Command and conquer

Be a macro man. Roger Gann makes use of a powerful DOSKEY feature.

I guess many readers will be familiar with the DOSKEY utility; a little goodie which surfaced with MS-DOS 5.0. It's a memory-resident utility that's a command stacker and macro generator. By default, DOSKEY is set up to record about 35 of the most recently-typed commands but you can increase its storage capacity by changing the buffer size and the commands can be recalled via the cursor keys, which is very useful. However, I guess that while most readers are familiar with DOSKEY's command-stacking capabilities, only a few will be familiar with its macro capabilities which is a shame as this is a very powerful feature and a real time saver.

There are two ways in which you can enable DOSKEY. One is to type DOSKEY at the command prompt. The other is to add a line to your AUTOEXEC.BAT file, which reads:

```
c:\dos\doskey
```

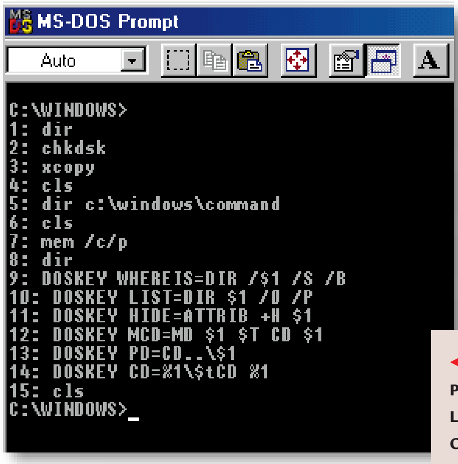
If you use a 386- or 486-based PC and want to load DOSKEY into upper memory, type instead:

```
loadhigh c:\dos\doskey
```

When you first load DOSKEY, it has a buffer size of 512 bytes, with the entire program taking up a mere 4Kb of memory. So, if you do have a 386 or 486, load DOSKEY into upper memory to avoid wasting conventional memory, then expand the buffer to 1Kb, as described below, (it still only takes up 5.5Kb of upper memory) with no loss of conventional memory.

Most basically, DOSKEY allows you to easily recall previous command line entries via the cursor keys. It's a bit like the F3 key which lets you recall the previous command, but this is much better; with DOSKEY, you can hit F7 to select from a list of commands you used in the current session [Fig 1].

There are many ways to recall commands stored in the DOSKEY buffer. Besides pressing the up arrow key to get the previous command, you can use the down arrow key to find the next. Additionally, pressing Page Up or Page Down recalls the oldest or newest command, respectively. Once you have recalled the appropriate one, use your left or right cursor keys to move one character at a time and edit any line as you see fit.



```
C:\WINDOWS>
1: dir
2: chkdsk
3: xcopy
4: cls
5: dir c:\windows\command
6: cls
7: mem /c/p
8: dir
9: DOSKEY WHEREIS=DIR /$1 /S /B
10: DOSKEY LIST=DIR $1 /O /P
11: DOSKEY HIDE=ATTRIB +H $1
12: DOSKEY MCD=MD $1 $T CD $1
13: DOSKEY PD=CD..\$1
14: DOSKEY CD=%1\$tCD %1
15: cls
C:\WINDOWS>_
```

◄ FIG 1 WITH DOSKEY LOADED, PRESSING THE F7 KEY BRINGS UP A LIST OF YOUR PREVIOUSLY ENTERED COMMANDS

The Home and End keys move to the beginning or end of the line. Pressing ESC clears the line, CTRL-left-cursor key moves to the previous word, while CTRL-right cursor moves to the next.

Even better, DOSKEY actually enables you to compose your own custom DOS commands using its macro feature — and DOSKEY is so good that you will want to store plenty of macros. However, its default buffer size of 512 bytes is just not sufficiently large to hold lots of macros so you should install it with a 1Kb buffer size by typing the following command:

```
DOSKEY /BUFSIZE=1024 <CR>
```

Creating a macro with DOSKEY is simple. Just type DOSKEY followed by the macro name, an equal sign and the command(s) to be assigned. You can include more than one command by separating individual commands with the character $T. The symbols $1 and $2 are like the replaceable parameters %1 and %2 in batch files.

DOSKEY substitutes the parameters that you enter on the command line for the replaceable parameters in the macro. It also supports the replaceable parameter $*, which is equivalent to everything entered on the command line after the macro name.

Unlike the size of a batch file, a macro's size is restricted by the maximum length of the command line (127 characters). However, a macro runs much faster than a batch file because DOS doesn't have to pause to load the macro from disk before executing it. In fact, macros behave almost exactly like internal DOS commands. If you give a macro the same name as a built-in command such as DIR or CLS, the macro runs when you type the command at a prompt.

You can combine commands in a DOSKEY macro by separating each with a $T. For example, if you use a hard disk partitioned into six logical drives you could create a macro called CHECKALL which checks and fixes all six drives. The DOSKEY macro would appear as:

```
doskey checkall=chkdsk c:/f↙
 $t chkdsk d:/f $t chkdsk e:↙
/f $t chkdsk f:/f
```

(↙ Code string continues).

To run the macro, type checkall, and it will check and fix the drives C: to F:.

DOSKEY macros are powerful and fast but unless you save them to a file, they are only temporary. That is, they only work until the buffer fills up, or you load Windows, or you reboot. If you only have a few macros just add them to your AUTOEXEC.BAT file. If you have several, though, you may wish to store them all in a batch file and load them on demand.

Here's an easy way to save all your macros in a batch file. Type:

```
doskey/m>macros.bat
```

By itself, DOSKEY/M will list all DOSKEY macros on your screen Fig 2. Adding the redirection symbol, >, will redirect output to a text file called MACROS.BAT.

Next, load the file MACROS.BAT into a text editor such as EDIT and edit each command line to commence with the word DOSKEY. You can then 'call' MACROS.BAT from your AUTOEXEC .BAT file. DOSKEY understands the following special metacharacters in macro definitions:

$B	Piping operator (|)
$G	Output redirection operator (>)
GG	Output redirection append operator (>>)
$L	Input redirection operator (<).
$T	Command separator, CTRL-T ()
$$	Dollar sign
$1 to$9	Replaceable parameters 1 through 9
$*	All command-line parameters.

[FIG 3]

Try these macros

- **Change** to a parallel directory:

```
DOSKEY PD=CD..\$1
```

- **Copy** a file to the printer:

```
DOSKEY FP=COPY $1 PRN
```

- **Count** the number of files:

```
DOSKEY FILES=CHKDSK $1 $B FIND "files"
```

- **Delete** files so they can be easily recovered from a 'trash' directory:

```
DOSKEY DEL=COPY $1 C:\TRASH $T DEL $1
```

- **Delete** multiple files:

```
DOSKEY MDEL=FOR %%F IN ($*) DO DEL %%F
```

- **Delete** system files:

```
DOSKEY ZAP=ATTRIB -H -S -R $1 $T DEL $1
```

- **Display** a sorted, paged DIR:

```
DOSKEY LIST=DIR $1 /O /P
```

- **Hide** files:

```
DOSKEY HIDE=ATTRIB +H $1
```

- **Unhide** files:

```
DOSKEY UNHIDE=ATTRIB -H $1
```

- **Make** and then change to that directory:

```
DOSKEY MCD=MD $1 $T CD $1
```

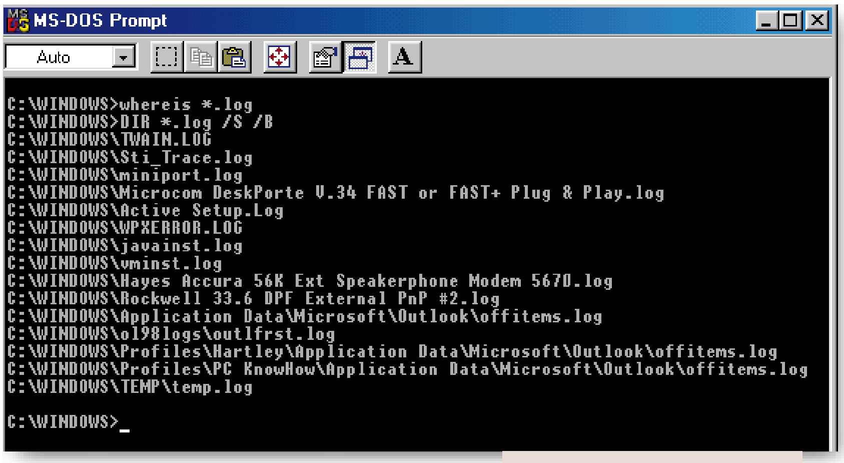
- **Move** a file:

```
DOSKEY moveit=copy $1 $2 $t del $1
```

- **Search** for a file:

```
DOSKEY WHEREIS=DIR $1 /S /B
```

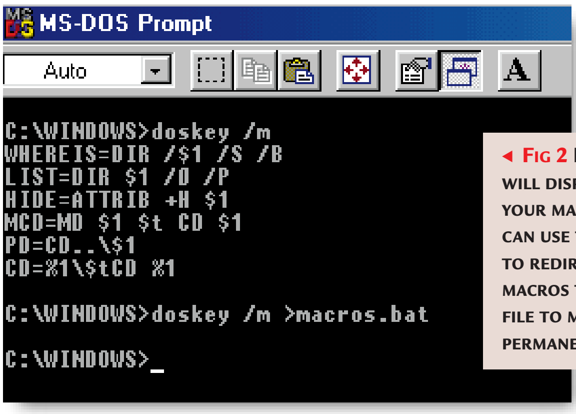
The capabilities of DOSKEY are almost limitless and are dependent largely on your batch-programming skills. Most of the time you can use it to simplify wordy DOS commands. For instance, DoubleSpace /Mount could be abbreviated to DSM or you could redefine CHKDSK so that typing it would actually run CHKDSK/F. Anyway, Fig 3 lists some example macros you might like to try out. In all cases, the actual 'new' DOS command we've created is the word following DOSKEY. Use the new commands as you would an ordinary DOS command: for example,



```
C:\WINDOWS>whereis *.log
C:\WINDOWS>DIR *.log /S /B
C:\WINDOWS\TWAIN.LOG
C:\WINDOWS\Sti_Trace.log
C:\WINDOWS\miniport.log
C:\WINDOWS\Microcom DeskPorte V.34 FAST or FAST+ Plug & Play.log
C:\WINDOWS\Active Setup.Log
C:\WINDOWS\WPXERROR.LOG
C:\WINDOWS\javainst.log
C:\WINDOWS\vminst.log
C:\WINDOWS\Hayes Accura 56K Ext Speakerphone Modem 5670.log
C:\WINDOWS\Rockwell 33.6 DPF External PnP #2.log
C:\WINDOWS\Application Data\Microsoft\Outlook\offitems.log
C:\WINDOWS\ol98logs\outlfrst.log
C:\WINDOWS\Profiles\Hartley\Application Data\Microsoft\Outlook\offitems.log
C:\WINDOWS\Profiles\PC KnowHow\Application Data\Microsoft\Outlook\offitems.log
C:\WINDOWS\TEMP\temp.log

C:\WINDOWS>_
```

▲ Fig 4 A macro in action. The Whereis macro lists every occurrence of a file



```
C:\WINDOWS>doskey /m
WHEREIS=DIR /$1 /S /B
LIST=DIR $1 /O /P
HIDE=ATTRIB +H $1
MCD=MD $1 $t CD $1
PD=CD..\$1
CD=%1\$tCD %1

C:\WINDOWS>doskey /m >macros.bat

C:\WINDOWS>_
```

◄ Fig 2 DOSKEY/M will display a list of your macros. You can use this feature to redirect the macros to a batch file to make them permanent

WHEREIS AUTOEXEC.* will search your hard disk for all files that start AUTOEXEC Fig 4.

DOSKEY is useful under Windows 98, too. To save yourself the hassle of having to run it manually each time you open a DOS session, right-click

```
DOSKEY CD=%1\$tCD %1
```

This DOSKEY macro, which replaces the regular CD command, changes drive and directory automatically. When you use it with a relative path name (CD SYSTEM, for instance, when the current directory is C:\Windows), you get a Bad Command or File Name message but the macro still changes to the correct directory. If you type a space before this or any other DOSKEY macro which replaces a DOS command, you get the old command rather than the macro.

A final tip

DOSKEY insists that you type DOSKEY /MACROS to get a list of macros but, as we have seen in the past, it is no big deal to reassign function keys. So, it is a piece of cake to get the F10 key to do this for us. With the ANSI.SYS device driver loaded, just enter the following command, either in your AUTOEXEC.BAT file or at a command prompt:

```
PROMPT $E[0;68;"DOSKEY /↙
MACROS";13p
```

(Key: ↙ code string continues).

PCW CONTACTS

Roger Gann welcomes your comments about the 16-bit column. He can be contacted via the PCW editorial office (address, p14) or email 16bit@pcw.co.uk