ESCape from the dark

Don't DOS in the dark. Roger Gann puts colour into your screen with help from the ESC character.

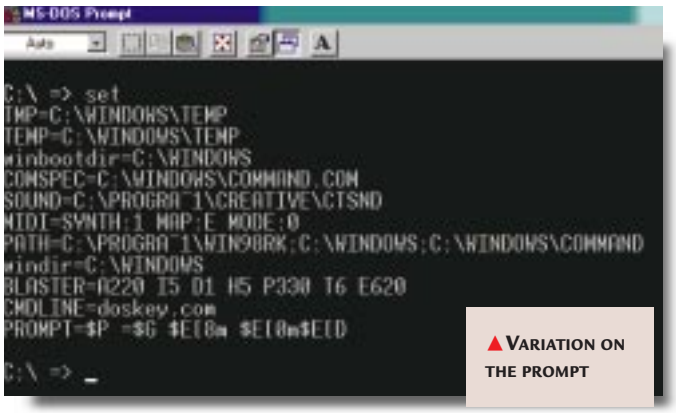
Visually, the DOS environment is pretty dull. For most users it's all black and white, but it's easy to inject some colour into it courtesy of a seldom-used DOS device driver, ANSI.SYS. Even better, this little driver lets you perform other useful tricks, too. ANSI.SYS provides extended screen and keyboard control to any application which uses DOS functions for its input or output. Like other device drivers, it's installed at boot time from the CONFIG.SYS file. The ANSI driver gives access to the ANSI functions only to programs which use DOS services for I/O. For performance reasons most DOS applications by-pass DOS I/O, either by using the BIOS or addressing the hardware directly. However, one program which benefits from ANSI.SYS is the DOS command line interpreter, COMMAND.COM.

ANSI works as a console filter and looks for an 'escape sequence' as its cue to perform a special task. Each ANSI sequence begins with two characters: an ESC followed by [, the left bracket. The ESC character (*not* the letters, E-S-C) stands for the control character, decimal 27 (Hex 1B). ESC is among the 32 characters in the ASCII table which appear before the first text character (the space, character 32). These initial characters are known as control characters as they are used to control devices, like printers. ESC is CTRL-[, often printed as *[. The ASCII on-screen interpretation is a small left arrow.

Thanks to ANSI.SYS you can specify screen colours and resolution, control the cursor, clear the screen and remap specific keys. The ANSI control sequences have the general format:

```
ESC[PARAMETERSCOMMAND
```

The left bracket is the second control character and is followed by the parameters for the specific ANSI command. These parameters are either a decimal number or a literal string.



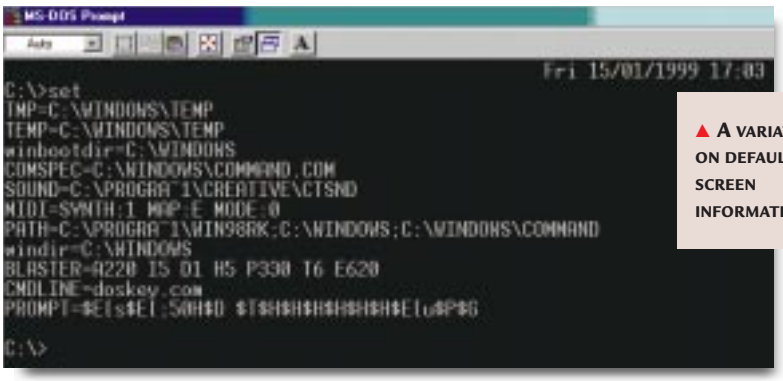
▲ VARIATION ON THE PROMPT

The elusive ESC

There's one small problem with the ESC character. As it is a control character it is not meant to be printed on paper, nor on-screen. You can't just hit an Esc key to generate it. A word processor is no great help here. You need a much simpler text editor for this task.

Here's the trick. Using MS-DOS EDIT, you can generate the elusive ESC character by pressing CTRL-P and then hitting the Esc key. On screen you'll see it represented as a left-pointing arrow.

Die-hard EDLIN users have to employ a different trick: in EDLIN you press CTRL-V and then the left bracket [. This is potentially confusing though, because the good old left bracket already features heavily in ANSI escape codes and it can look like you've got too many brackets in your command string. In this way you can create a simple text file comprising the required escape codes and just use the TYPE command to execute them. However, a simpler way is to use the much-neglected PROMPT command which lets you summon the ESC character from the command line.



▲ A VARIATION ON DEFAULT SCREEN INFORMATION

Customise the prompt

Beginning with MS-DOS 2, Microsoft allowed you to customise the DOS prompt to make it more descriptive. But this goes beyond merely labelling sub-directories, it lets you include all sorts of other information in the prompt as well. The key to changing the prompt is the PROMPT command which can be included in your AUTOEXEC.BAT or simply typed at the command prompt. Its syntax is:

```
PROMPT [PROMPTSTRING]
```

where [promptstring] is the string which defines the appearance of the prompt. This comprises one or more metacharacters, prefixed by a $ sign:

$B	Piping operator (|)
$D	Current date
$E	Escape character
$G	Output redirection operator (>)
$H	Backspace (delete previous character)
$L	Input redirection operator (<)
$N	Current drive
$P	Current drive and directory
$Q	Equal sign (=)
$T	Current time
$V	DOS version number
$$	Dollar sign
$_	Carriage return

The metacharacters can be lower or upper case. So, to have a prompt that

includes the drive and sub-directory details you'd type:

```
PROMPT $P$G <CR>
```

Since MS-DOS 6, this has become the default prompt and it's fine as far as it goes, but thanks to the wide range of prompt metacharacters and some screen manipulation commands that you get when you load ANSI.SYS, you can go completely crazy with the prompt and devise some really fancy ones.

Here's one that's a little over the top. To make it work, be sure that you have the ANSI driver loaded in CONFIG.SYS:

```
DEVICE=C:\DOS\ANSI.SYS
```

Note that because the following prompt string is so long, you may run out of environment space when you try to run it. If you do, add this line to your CONFIG.SYS file:

```
SHELL=C:\DOS\COMMAND.↙
COM /E:1024
```

(↙ code string continues)

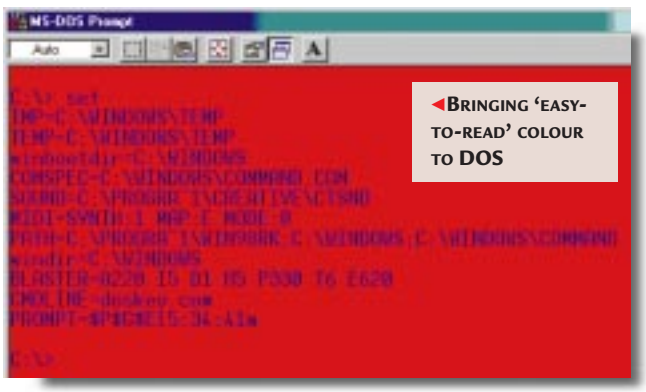
When you reboot, your environment will be expanded from its default of 256 bytes to a more reasonable 1,024.

Then type this in at the prompt or add it to your AUTOEXEC.BAT :

```
PROMPT $E[B$E[s$E[H$E[K$E[1↙
;33;44m$E[K$E[1C$P$E[1;30↙
 $V$E[1;66H$D$E[0;1;37;↙
40m$E[u$G
```

(↙ code string continues)

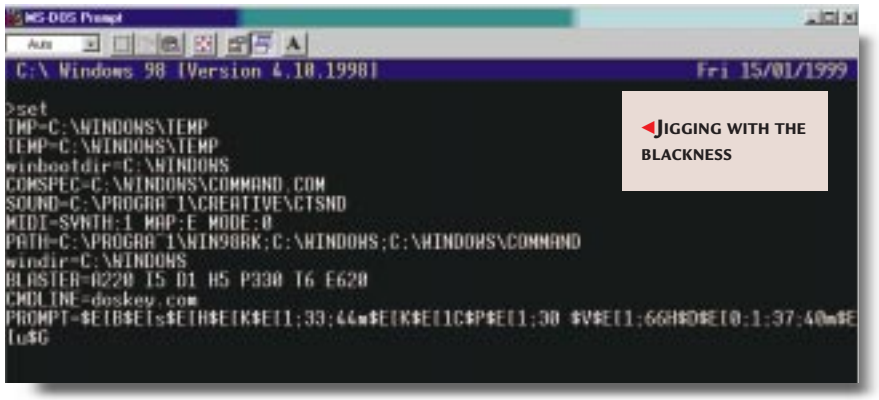
If you type this line in carefully at the prompt you should get a blue bar at the top of your screen. It will contain yellow text, comprising the current drive and directory, the version of DOS you are running, and over on the right it will tell you the day and date. Underneath will be a 'greater than' symbol — your prompt.



BRINGING 'EASY-TO-READ' COLOUR TO DOS

Here are some other prompts you may like to try:

```
PROMPT $P =$G $E[8m $E[0m$E[D
```



JIGGING WITH THE BLACKNESS

```
PROMPT $E[s$E[;50H$D↙
 $T$H$H$H$H$H$H$E[u$P$G
```

(↙ code string continues)

As ever, watch the capitalisation. This kind of jiggery pokery also works fine under Windows 98 as well as MS-DOS 6.2. If you want info on the various keyboard scan codes and other features of ANSI.SYS check out the on-line help for ANSI.SYS under MS-DOS 6.2 or the text file, MSDOSDRV.TXT in the Windows 98 C:\WINDOWS folder.

Colour or black and white?

While we're on the subject of colour, even on a colour screen DOS will default to boring old white text on a black background. But if you have the ANSI.SYS driver installed and send the appropriate escape codes to it, then the CLS command will clear the screen to the colours you selected. So, for example, the command

```
PROMPT $E[1;37;44m
```

displays bright white text on a blue background. Once again, if you use the PROMPT command to change the screen colours, you will need to reset your DOS prompt with another PROMPT command.

Here follows the syntax for changing DOS screen colours:

```
ESC[STYLE;TEXTCOLOUR;↙
BACKCOLOURm
```

(↙ code string continues)

Note that the final m at the end must be in lower case. Simply replace STYLE;TEXTCOLOUR and BACKCOLOUR with three numbers.

- There are three Style options:

0 = Normal prompt
1 = Bold prompt
5 = Blink prompt

- The TextColour variable must be replaced with…

30 = black (grey if bold was set) text.
31 = red text
32 = green text
33 = orange (yellow if bold was set text)
34 = blue text
35 = magenta text
36 = cyan text
37 = white text

…and the BackColour variable must be replaced with:

40 = black background
41 = red background
42 = green background
43 = brown background
44 = blue background
45 = magenta background
46 = cyan background
47 = white background

For example, to get blue blinking text with a red background, the command would be:

```
PROMPT $P$G$E[5;34;41m <CR>
```

Coincidentally, a vigilant reader, 15-year-old Arvinder Sehmi sent me a long email reminding me of the colour options available using ANSI escape codes, together with a little executable he had cobbled together to simplify this task. You'll find COLOUR.ZIP on our CD.

Don't forget that these commands only affect the DOS session. Most DOS apps will re-initialise the screen when they load and will set their own colour scheme, re-imposing the default B&W scheme on exit.

PCW CONTACTS

Roger Gann can be contacted via the PCW editorial office (address, p14) or email 16bit@pcw.co.uk