The library SysLibFileStream.lib

This library provides functions which correspond to ANSI C functions for file stream operations. The execution is synchronous.

To enable the functions contained in Version SysLibFileStream23.lib, the runtime system has to be provided with the additional component SysFileStream.

The functions:

SysLibFileStream function	***ANSI C function***	***Data type***	***Description***
SysFileStreamFOpen	*fopen(char *filename, char *mode);	DWORD	File with name *filename* will be opened as stream; possible values for inputvariable *Mode*: 'w' (write), 'r' (read), 'a' (append), '+', 'b', 't'
SysFileStreamClearerr	clearerr(FILE* pFile);	DINT	internal error state of pFile will be deleted; always returns 1
SysFileStreamFClose	fclose(FILE *pFile);	DINT	all open streams will be closed (except for *stdin*, *stdout*, *stderr*). Returns SysFileStreamFClose_EOF in case of error, otherwise 0.
SysFileStreamFEOF	*feof(FILE* pFile);	DINT	returns !=0, as soon as end of file in *pFile* is reached
SysFileStreamFError	ferror(FILE* pFile);	DINT	returns !=0, as soon as an error has been detected for *pFile*
SysFileStreamFFlush	fflush(FILE *pFile);	DINT	Characters which are still buffered internally, will be output
SysFileStreamRemove	remove(char* filename);	BOOL	File will be deleted; returns 1 for OK, 0 in case of an error
SysFileStreamRename	rename(char* filename);	BOOL	Renaming a file; returns 1 for OK, 0 in case of an error
SysFileStreamRewind	rewind(FILE* pFile);	DINT	sets file position to start and deletes internal error state; always returns 1
SysFileStreamFGetC	fgetc(FILE *pFile);	DINT	returns the next character in the stream (0--255, SYSFILESTREAM_EOF in case of an error
SysFileStreamFGetPos	*fgetpos(FILE* pFile, fpos_t * ptr);	DINT	writes current file position of *pFile* to *ptr; fpos*_t here defined as an unsigned long (32 bits)
SysFileStreamFSetPos	fsetpos(FILE* pFile, fpos_t * ptr);	DINT	sets file position of *pFile* according to *ptr*; *fpos_t* here is defined as unsigned long (32 bits); pFPos:DWORD; (* pointer !!*)
SysFileStreamFGetS	* fgets(char * str, int n, FILE * pFile);	POINTER TO STRING	Reads at most the next n-1 characters into the array *s*, (termination automatically with 0); Truncation at '\n', the '\n' will be taken over to *s*; Return value: *s* resp. 0 (at end of file or error)
SysFileStreamFPrintf_Int	fprintf(FILE* pFile, char* szFormat, intnArg);	DINT	formatted output in stream *pFile*; Restrictions compared to C:only 1argument of type INT/DINT etc. can be printed; *szFormat* should be e.g. '%d'

SysLibFileStream function	***ANSI C function***	***Data type***	***Description***
SysFileStreamFPrintf_Real	fprintf(FILE* pFile, char* szFormat, float fArg);	DINT	formatted output in stream *pFile*; Restrictions compared to C:only 1argument of type REAL etc. can be printed; *szFormat* should be e.g. '%f'
SysFileStreamFPrintf_String	fprintf(FILE* pFile, char* szFormat, char *pcArg);	DINT	formatted output in stream *pFile*; Restrictions compared to C:only 1argument of type STRING etc. can be printed; *szFormat* should be e.g. '%s'
SysFileStreamFPutC	fputc(int c, FILE *pFile);	DINT	Writing character (unsignedchar) *c* to stream *pFile* Returns *c* (converted to DINT) or SYSFILESTREAM_EOF in case of an error
SysFileStreamFPutS	fputs(char* str, FILE * pFile);	DINT	Writing string s in stream *pFile* Returns str (pointer to string) or SYSFILESTREAM_EOF in case of an error
SysFileStreamFRead	fread(void* ptr, size_t size, size_t nobj, FILE* pFile);	DWORD	nobj objects of size *size* will be read from pFile to ptr; Returns number of read objects
SysFileStreamFWrite	fwrite(void* ptr, size_t size, size_t nobj, FILE* pFile);	DWORD	nobj objects of size *size* wil lbe written from ptr to *pFile*; Returns number of written objects
SysFileStreamFScanf_Int	fscanf(FILE* pFile, char* szFormat, int * pnArg);	DINT	formatted input from stream pFile; Restrictions compared to C: only 1 DINT argument can be read; *szFormat* should be e.g. '%d'
SysFileStreamFScanf_String	fscanf(FILE* pFile, char* szFormat, char *pcArg);	DINT	formatted input from stream pFile; Restrictions compared to C: only 1 STRING argument can be read; *szFormat* should be e.g. '%s'
SysFileStreamFScanf_Real	fscanf(FILE* pFile, char* szFormat, float* pfArg);	DINT	formatted input from stream pFile; Restrictions compared to C: only 1 REAL argument can be read; *szFormat* should be e.g. '%f'
SysFileStreamFSeek	fseek(FILE* pFile, long offset, int origin);	DINT	sets file position on *offset* Bytes based on *origin*; values for *origin*: SEEK_SET=Start of file, SEEK_CUR=current position; SEEK_END=End of file; 0=OK
SysFileStreamFTell	ftell(FILE* pFile);	DINT	returns current file position (based on file start) in Bytes (-1 in case of error)