NT INSIGHTS

Take Command

BY GUY C. YOST

Past columns have focused on NT's inherent abilities as a network operating system and an operating system. Starting this issue, I'll be examining add-on products that I've found to be very useful in managing NT servers and workstations. If you've found products that have increased your effectiveness in managing NT servers, I'd love to hear about them. I'll feature your comments and expertise in an upcoming column.

I'm particularly interested in products designed to alleviate routine administrative tasks across large server installations in enterprise environments. Please feel free to forward your suggestions to me at gyost@logical.net. I'll need to obtain an evaluation or demo copy of the software to test, so if you know of an FTP server, URL, or sales contact I can use to obtain the product, please include that information as well. Your help and participation in this new direction will add a lot of value to ***Technical Support*** and will be appreciated by other readers.

TAKE COMMAND

Using NT as a "widget" server has become a cost-effective alternative to using UNIX; however, I'll be the first to acknowledge that NT's abilities comprise a subset of what most flavors of UNIX have to offer. When veteran UNIX users are introduced to NT solutions, most are quick to point out NT's lack of support for versatile command line scripting. The ability to run a series of automated tasks and perform useful functions with files, strings, arithmetic expressions, and system environment variables has been lacking not only in NT's inherent ability, but also with Windows 95. For those versed in the C programming language, there's always the option of compiling your own custom programs or using interpretive PERL. However, for non-programmers and programmers alike, it would be nice to have a way to quickly produce scripts that can perform functions not found by NT's inherent abilities.

The idea behind augmenting NT's command line is to allow power users and administrators to create scripts that can process a series of commands while incorporating advanced functions.

This month, I'll examine a popular NT and Windows 95/98 command line enhancement utility called "Take Command" that is based on JP Software's 4DOS and 4NT utilities but adds a useful GUI-based command window with powerful drop-down menus. Take Command 32 (TCMD32), examined here, is designed specifically to run on NT and 95/98 workstations; however, Take Command 16 is available for Windows 3.1.

The idea behind augmenting NT's command line is to allow power users and administrators to create scripts that can process a series of commands while incorporating advanced functions. Take Command files can include subroutines, FOR and WHILE loops, IF/THEN/ELSE logic, and can prompt for user input. TCMD32 files can run in single-step mode so that debugging complex routines is easy. You can even view changes to environment variables and other information as the batch file executes.

Take Command installs in a non-intrusive manner. My first concern in using TCMD32 was that it would replace the default NT command interpreter on the system in general. Instead, the regular DOS (window) prompt is left alone and the TCMD32 command interpreter is used only in the Take Command window. You can opt to have a few registry entries made that will allow Take Command batch files (BTM extension) to be registered and recognized by Window's Explorer. To do so, open BTM files with Take Command using the right mouse button and add a "Take Command Prompt Here" option to the context menu in any directory.

The TCM32 engine supports alias (short nicknames for long commands) and all standard DOS-like functions while enhancing parameter options. For example, Take Command's DIR, DEL, and COPY commands support numerous options, allowing advanced selection criteria for the command operation. But the real power in Take Command lies in its supported functions. There are too many options to discuss in this column, however, the program comes with an EXAMPLES.BTM that demonstrates the utility's abilities.

JP Software Inc.
P.O. Box 1470
East Arlington, MA 02474
781/646-3975 • 781/646-0904 Fax
Order Line: 800/368-8777
(U.S./Canada, orders only)
Internet: www.jpsoft.com
For direct FTP access connect to ftp.std.com and look in the /vendors/jpsoft directory and its subdirectories
Site Licensing Information: sales@jpsoft.com

Figure 1: BTM File

```
REM TEST1.BTM, a short demonstration batch file for Take Command 32 for NT/98/95

@echo off
set i=data.txt

REM  Display Contents of DATA.TXT
type %i

REM  Open file for READ in Text mode as file handle=f1
set f1=%@fileopen[%i,read,t]

REM  Read in the first line of the file, and assign string to variable 'n'
REM  We'll use 'n' later to demonstrate output formatting.
set  n=%@fileread[%f1]

REM  Move read pointer 8 characters from beginning of second Line
REM  and read in 10 characters. 'Fileseek' returns an integer
REM  value of position reached in file
echo File position=%@fileseek[%f1,8,1], First read result=%@fileread[%f1,10]

REM  Move another 7 characters, and read in 8 more characters
echo File position=%@fileseek[%f1,7,1], Second read result=%@fileread[%f1,8]

REM  Read in and display the next 11 chars:
echo Third read result=%@fileread[%f1,11]

REM  Create a new file named "out.txt" in the current directory
set  j=%@filename["out.txt"]

REM  Assign file handle to OUT.TXT so we can write to it
set  f2=%@fileopen[%j,write,text]

REM  From the current pointer position, read in 40 more characters from f1
REM  and assign result to new variable 's'.
set  s=%@fileread[%f1,40]

REM  Write the value of 's' to OUT.TXT (f2).
set  i=%@filewrite[%f2,%s]

REM  Check 'i' (operation return code) for errors. This code could be repeated for other
REM  file operations that use 'i' as the return code in this example.
iff  %i == "-1"  then
  echo Operation Failed!
  quit
else
  echo Operation Successful!
endiff

REM  Write the value of 's' to OUT.TXT again, but this time with all 'o's changed to '!'
set  i=%@filewrite[%f2,%@replace[o,!,%s]]

REM  Set variable 't' to the number of lines in this batch file, and 'v' to
REM  the number of characters in string variable 's'. We'll use 't' and 'v'
REM  to demonstrate the mathematical 'eval' function.
set  t=%@lines[%_batchname]
set  v=%@len[%s]
echo t=%t v=%v

REM  Write to OUT.TXT the result of 'v' divided by 't' and format the output so that the
REM  result is right justified by 20 characters. Note the use of functions within functions.
set  i=%@filewrite[%f2,Result of 'v'/ 't' =%@format[20,%@eval[%v/%t]]]

REM  Close handle for files f1 and f2
set  i=%@fileclose[%f1]
set  i=%@fileclose[%f2]

REM  Display the contents of OUT.TXT and the value of 'n' to verify output formatting
type %j
echo %n
```

Batch File Results:

```
12345678901234567890123456789012345678901234567890123456789O
A quick brown fox jumped over the lazy dogs.
Now is the time for all good Men to come to the aid of their country.

File position=70, First read result=brown fox
File position=87, Second read result=over the
Third read result= lazy dogs.
Operation Successful!
t=64 v=40

Now is the time for all good Men to co

N!w is the time f!r all g!!d Men t! c!
Result of 'v'/ 't' =               0.625
12345678901234567890123456789012345678901234567890123456789O
```

Figure 1 presents a sample BTM file I created to demonstrate a variety of file, string, format, and mathematical functions. Each section is commented, so figuring out what is happening in each section should be easy. Note how this example can open a file, read the contents of that file, perform math and string functions on the input read in, then create another file containing the program results. Try that with a standard batch file.

SUMMARY

The applications for Take Command are countless, and if you've ever encountered a situation where NT's or 95's command line abilities have left you cold, you should test drive this utility. You can obtain a fully functional 21-day evaluation from www.jpsoft.com or www.shareware.com. If you find TC 32 as useful as I did, then a supported, licensed copy can be obtained for $69.95.

NaSPA member Guy C. Yost is the owner of Redstone Consulting, an IT management consulting firm in New York. He has authored several books on networking for Que Publishing, including Learning NetWare 4.1, and NetWare 4.1 SmartStart, and contributes to Technical Support *magazine as an author, columnist and technical editor. Guy also develops and conducts seminars on networking with Windows NT, UNIX, NetWare and Internet/intranet technologies across the United States and Canada. He can be reached at (518) 674-5606 or gyost@logical.net.*