2 Permissions

SOS Utilities is a collection of small useful utilities written in REXX, the OS/2 batch file processing language. Also included is SOS Tree, a utility for displaying the directory tree.

3 ChName—file name modifier

ChName changes long HPFS file names to FAT format and vice versa. It is a useful tool when copying files to or from diskettes.

3.1 ChName usage

```
[D:\] chname FILE ... [ /f /h /q /x /? ]
```

‘/f’ HPFS to FAT (default)

‘/h’ FAT to HPFS (ie. reverse what was done with /f)

‘/q’ Quiet mode

‘/x’ Send output to a REXX queue instead of standard output

‘/?’ Show help page and quit

3.2 How it’s done

When changing HPFS names to FAT, the program first simply tries to truncate the name to the ‘????????.???’ mask. Spaces are converted to underlines, and the first and the last dot-separated part are put together. For example,

```
"What ever happened to Baby Jane.txt.Z" -> WHAT_EVE.Z
```

If there already is a file with the same name, a part of the resulting name is substituted with a number, eg. ‘WHAT_EV0.Z’. Only as many letters are substituted as necessary—first one number is tried (‘0...9’), then two numbers (‘00...99’) and so on. Eventually a unique file name is very probably found.

The long HPFS file name is saved to the ‘.LONGNAME’ extended attribute so that the name can later be restored with the ‘/h’ switch, eg.

```
WHAT_E00.Z -> "What ever happened to Baby Jane.txt.Z"
```

3.3 Using REXX queues

The ‘/x’ switch is designed to be a helpful addition for REXX programmers. With it, the output of ChName (ie. the old file names, whether long or short, and corresponding new names) can be easily fetched. For example, the following program saves all the long names to long_name stem variable, and corresponding short names to short_name stem.

```
/* example.cmd */

"@echo off"
Parse Arg list

n = 1
Do i = 1 To Words(list)
    file = Word(list, i)
    Call ChName file "/f /x"
    Do While Queued() > 0
        Parse Pull long " -> " short
        long_name.n = long
        short_name.n = short
        n = n + 1
    End
End
```

3.4 Known bugs

Spaces in file names cannot be handled from the command line, ie. you cannot use `ChName` like this:

```
[D:\] chname "What really happened to Baby Jane.txt.Z"
```

Because OS/2 seems to strip the quotes before anything reaches REXX, `ChName` gives you a list of silly messages about files not found:

```
ChName v1.1: File 'What' not found
ChName v1.1: File 'really' not found
...
```

But spaces **are** still allowed in file names, so the following is possible even if the '`*.txt.Z`' mask matched also files with spaces in their names:

```
[D:\] chname *.txt.Z
```

4 Extract

Extract—extract files from archives or list archive contents. A REXX script.

Note: Extract only chooses the appropriate archiver and forms the command line which is passed to it; you will still need all the archivers whose archive files you encounter. However, Extract makes working with the archivers much more comfortable since you don't need to remember every switch of every program (or indeed, remember the actual executable names of the archives). It also makes it possible to put the archivers to a directory not contained in 'PATH'—just edit the line arcdir=... in 'extract.cmd' to point to your archiver directory.

4.1 Extract usage

```
[D:\] extract ARCHIVE [ FILE ... ] [ /jlp? /t DIR ]
```

'/j' Junk directories (ie. don't extract them). With this switch, files are extracted to the current directory, or the directory given with '/t' switch. Can't be used together with '/l' or '/p'.

'/l' Show archive contents. Output goes to stdout. Can't be used together with '/l', '/p', or '/t'.

'/p' Extract to stdout. Useful in viewing document files stored in the archive.

'/t *DIR*' Set directory in which files are extracted. Without the '/j' switches, any subdirectories contained in the archive file will be created under *DIR*. The space between the switch and the directory name isn't necessary.

'/?' Display help page and quit.

4.2 Examples of Extract usage

Suppose you have a file 'foo.zoo', which includes the following files (and subdirectories):[1]

```
src\foo.c
src\foo.h
src\foo.mak
exe\foo.exe
man\foo.man
README
Copyright
```

To extract all the files, use

```
[D:\] extract foo.zoo
```

To extract all the files in 'src' subdirectory (ie. 'foo.c', 'foo.h', and 'foo.mak'):

```
[D:\] extract foo.zoo src\*
```

The same as above, but do not create 'src' subdirectory, ie. extract files to the current directory:

```
[D:\] extract foo.zoo src\* /j
```

[1] Zoo, Zip, gzip, tar and Compress use slashes instead of backslashes internally to separate directories, but in the command line of Extract you should use backslashes—otherwise you'll get silly messages of unknown switches. . .

The same as above, but extract to 'C:\tmp\thrash' directory:

```
          [D:\] extract foo.zoo src\* /j /t C:\tmp\thrash
```

To list the contents of this archive, use:

```
          [D:\] extract foo.zoo /l
```

To view the 'README' file with 'more', use:

```
          [D:\] extract foo.zoo README /p |more
```

To redirect 'README' to printer:

```
          [D:\] extract foo.zoo README /p >prn
```

5 FileFind

FileFind—whereis-like find file utility written completely in REXX.

5.1 FileFind usage

```
[D:\] filefind [PATH\]FILENAME ... [ /h /n /? ]
```

FileFind finds files under *PATH* if that is specified, or the current directory (= '.') if it isn't. Multiple command line arguments are allowed.

'/h'	find hidden and system files too
'/n'	brief format: display only file names
'/?'	display help page and quit

5.2 Examples of FileFind usage

Note the prompt (ie. current directory info) in the following examples!

Find all '*.inf' files under 'D:\OS2'

```
[D:\OS2] filefind *.inf
```

The same as preceding, but note that the current directory is now different:

```
[D:\] filefind \OS2\*.inf
```

Find all '*.exe' files under 'C:\Bin'

```
[D:\] filefind C:\Bin\*.exe
```

Find all '*.inf' files under 'D:\OS2\DLL' and 'D:\os2\apps'—this demonstrates multiple command line arguments

```
[D:\OS2] filefind DLL\*.inf apps\*.inf
```

5.3 Warning

FileFind displays file dates and times in the national format, which is found with a SysIni() call to 'os2.ini'. However, this seems to be causing troubles every now and then—sometimes SysIni() returns insane results, like '0' as the date or time separator. **If you are having problems with this**, remove all date and time formatting lines from the program, and change the following line:

```
ok = SysFileTree(Word(find, i), files, "fst", "*--*-")
```

to

```
ok = SysFileTree(Word(find, i), files, "fs", "*--*-")
```

FileFind will then display dates and times in the American formats ('MM/DD/YY' and 'HH:MM{am|pm}', respectively) which are insane but work fine.

6 kill—kill processes

kill—kill processes by their process id's (PID).

6.1 kill usage

```
[D:\] kill PID ...
```

Process id's are given in decimals, not hexadecimals. To get decimal process id's, use ps (See Chapter 7 [ps], page 8) instead of the OS/2 system utility PSTAT (well, actually ps runs PSTAT).

6.2 System requirements

kill is a Perl program and requires you to have Perl installed. Perl is a programming language which originally was designed for UNIX (how else) but is now ported for many other operating systems. Perl should be find by anonymous FTP'ing to one of the following places:

```
hobbes.nmsu.edu
ftp.cdrom.com
ftp.luth.se
ftp.informatik.tu-muenchen.de
ftp.urz.uni-heidelberg.de
```

and possibly many others. Try directories like

```
.../os2/2_x/programming                or
.../os2/2_x/unix/programming
```

and search for files starting 'perl'.

6.3 Tips

If you are using an aliasing shell like 4OS2, this script should be easy to convert to an alias. For example, when using 4OS2, put the line

```
KILL=perl -e "kill(9, %$);"
```

in your alias file, or type

```
[D:\] alias kill=`perl -e "kill(9, %$);"`
```

at the command line.

I haven't yet figured out how command line can be passed to sh (ms_sh 2.1C). The easiest solution (using '$*') doesn't seem to work.

7 ps—process status

`ps`—display processes with their process id's (PID), parent process id's (PPID), and session id's (SID).

7.1 ps usage

```
[D:\] ps
```

`ps` output looks like this:

```
PID  PPID   SID  PROCESS
 3     1     1   d:\os2\pmshell.exe
 7     3    18   d:\bin\pc2\pc2.exe
 6     3    17   d:\bin\xfeel.exe
 5     3    16   d:\os2\pmshell.exe
 4     3     0   d:\os2\system\harderr.exe
```

All the numeric values are decimals. If you want hexadecimals, just remove all calls to `Format()` and `X2D()` in lines 24–26 of '`ps.cmd`'; ie. change

```
pid = Format(X2D(Word(line, 1)), 4)
```

to

```
pid = Word(line, 1)
```

You may have to reformat the header line to get the columns right.

7.2 System requirements

`ps` is a REXX program and requires you to have REXX installed. Also, `ps` calls `PSTAT` to obtain the information, and only re-formats the output to a more readable and at least shorter format. `PSTAT` is part of the OS/2 Diagnostic Tools installation options; if you don't have the file '`pstat.exe`' in your '`\OS2`' directory, use the 'Selective Install' object in your System folder to install it. REXX can be installed in the same way, in the unlikely case you haven't it installed already.

'`rxqueue.exe`' should also be found in the '`\OS2`' directory. It is part of the REXX installation option.

7.3 Implementation notes

It took a long while before I found out how to pipe output to REXX routines inside a REXX program. Maybe there are even better altern- atives than the `rxqueue` method this program uses, but none of them is documented in the standard OS/2 package. I really wish IBM would some day give a full, decent documentation **free** with the OS/2 system...

8 SOS Tree

SOS Tree—directory tree drawer utility for OS/2.

8.1 SOS Tree usage

```
[D:\] sostree [ -adfnqz ] PATH
```

'-a' Display file attributes in the format 'ADHRS'

'-d' Display dates and times too (in current national format).

'-f' Include plain files too in the listing.

'-n' List all directories (or all files and directories with -f option) with full path names; do not draw the tree graph.

'-q' Don't display the 'Directory tree of ...' header.

'-z' Display file sizes too. This switch is meaningless if '-f' isn't used simultaneously.

8.2 Note of re-compiling SOS Tree

I have tried to make SOS Tree as easy to be translated to other languages as possible, but without the IBM mkmsgf utility I'm forced to use different methods than the handy message file system OS/2 offers. However, it should still be easy to make other-language versions of the program. From version 1.0a onwards, after the standard '.h' files have been included, there are conditional #include clauses (lines 24- in 'sostree.c'):

```
#if defined(LANG_SUOMI)
#include "msg_suomi.inc"
...
#else
#include "msg_english.inc"      /* The default */
#endif
```

To get the Finnish messages, compile with 'LANG_SUOMI' symbolic constant defined. For example, using Borland C++:

```
[D:\] bcc -DLANG_SUOMI sostree.c
```

To create a custom message file, examine 'msg_english.inc' or for an example, and then add an appropriate #elsif clause to 'sostree.c':

```
#if defined(LANG_SUOMI)
#include "msg_suomi.h"
#elsif defined(LANG_ESPERANTO)
#include "msg_esperanto.h"
...
#else
#include "msg_english.h"
#endif
```

No other changes to 'sostree.c' are necessary. To compile this new version, you need to define 'LANG_ESPERANTO' constant in the command line of the compiler, of course.

NOTE: please send your translations to me so that I can include them in the next 'official' release of SOS Tree.

9 Version and copyrights

SOS Utilities version 1.2

SOS Tree v1.0a—directory drawer
ChName v1.1b—change names to/from FAT format
Extract v1.1—archive extractor/lister
FindFile v1.4a—file finder
kill v1.0–kill process
ps v4.0a—display process status
Copyright © 1993, SuperOscar Softwares, Tommi Nieminen.