GXSCR$ - Scratch Directory on GX

The GXSCR$ routine can be used to scratch a directory on the PC that is running the current GX session.

1. Invocation

To scratch a directory on the PC that is running GX code:

```
      CALL GXSCR$ USING dir_name [sc]
```

where *dir_name* is a zero-terminated string which specifies the directory to be deleted (the trailing back-slash is optional); and *sc* is an optional control block of the following structure:

```
      01    SC
        02  SCVERS      PIC 9(2) COMP         * Version number
                        VALUE 1               * Must be 1
        02  SCATTR      PIC 9(4) COMP         * File attribute
                                              * RESERVED FOR FUTURE USE
        02  SCDATE      PIC 9(2) COMP         * Check file date flag
                                              * 0 = No date/time checking
                                              * 1 = Check creation date
                                              * 2 = Check last access date
                                              * 3 = Check last write date
        02  SCDAYS      PIC 9(4) COMP         * Timespan of file in days (only if
                                              * SCDATE is nonzero)
```

The SCATTR field is **reserved for future use** and will be used to specify the attribute of the files that will be deleted.

GXSCR$ will only delete write-able files with either the "normal" or "archive" attributes. In particular, files with the following attributes will **not** be deleted by GXSCR$:

- compressed files
- hidden files
- read-only files
- directories;
- off-line files;
- system files;
- temporary files.

If SCDATE is 0, **all** "normal" or "archive" files on the specified directory will be deleted.

If the SCDATE field is set to either 1, 2 or 3 the File Creation date (SCDATE = 1), Last Access date (SCDATE = 2) or Last Write date (SCDATE = 3) of any delete-able files will be considered. The SCDAYS field must contain a number of days. Only files that have a "time-span" (measured by the Creation Date, Last Access Date or Last Write Date) longer than the value of SCDAYS will be deleted. For example, if GXCSR$ is being used at 13:12:45 on 3-May-2002 and SCDATE contains 1 and SCDAYS contains 2:

File Creation Date		*Action*
13:12:44	30-Apr-2002	Deleted
13:12:44	1-May-2002	Deleted
13:12:45	1-May-2002	Deleted
13:12:46	1-May-2002	Not deleted

13:12:45	2-May-2002	Not deleted
13:12:45	4-May-2002	Not deleted

2. STOP Codes and Exception Conditions

The following STOP codes may be generated by GXSCR$:

STOP code	Description
23919	GXSCR$ has been called by an application that is not running on GX.
23920	Unknown version of SC control block (i.e. SCVERS does not contain 1)
23921	An internal error has occurred (the file name was longer than 8Kb)
23922	An attempt was made to call GXSCR$ on an incompatible version of GX. The version of GX must be V2.6, or later.
23923	An unexpected error has been reported by GX.

The following exception conditions may be returned by GXSCR$:

EXIT code	$$COND	Description
23919	19	An internal error has occurred (GXSCR$ was unable to allocate an additional temporary data page)
23920	20	The replaceable file parameter is not present in the GX.INI file.
23921	21	The Windows DeleteFile (or FindFirst or FindNext) operation failed on more, or more, files. The Windows 32-bit error code from the first failed DeleteFile etc. operation is available in the $$CRES System Variable. This should be considered a "soft error"
No unique error code returned		No files with the specified attribute were found on the specified directory. This should be considered a "soft error"

3. Programming Notes

GXSCR$ is only available when running on GX. Any attempt to use GXSCR$ on a non-GX terminal will result in a STOP code.

Because Windows does not support a "Scratch Directory" operation, GXSCR$ is implemented by individually deleting all the **files** on the directory specified.

Important Note: The GXSCR$ function will **NOT** affect any sub-directories within the specified directory; or any files within those sub-directories.

If GXSCR$ returns an Exception 4 the 32-bit Windows error code returned from the first failing DeleteFile operation is available in $$CRES. The FMESS$ sub-routine may be used to convert the error code to a verbose error message. However, the name(s) of the file(s) that could not be deleted are NOT returned by GXSCR$.

The file name parameter can specify either:

- an absolute directory name (on the filing system of the PC that is running GX);
- an absolute UNC directory name on a shared directory (as "seen" from the PC that is running GX);
- a directory on the current directory on the PC that is running GX;
- a directory on the Windows directory on the PC that is running GX;
- a directory on the LocalTempDirectory specified in the GX.INI file;
- a file on, or under, the folder "PFPrintImage", relative to the current GX folder;
- a file on, or under, the folder "XML", relative to the current GX folder;
- a directory on a free-format directory keyword specified in the GX.INI file;
- a directory on a free-format directory keyword specified as a Windows environment variable;

For example assuming that the "current directory" for GX is "C:\GX"; the "Windows directory" is "C:\Windows"; the GX.INI file contains the following setting in the [general] section:

```
LocalTempDirectory=C:\TEMPORARY
```

and the following settings in the [folders] section:

```
DTEMP1=D:\TEMPORARY
DTEMP2=F:\TEMPORARY
```

And the following Windows environment variable is defined:

```
DTEMP3=G:\TEMPORARY
```

then the following *dir_name* parameters will be mapped to the following pathnames on the PC running GX:

***Dir_name* parameter**	**Filename on PC running GX.EXE**
C:\WINDOWS\TEMP\	C:\WINDOWS\TEMP\
C:\WINDOWS\TEMP	C:\WINDOWS\TEMP\
\\SERVER1\SHARE\TEMP	\\SERVER1\SHARE\TEMP\
TEMP	C:\GX\TEMP\
.\TEMP	C:\GX\TEMP\
SUBDIR\TEMP	C:\GX\SUBDIR\TEMP\
%TEMP\TEMP2	C:\TEMPORARY\TEMP2\
%WINDOWS\TEMP	C:\WINDOWS\TEMP\
%DTEMP1\TEMP	D:\TEMPORARY\TEMP\
%DTEMP2\TEMP	F:\TEMPORARY\TEMP\
%DTEMP3\TEMP	G:\TEMPORARY\TEMP\
C:\%DTEMP3\TEMP	The results of including a variable parameter in the middle of a *dir_name* string are undefined.
%WINDOWS	Invalid attempt to scratch a logically protected directory
.	Invalid attempt to scratch a logically protected directory

If a %variable exists both as a Windows environment variable **and** as an option in the [folders] section of the GX.INI the GX.INI file setting will be used.

The %TEMP and %WINDOWS strings are effectively reserved words and cannot be used to define either Windows environments variables or options in the [folders] section of the GX.INI file. The %TEMP string is mapped to the value of the LocalTempDirectory setting in the [general] section of the GX.INI file. If the LocalTempDirectory setting is not defined, the %TEMP string is mapped to the Windows temp directory. The %WINDOWS string is always mapped directly to the Windows folder.

The %XML and %TCPIIMAGES strings are also reserved words and cannot be used to define either Windows environments variables or options in the [folders] section of the GX.INI file. Both these special strings are reserved for internal use only. The %XML string is always mapped directly to the folder "XML", relative to the current GX folder. The %TCPIIMAGES string is always mapped directly to the folder "PFPrintImage", relative to the current GX folder;

If the directory specified by *dir_name* does not exist an exception condition 21 with a $$CRES value of 2 will be returned by GXSCR$.

4. Examples

[EXAMPLES REQUIRED]

5. Copy-Books

None.

6. See Also

GXDEL$	Delete Windows file on GX (passing FD parameter)
GXDEX$	Delete Windows file on GX (passing filename)