remain disabled, because no source code is provided for supporting

these features under MS-DOS. You can't enable them unless you write the necessary WinSock code. (And if you do write that code, please share it with me!)

* **FEATURE_SHOWTAG** - The showtag option, which attempts to continually display the name of the function you're editing.
* **FEATURE_COMPLETE** - The <Tab> key in ex command lines normally attempts to complete a file name or other word.
* **FEATURE_RAM** - This would allow elvis to store its edit buffers in EMS/XMS memory instead of a session file, by specifying "-f ram" on the command line. This makes elvis run much faster, but you can't recover the contents of the edit buffers after a crash. Also, the Microsoft functions for accessing EMS/XMS memory are bulky and require a fairly large buffer space, so if you #define FEATURE_RAM then you must #undef every other FEATURE_XXXXX, and even then you may run out of memory occasionally.

9.2 MS-Windows95/MS-WindowsNT

Elvis 2.1 was ported to Windows95/WindowsNT as a text application by Steve Kirkendall. The graphical user interface was written by Serge Pirotte.

Windows doesn't allow a single program to have both a text-mode interface and a graphical interface, so we include two separate versions of elvis. ELVIS.EXE is the text-mode version, and WINELVIS.EXE is the graphical version. Run "elvis -G?" for a list of the text-mode user interfaces supported; you will generally want to use the "termcap" interface. The graphical version, of course, is unable to output such a list. It supports two user interfaces, named "windows" and "quit".

The "os" option
Under Windows95/WindowsNT, the os option's value is "win32".

The "shell" option
The default value of shell is "cmd". If the COMSPEC environment variable is set (and it normally is), then its value will be used instead of "cmd". This is important, because Windows95 uses "COMMAND.COM" instead of "CMD.EXE".

The "lpout" option
The default value of the lpout option is "prn", which causes the :lpr command to send text directly to the printer.

The "lptype" option
The default value of the lptype option is "dumb" in elvis.exe, or "windows" in WinElvis.exe.

The "windows" printer type is only available in WinElvis; it causes elvis to use the standard Windows graphics-oriented print spooler. This is a very slow way to print, but it is portable and produces good-looking output.

The "ccprg" option

The default value of the ccprg option is "cl /c ($1?$1:$2)", which is appropriate for MSVC++ 2.0 or later. This way, you can run :cc without arguments to compile (but not link) the current source file. If you supply arguments, they'll be used instead of the filename.

The "makeprg" option

The default value of the makeprg option is "make". You might consider changing it to "nmake -f ($1?$1:basename($2)".mak")", would allow MSVC++ 2.0 (or later) to compile "foo.c" via the "foo.mak" makefile, unless you invoke :make with command-line arguments naming a different makefile.

The "elvispath" option

The default value of elvispath is "~\elvislib;BIN;BIN\lib", where BIN is the name of the directory where ELVIS.EXE resides. Note that elements of the path are separated with semicolons. The ~ character is replaced by the value of the home option, which is usually C:\users\default for WindowsNT.

Unlike MS-DOS, Win32 doesn't pass the full pathname of the .EXE file as an argument to the program. So the when the Win32 version of elvis starts up, the first thing it does is search for ELVIS.EXE in your execution path. It must do this in order to find the value to use as BIN.

The "TERM" environment variable

The TERM environment variable tells elvis how to access the screen. It should probably be left undefined, or set to "console" or "cygwin". Other values have not been tested, but the following should work via terminal emulators or whatever: dosansi, nansi, vt100, vt100w, and vt52.

Console size

Win32 distinguishes between a console's buffer size and its window size. Many users like to set the buffer to a huge size (e.g., 100 lines) but leave the window set at 25 lines. They do this so they can scroll back and review earlier programs' output.

When elvis starts, it creates a separate buffer which is the same size as the window (except under Windows95 which has bugs in this area). Elvis does this mostly so the scrollbar will go away. If you change the buffer size while elvis is running, elvis will adjust the size of the window to match the new buffer size, and then redraw the text to take advantage of the new screen size.

You can change the console buffer's size by setting the ttyrows and ttycolumns options. In WindowsNT, you can also change the size via the console's Properties dialog.

When you exit elvis, the console will revert to its original buffer and original window size.

Mouse

In WinElvis (graphical mode), the mouse works about like you'd expect. You can select character spans by dragging the mouse through the text while holding the left button down. You can select a rectangular area by dragging with the right button down. To select whole lines, move the mouse to the left margin, where the cursor shape changes to an up-and-right arrow, and start dragging with the left button there. Double-clicking on a word with the left button will do a tag search on that word; double-clicking the right button pops returns to the original position. At present, there is no way to bring up a menu with the right button.

In elvis (text mode), the mouse should work both in full-screen mode and in a window. Dragging with the left, right, or both buttons pressed is will select characters, whole lines, or a rectangular block, respectively. If you have a three-button mouse and the appropriate driver, dragging with the middle button pressed will also select a rectangular block. Clicking the left button will cancel a pending text selection, and move the cursor. Clicking the right button will move the cursor; if a text selection is pending, it will be extended to the new cursor position. Double-clicking the left or right button will follow a hypertext reference or return from one, respectively.

There is a weird bug in WindowsNT's text-mode mouse support. If, when the console is shown in a window, part of that console is located off-screen, then the mouse won't be able to move there, *even if you're currently in full-screen mode!* Since WindowsNT will sometimes resize your console when you switch between full-screen and windowed modes, this might not be obvious. But if the mouse refuses to move onto part of your full-screen console, I suggest you switch back to windowed mode and reduce the size of your font there. When you switch back to full-screen mode again it will look exactly the same, but the mouse should be able to go where no mouse has gone before.

Cygwin

If you use Cygwin's mount utility to define a mount table, then elvis will try to use it. I say "try to" because that table is maintained in the Win32 registry, and although elvis can handle the current version of Cygwin, later versions may move the mount table to somewhere that elvis can't find. Elvis doesn't use the normal Cygwin functions for mapping file names via the mount table, because I didn't want to make elvis be dependant upon the CYGWIN.DLL library; instead, elvis uses standard Win32 registry functions to read the mount table. use t

Only elvis has been modified to use the mount table. The other utilities (such as ctags, fmt, and elvis' version of ls) will ignore the mount table. Also, even elvis will ignore it with respect to the names of the session file and temporary files (the session, sessionpath, and directory options).

Some problems have been reported when running the text-mode version of elvis via the bash shell. I'm trying to fix that. The graphical version of elvis works fine, though.

9.3 OS/2

Elvis 2.0 was ported to OS/2 by Lee Johnson, but that port was too late to be included in the standard distribution of elvis 2.0. After that, Martin "Herbert" Dietze took over the job, and brought it up to date for elvis 2.1. OS/2-specific bug reports or comments should be sent to herbert@paulina.shnet.org.

There are four separate versions of elvis for OS/2, each one compiled with slightly different features. The version that most people will want to use is named elvis.exe. It uses the vio user interface, and includes network support so elvis can read via HTTP, and read or write via FTP.

But a few OS/2 systems might not have the TCP/IP libraries installed, so the normal elvis.exe version won't work there. A stripped-down version named elvis-no-tcp.exe is provided for use on those systems. It also uses the vio user interface, but it omits the network support so it doesn't require any special libraries.

There is also a version named elvisemx.exe which supports the termcap user interface in addition to the standard vio interface. This should allow you to run elvis over a telnet connection. However, it requires the EMX.DLL library, in addition to OS/2's TCP/IP libraries. (See the Tips chapter for EMX links.)

The last version is named elvisx11.exe. It supports the x11 graphical interface in addition to VIO and termcap interfaces. For instructions on how to run elvis with X11 under OS/2, see section 9.3.1 below. It requires the TCP/IP and EMX libraries, and at least some parts of the XFree86 package.

The following lists the OS/2-specific quirks that are common to all versions:

The "os" option
Under OS/2, the os option's value is "os2".

The "shell" option
The default value of shell is "cmd.exe".

The "lpout" option
The default value of the lpout option is "prn", which causes the :lpr command to send text directly to the printer.

The "ccprg" option
The default value of the ccprg option is "gcc -c ($1?$1:$2)". You may want to change it. This way, you can run :cc without arguments to compile (but not link) the current source file. If you supply arguments, they'll be used instead of the filename.

The "makeprg" option

The default value of the makeprg option is "make $1".

The "elvispath" option

The default value of elvispath is "~\elvislib;BIN;BIN\lib", where BIN is the name of the directory where elvis*.exe resides. Note that elements of the path are separated with semicolons. The ~ character is replaced by the value of the home option.

The "TERM" environment variable

Although the OS/2 version of elvis may support the "termcap" interface, it usually uses its own "vio" interface for text console I/O. The "vio" interface ignores TERM completely.

But the termcap interface is still supported, if compiled using emx/gcc and linked against emx.dll both the usual "vio" and a termcap based interface are available.

The mouse

The vio and termcap interfaces ignore the mouse. The X11 interface does use the mouse, as described in the GUI chapter.

9.3.1 X11 under OS/2

To compile the X11 version of elvis for OS/2, you need the full XFree86 package. The Tips chapter contains a link to the Web page for the OS/2 version of XFree86.

You should be able to run elvisx11.exe with XFree86 without any trouble. You can also run elvisx11.exe under PMX provided you install a few critical files from XFree86, as described in the following steps. (Note that these steps are only necessary to run elvisx11 under PMX. If you're using XFree86 then you can skip this.)

1) Unzip the XFree86 archive (normally XF32BIN.ZIP).

```
unzip xf32bin
```

2) Copy the file x11.dll from XFree86\lib to a place in your LIBPATH.

```
copy XFree86\libx11.dll e:\emx\dll
```

3) Copy the complete directory hierarchy XFree86\lib\X11\locale to a location of your choice (don't change it, e.g. by moving 'locale' to XFree86 etc.). The following installs it under e:\emx.

```
md e:\emx\XFree86
md e:\emx\XFree86\lib
md e:\emx\XFree86\lib\X11
xcopy XFree68\lib\X11\locale e:\emx\XFree86\lib\X11\locale\ /s
```

4) Set the X11ROOT environment variable to the drive and directory that contains XFree86 using forward instead of backslashes to specify paths.

```
set X11ROOT=e:/emx
```

5) Remove the XFree86 files, now that you've copied the ones you need. Since OS/2 doesn't have a shell command for removing a whole directory tree in one step, the easiest way to do this is via the WPS (OS/2's GUI).

9.4 UNIX

Elvis was originally written under UNIX, so it seems strange to say I ported it to UNIX, but I'll say it anyway: Elvis 2.1 was ported to UNIX by Steve Kirkendall, with a lot of feedback from many people on the net.

Since elvis (and vi, for that matter) were originally designed and written for UNIX, there aren't many quirks that show up for generic UNIX. Most of this section will describe the quirks of individual UNIX versions.

The "os" option

Under all versions of UNIX, the os option's value is "unix".

The "lpout" option

The default value of the lpout option is either "!lp -s" or "!lpr". The choice is made by the configure script; if /usr/bin/lp exists then it will use "!lp -s", else it will use "!lpr".

The "elvispath" option

The default value of elvispath is usually ~/.elvislib:/usr/local/lib/elvis. The /usr/local/lib/elvis member of that path can be set via the **--datadir=***directory* flag to the configure script.

Missing functions

If your linker reports an undefined function, such as strdup(), then you should check the need.h and need.c files; they contain implementations of many such functions. To use elvis' version of the missing function, edit the config.h file and change "#undef NEED_XXX" to "#define NEED_XXX".

In all of the officially supported Unix variants listed below, the configure script automatically selects any necessary NEED_XXX macros.

9.4.1 UNIX versions

The configure script works around most of the quirks of individual UNIX versions. Here, I will describe what configure does, or fails to do, for each type of UNIX.

SunOS and Solaris

SunOS and Solaris2 both claim to be SunOS, according to the uname command. However, their configuration is quite different, so the configure script distinguishes between them by looking at the revision number output by "uname -r".

The SunOS port uses BSD's sgtty ioctl calls, instead of the POSIX calls, even though SunOS supports the POSIX calls. This choice was made because differences in signal handling were preventing the SunOS port from responding to window resize signals.

Solaris2 seems to have a bad implementation of rlogin, at least on the system where I tested it. It couldn't handle 2000-character write() calls. Because this is a communication issue, and not purely a Solaris issue, I decided to reduce the size of the output buffer for *all* systems to 1500 bytes.

The SunOS port uses termcap because I prefer it. The Solaris port uses terminfo because Solaris doesn't seem to have a termcap library.

Sun's normal C compiler, "cc", only supports the old K&R syntax, not ANSI syntax. Some of your standard (non-elvis) header files may include a "const" declaration which K&R doesn't understand, so if you aren't using GCC then the configure script will append "-Dconst=" to the value of the CC macro in your Makefile, so the "const" keyword will be ignored.

If you configure elvis to support X-windows, and your LD_LIBRARY_PATH environment variable doesn't contain an X11 library directory, then the configure script will output a warning message telling you how to set that variable. However, you might not need to set it; try running elvis without setting it first.

OSF-1

This configuration is rather weird. I don't have access to an OSF-1 system myself, so it's hard for me to make it less weird.

The primary weirdness is that it uses tinytcap.c (by defining NEED_TGETENT in the config.h file) instead of the real terminfo functions. I suggest you try compiling without NEED_TGETENT.

BSD

For BSD, configure will try to use the shlicc2 compiler if it is available, so elvis can use shared libraries.

Interestingly, the BSD port is configured to use POSIX ioctl calls instead of BSD's own sgtty calls. This may be a mistake. If elvis fails to adjust when you resize your xterm, then I suggest you run "configure --ioctl=sgtty", and recompile.

SCO

SCO likes to change the operating system name (as reported by the "uname" command) to match the network node name. If uname reports a brand of Unix that configure doesn't recognize, then configure will inspect the files on your system for evidence that you're running SCO Unix/ODT or SCO Xenix. If that test fails, you'll need to give the OS name on the the command line when invoking configure (e.g., "configure sco" or "configure xenix").

When using an ANSI compiler, SCO seems to require extern declarations of the termcap functions. I added those declarations to the end of the "osunix/osdef.h" file.

Linux

Newer versions of Linux seem to have moved the definition of the speed_t data type from its traditional location, <sys/types.h>. The new location is <termios.h>, which makes sense since that's where the legal values of speed_t are defined... but because it is non-standard, I had to add a little extra code to the "osunix/osdef.h" file to include <termios.h> for Linux.

9.5 Other operating systems

Elvis has been reported to work without modification on **Atari ST/TT systems with MiNT**. The only special trick is that you must configure it with "configure --ioctl=sgtty". Without the "--ioctl=sgtty", elvis would be configured to use the POSIX termios ioctl() calls, which don't work quite right under MiNT.

If you would like to port elvis to another operating system, please contact me first. I can send you some notes I have about how the OS-dependent functions in the os*/*.c files should behave, and also how the GUI functions should behave. That's normally all you need to write in order to port elvis to a new operating system.

A **Macintosh** port is currently underway. I still need volunteers to port it to **VMS** and **OS-9**. I would also welcome a port for any other popular system.