embedded-os.de
a little world of RTOS and data-communication protocols

pC/TFS Reference

V2.38b

The Tiny-File-System is based on hierarchical, double-interlinked lists, through what no limitation is available in number of the files within a table or file size (max 4GB). Each file can produce on that occasion in fragmented or none-fragmented form, administers and is deleted.
This file-system handles all names of directories/files as a STRING, so can used special characters in the names. Not to use are two points in series and the slashes.

reserved names-elements:

.. - one directory back
/ - on the beginning of the Path: for from ROOT
/ - in middle of the Path: as separation for directory/file-names

The whole file-system works case-sensitive !
But no mechanisms of the File-sharings are deposited. So a file can be opened simultaneously from many Users to reading but only from one User to writing.
As directory and File-Attribute, Read-Only and Hidden(System) to the disposal, stand. Hidden entries are not declared on that occasion with a **TFS_GetFirst..** or **TFS_GetNext..**, can be grabbed directly, however.
The entire File-System does 4 Gbytes on a linear storage area of maximum. The utilization of a MMU is included possible without further.
A use of modern data Flashes with PAGE sizes of 128byte .. 64kbytes can be made possible at any time by means of a "ReadPageToBuffer-UpdateBuffer-ErasePage-WriteBufferToPage" mechanism, however the TinyFileSystem does not examine for optimized (summarized) PageWrite.
When using NOR-Flash, however, it is important to remember that 10.000 .. 100.000 cycles is a very short life with frequent updates in the same page.
In addition, the TinyFileSystem do not respect hot spots (high-update files).

User-Functions:

TFS-Control:	Description
TFS_Init	Initialization of the File-System
TFS_GetRev	It returns a pointer on TFS revision
TFS_Flush	save the TFS as image into a Windows/LINUX-file
TFS_Format	Format the drive

User:	Description
TFS_BecomeUser	As User announce
TFS_BecomeUserSubROOT	As User in a subdirectory as user-ROOT announce
TFS_CloseUser	As User cancel
TFS_GetFreeSize	Return the brutto free-memory of the drive

Directories:	Description
TFS_CreateDir	Create a directory
TFS_RemoveDir	Remove a directory
TFS_RenameDir	Rename a directory
TFS_ChangeDir	Change current path
TFS_ChangeDirTemp	Change current path temorary
TFS_BackDirTemp	returns from temporary path
TFS_GetCurrentDir	Return the name of current directory
TFS_GetCurrentPath	Return the complete current path from ROOT

Files:	Description
TFS_CreateFile	Create a file
TFS_RemoveFile	Remove a file
TFS_RenameFile	Rename a file
TFS_MoveFile	Move a file
TFS_AttribFile	Change the attributes of a file
TFS_GetFileAttrib	It returns the attributes of a file
TFS_ResizeFile	Resize a file
TFS_GetFileSize	It returns the size of a file
TFS_OpenFile	Open a file in "mode"
TFS_CloseFile	Close the opened file
TFS_SeekFile	Place the pointer in opened file absolutely
TFS_TellFile	It returns the pointer in opened file
TFS_ExpandFile	Expand the size of opened file
TFS_SetEOF	Set EndOfFile in opened file to actual R/W-pointer
TFS_ReadFile	Read data from opened file
TFS_WriteFile	Write data into opened file
TFS_WriteFileE	Write data into opened file, expand this file if its needed
TFS_GetErrNo	read the error-code from Open, Read, Write ...

Links:	Description
TFS_CreateLink	Create a Link to a directory or file
TFS_RemoveLink	Remove a Link
TFS_RenameLink	Rename a Link
TFS_ReadLink	It returns the name of the linked directory or file

Entries:	Description
TFS_GetFirstName	It returns the name of first entry in current directory
TFS_GetNextName	It returns the name of next entry in current directory
TFS_GetFirst	It returns all infos of first entry in current directory into a struct
TFS_GetNext	It returns all infos of next entry in current directory into a struct
TFS_GetEntry	It returns all infos of the given entry (dir/file/link) into a struct

Optional:	Description
TFS_Defrag	defragment the drive
TFS_Repair	repairs the hierarchical pointersystem

Error-Codes:

Name	Decimal_Value	Description
TFS_NO_ERR	0	no errors
TFS_USR_OVF	200	no user free
TFS_DBL_USER	201	double user
TFS_NO_USER	202	not a valid user
TFS_SUB_USER	203	this DIR is ROOT of a active user
TFS_NAME_EXIST	210	name of entry exist in this DIR
TFS_NOT_EXIST	211	DIR or FILE not exist
TFS_PATH_ERR	212	error on PATH
TFS_TMP_DIR	213	Temp-Dir is still used / not set
TFS_NO_FILE	214	error on PATH / no FILE-Name given
TFS_FILE_RO	215	file to open for writing is read-only
TFS_FILE_WO	216	file to open for reading is write-only
TFS_FILE_EOF	217	end-of-file while reading or writing
TFS_NOT_EMPTY	218	entry not empty
TFS_FILE_OPEN	219	current user have a opened file
TFS_NO_DATA	220	current file-lenght is zero
TFS_WRONG_PTR	221	offset into open file is wrong (or size for R/W)
TFS_LINKED	230	entry is linked
TFS_MAX_LINK	231	entry is max count linked
TFS_LINK_ERR	232	error in link-mechanism
TFS_NO_LINK	233	no link to an entry in link-entry
TFS_MEM_ERR	250	error in memory-manager
TFS_MEM_OVF	251	memory overflow
TFS_WR_PTR	252	user-buffer for write/read is in TFS-area
TFS_WRITE_ERR	253	memory write error
TFS_INVALID	254	invalid TFS-image (for Windows/LINUX-Host)

Configuration of the File-System

The pC/TFS File-System can be configured in addition to the to-use HW-driver some numbers of ways to configure services as well as to reduce the memory requirements - code-size for the compilers "unused code" may not clearly identify - available. These are in the file "TFS_cfg.h" together.

user configuration	description
TFS_MAX_USER	max users (tasks)
TFS_HANDLES	max files opened by a user (task)
TFS_Name_SIZE	max name size of every entry
TFS_TempDIR	use the one-level temporary current-dir feature
TFS_SubROOT	use the user Sub-ROOT feature
TFS_AddMin	minimum size of a DB (DataBlock of a file)

internal configuration	description
TFSDBLLINKED	double linked system
TFSLONGAUTO	automatic TFS_LONG switching

Of course, the file system still needs the memory to be used, declared by the start address `TFS_START` and the size `TFS_SIZE` in bytes.
The memory of the file system can be declared by two ways:

- for xRAM-Drive (compiler known memory):

```
#define  TFS_SIZE     0x00010000
U08      TFS_START[TFS_SIZE];
```

- for EE / FLASH / xRAM (compiler unknown memory):

```
#define  TFS_START    0x00200000      // TFS startaddress
#define  TFS_SIZE     0x00010000      // TFS size
```

allgemeines

TFS_Init

```
U08 TFS_Init(void)
```

Initialize the File-System and installs on use of a RTOS the required mutex/semaphore. If the system should be recognized as unformatted, so this is executed.
This function must be called before all other fileservices at the system initialization once.
If you using the Windows/Linux-HOST Port, a Windows/LINUX Image-file will be loaded first.
The memory of the filesystem can be declared by two ways:

- for RAM-Drive (compiler known memory):

```
#define  TFS_SIZE     0x00010000
U08      TFS_START[TFS_SIZE];
```

- for EE / FLASH / RAM (compiler unknown memory):

```
#define  TFS_START    0x00200000      // TFS startaddress
#define  TFS_SIZE     0x00010000      // TFS size
```

Parameters

none

Return Value

TFS_NO_ERR	filesystem initialised
TFS_MEM_ERR	Mistakes in the memory management
TFS_MEM_OVF	File-system to small for ROOT-Entry
TFS_INVALID	IMAGE invalid (only on Windows/Linux-HOST)

Example

```
#define  TFS_SIZE     0x00010000
U08      TFS_START[TFS_SIZE];

void main(void)
{
    U08  returnOk;

    OS_Init();
    .
    .
    returnOk=TFS_Init();
    .
    .
    OS_Start();
}
```

TFS_GetRev

```
void TFS_GetRev(TFS_PATHNAME OS_HUGE **pointer)
```

It returns a pointer on the TFS-revision (NULL-terminated ASCII-array).

Parameters

**pointer	pointer to pointer will get the address of array

Return Value

none

Example

```
void OS_FAR Task1(void *data)
{
    TFS_PATHNAME OS_HUGE  *Revision;

    .
    .
    while(1)
    {
        .
        TFS_GetRev(&Revision);
        .
    }
}
```

TFS_Flush

```
U08 TFS_Flush(void)
```

Only by using the Windows or Linux HOST
Saves the filesystem as IMAGE into a Windows/Linux-file.

Parameters

none

Return Value

TFS_NO_ERR	*Drive saved*
from Windows/LINUX	*see Windows/LINUX*

Example

```
void main(void)
{
    U08  returnOk;

    .
    returnOk=TFS_Init();
    .
    .
    returnOk=TFS_Flush();
}
```

TFS_Format

```
U08 TFS_Format(void)
```

Formats the TFS-Drive and writes down the ROOT-Entry. At this time no user may be known.

Parameters

none

Return Value

TFS_NO_ERR	Drive formatted
TFS_USER	at minimum one user is known
TFS_MEM_ERR	Mistakes in the memory management
TFS_MEM_OVF	File-system to small for ROOT-Entry

Example

```
void OS_FAR Task1(void *data)
{
    U08  returnOk;
    .
    .
    while(1)
    {
        .
        returnOk=TFS_Format();
        .
        .
    }
}
```

TFS_BecomeUser

```
U08 TFS_BecomeUser(TFS_USER OS_HUGE *TFSUser)
```

If creates a new User.
This function initializes the User-Control-Block and writes down the new user into the internal list. Every Task need only once to register, after this evey User can handle TFS_HANDLES files. After registration of an user can call this drive and file accesses.

Parameters

*TFSUser	pointer to user-control-block

Return Value

TFS_NO_ERR	User successfully created
TFS_DBL_USER	this User already is announced
TFS_USR_OVF	already TFS_maxUSER are announced

Example

```
TFS_USER TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
    }
}
```

TFS_BecomeUserSubROOT

```
U08 TFS_BecomeUserSubROOT(TFS_USER OS_HUGE *TFSUser, TFS_PATHNAME OS_HUGE *FullDirPath)
```

Creates a new User and locks him up in the sub-directory of the drive as his ROOT-directory.
This function initializes the User-Control-Block and writes down the new user into the internal list. Every Task need only once to register, after this evey User can handle TFS_HANDLES files. After registration of an user can call this drive and file accesses.
This user can not leave his user-ROOT directory to Drive-ROOT using all possible pathdata like "**..**" , "**/**" , "**/Dir1**".

WARNING! Is in this Sub-Tree a link to an other tree beginning on the Drive-ROOT included (created by an Drive-ROOT user), so this user here can exit this lock up using this link !

Parameters

*TFSUser	pointer to user-control-block
*FullPathDir	complete path to directory beginning from Drive-ROOT

Return Value

TFS_NO_ERR	User successfully created
TFS_DBL_USER	this User already is announced
TFS_USR_OVF	already TFS_maxUSER are announced
TFS_PATH_ERR	an element of the path statement doesn't exist
TFS_NOT_EXIST	the directory doesn't exist

Example

```
TFS_USER TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUserSubROOT(&TFS_User1, "/Dir2/Dir21/Dir214");

                                // locks up the user in Dir214 as his ROOT
        .
        .
    }
}
```

TFS_CloseUser

```
U08 TFS_CloseUser(void)
```

If deletes a registered user from the internal list. This user must be announced again for it before accesses to the drive become again.

Parameters

none

Return Value

TFS_NO_ERR	User successfully deleted
TFS_FILE_OPEN	User opened a file currently
TFS_NO_USER	User unknown

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        returnOk=TFS_CloseUser();
        .
    }
}
```

TFS_GetFreeSize

```
TFS_LONG TFS_GetFreeSize(void)
```

Returns the actual brutto free-memory of the TFS-drive.

Parameters

none

Return Value

If the returned value zero, so get the following Error-Codes with `TFS_GetErrNo()`.

TFS_NO_ERR	no free memory (drive is full)
TFS_NO_USER	User unknown
TFS_MEM_ERR	Mistakes in the memory management

Example

```
TFS_USER TFS_User1;

void OS_FAR Task1(void *data)
{
    TFS_LONG  free_mem;
    U08       returnOk;
    .
    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        free_mem=TFS_GetFreeSize();
        if(!free_mem)
            returnOk=TFS_GetErrNo();
        .
        .
    }
}
```

Directory and File-Handlings

TFS_CreateDir

```
U08 TFS_CreateDir(TFS_PATHNAME OS_HUGE *Name)
```

Creates a new directory in the current or handed over path. The path statement can absolutely or relatively take place on that occasion.

Parameters

*Name	Directory-name [with path]

Return Value

TFS_NO_ERR	Directory created
TFS_NO_USER	User unknown
TFS_PATH_ERR	an element of the path statement doesn't exist
TFS_NAME_EXIST	a directory with same name exists already in this directory
TFS_MEM_ERR	Mistakes in the memory management
TFS_MEM_OVF	Drive full

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        returnOk=TFS_CreateDir("/usr");               // from ROOT
        .
        returnOk=TFS_CreateDir("../local/test.src");  // from one level back
        .
        returnOk=TFS_CreateDir("config.save");        // in actual directory
        .
        .
        returnOk=TFS_CloseUser();
        .
    }
}
```

TFS_RemoveDir

```
U08 TFS_RemoveDir(TFS_PATHNAME OS_HUGE *Name)
```

Removes the directory in the current or handed over path. The path statement can absolutely or relatively take place on that occasion. That to deleting directory must be included empty and no link may point this entry.

Parameters

*Name	Directory-name [with path]

Return Value

TFS_NO_ERR	Directory deleted
TFS_NO_USER	User unknown
TFS_PATH_ERR	an element of the path statement doesn't exist
TFS_NOT_EMPTY	the directory is not empty
TFS_SUB_USER	the directory is ROOT of an active user
TFS_LINKED	the directory is linked from another entry
TFS_NO_LINK	the directory is a link but points to no entry or this linked entry dosn't know this
TFS_LINK_ERR	Mistakes in the link-mechanism
TFS_MEM_ERR	Mistakes in the memory management

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        returnOk=TFS_RemoveDir("config.save");      // in actual directory
        .
        .
    }
}
```

TFS_RenameDir

```
U08 TFS_RenameDir(TFS_PATHNAME OS_HUGE *Name, TFS_PATHNAME OS_HUGE *NewName)
```

Changes the name of directory in the current or handed over path. The path statement can absolutely or relatively take place on that occasion. A Path in the new name will be ignored, so the directory can't be moved !

Parameters

*Name	Directory-name [with path]
*NewName	new Directory-name (a path will ignored)

Return Value

TFS_NO_ERR	directory renamed
TFS_NO_USER	User unknown
TFS_PATH_ERR	an element of the path statement doesn't exist
TFS_NAME_EXIST	a entry with same name exists already in this directory
TFS_NOT_EXIST	the directory doesn't exist
TFS_MEM_ERR	Mistakes in the memory management

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        returnOk=TFS_RenameDir("/usr", "user");      // in ROOT
        .
        .
    }
}
```

TFS_ChangeDir

```
U08 TFS_ChangeDir(TFS_PATHNAME OS_HUGE *Name)
```

Changes the current directory. The path statement can absolutely or relatively take place on that occasion.

Parameters

*Name	Directory-name [with path]

Return Value

TFS_NO_ERR	Directory changed
TFS_NO_USER	User unknown
TFS_PATH_ERR	an element of the path statement doesn't exist
TFS_NOT_EXIST	the directory doesn't exist

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        returnOk=TFS_ChangeDir("../test.src");
        .
        .
        .
    }
}
```

TFS_ChangeDirTemp

```
U08 TFS_ChangeDirTemp(TFS_PATHNAME OS_HUGE *Name)
```

Changes the current directory temporary. The path statement can absolutely or relatively take place on that occasion. The current directory up to this call will registrated internal. This feature can only be used one level per user.

Parameters

*Name	Directory-name [with path]

Return Value

TFS_NO_ERR	Directory changed
TFS_NO_USER	User unknown
TFS_TMP_DIR	Temp-Dir is still used
TFS_PATH_ERR	an element of the path statement doesn't exist
TFS_NOT_EXIST	the directory doesn't exist

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;
    U08  Name[TFS_Name_SIZE];

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        returnOk=TFS_ChangeDirTemp("../userfiles");
        if(returnOk == TFS_NO_ERR) {
            .
            returnOk=TFS_GetFirstName(Name);
            .
            .
            do {
                .
                returnOk=TFS_GetNextName(Name);
                .
                .
            } while(returnOk == TFS_NO_ERR);
            .
            returnOk=TFS_BackDirTemp();
        }
        .
        .
    }
}
```

TFS_BackDirTemp

```
U08 TFS_BackDirTemp(void)
```

Returns from the temporary directory to the registrated directory from TFS_ChangeDirTemp().

Parameters

none

Return Value

TFS_NO_ERR	Directory changed
TFS_NO_USER	User unknown
TFS_TMP_DIR	Temp-Dir is not set

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;
    U08  Name[TFS_Name_SIZE];

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        returnOk=TFS_ChangeDirTemp("../userfiles");
        if(returnOk == TFS_NO_ERR) {
            .
            returnOk=TFS_GetFirstName(Name);
            .
            .
            do {
                .
                returnOk=TFS_GetNextName(Name);
                .
                .
            } while(returnOk == TFS_NO_ERR);
            .
            returnOk=TFS_BackDirTemp();
        }
        .
        .
    }
}
```

TFS_GetCurrentDir

```
U08 TFS_GetCurrentDir(U08 OS_HUGE *Name)
```

returns the current directory. It will return only the directory name on that occasion without path.

Parameters

*Name	pointer to array for Directory-name

Return Value

TFS_NO_ERR	Directory readed
TFS_NO_USER	User unknown

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;
    U08  actDir[TFS_Name_SIZE];

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        returnOk=TFS_GetCurrentDir(actDir);
        .
        .
        .
    }
}
```

TFS_GetCurrentPath

```
U08 TFS_GetCurrentPath(U08 OS_HUGE *Path, TFS_LONG maxlen)
```

returns the complete path of current directory. For this a recursive function is used internally, in order to return the path from (user-)ROOT up to including current directory name.

Parameters

*Path	pointer to array for complete path
maxlen	max length of Path[]

Return Value

TFS_NO_ERR	path complete readed
TFS_MEM_OVF	pathsize greater / equal `maxlen`
TFS_NO_USER	User unknown

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;
    U08  actPath[1024];

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        returnOk=TFS_GetCurrentPath(actPath, 1024);
        .
        .
        .
    }
}
```

Directory and File-Handlings

TFS_CreateFile

```
U08 TFS_CreateFile(TFS_PATHNAME OS_HUGE *Name, TFS_ATTR Attr, TFS_LONG size)
```

Creates a new file in the current or handed over path in stated size. The path statement can absolutely or relatively take place on that occasion. As attributes, ReadOnly, WriteOnly and/or Hidden can be declared. ATTENTION! Files don't possess any type in this system.

Parameters

*Name	File-name [with path]
Attr	Attributes of this file
size	size of file

Return Value

TFS_NO_ERR	File created
TFS_NO_USER	User unknown
TFS_PATH_ERR	an element of the path statement doesn't exist
TFS_NAME_EXIST	a file with same name exists already in this directory
TFS_MEM_ERR	Mistakes in the memory management
TFS_MEM_OVF	Drive full

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        returnOk=TFS_CreateFile("/file1", TFS_ATTR_RO, 100);

                                                    // from ROOT
        .
        returnOk=TFS_CreateFile("../test.src/main.c", TFS_ATTR_SYS, 350);

                                                    // from one level back
        .
        returnOk=TFS_CreateFile("makefile.mak", 0, 140);

                                                    // in actual directory
        .
        .
    }
}
```

TFS_RemoveFile

```
U08 TFS_RemoveFile(TFS_PATHNAME OS_HUGE *Name)
```

Deletes the file in the current or handed over path. The path statement can absolutely or relatively take place on that occasion. The to deleting file cannot be opened by any other user on that occasion and no link may point this entry.

Parameters

*Name	File-name [with path]

Return Value

TFS_NO_ERR	File deleted
TFS_NO_USER	User unknown
TFS_FILE_OPEN	User, itself or other, this file opened currently
TFS_PATH_ERR	an element of the path statement doesn't exist
TFS_LINKED	the file is linked from another entry
TFS_NO_LINK	the directory is a link but points to no entry or this linked entry dosn't know this
TFS_LINK_ERR	Mistakes in the link-mechanism
TFS_MEM_ERR	Mistakes in the memory management

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        returnOk=TFS_RemoveFile("makefile.mak");      // in actual directory
        .
        .
    }
}
```

TFS_RenameFile

```
U08 TFS_RenameFile(TFS_PATHNAME OS_HUGE *OldName, TFS_PATHNAME OS_HUGE *NewName)
```

Changes the name of file in the current or handed over path. The path statement can absolutely or relatively take place on that occasion. The to changing file cannot be opened by any other user on that occasion.
A Path in the new name will be cut, so the file can't be moved !

Parameters

*OldName	File-name [with path]
*NewName	new File-name (a path will ignored)

Return Value

TFS_NO_ERR	File name changed
TFS_NO_USER	User unknown
TFS_NO_FILE	no file name in the path or as new name given
TFS_FILE_OPEN	User, itself or other, this file opened currently
TFS_NAME_EXIST	a file with same name exists already in this directory
TFS_NOT_EXIST	File doesn't exist in this directory
TFS_PATH_ERR	an element of the path statement doesn't exist
TFS_MEM_OVF	Drive full

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        returnOk=TFS_RenameFile("/test.src/main.c", "modul.c");
        .
        .
    }
}
```

TFS_MoveFile

```
U08 TFS_MoveFile(TFS_PATHNAME OS_HUGE *Name, TFS_PATHNAME OS_HUGE *NewPath)
```

Move the file in the current or handed over path into given directory. The path statement can absolutely or relatively take place on that occasion. The to moving file cannot be opened by any user on that occasion. A filename in the new path will be cut, so the file can't be renamed !

Parameters

*Name	File-name [with path]
*NewPath	new Path (a filename will ignored)

Return Value

TFS_NO_ERR	File moved
TFS_NO_USER	User unknown
TFS_NO_FILE	no file name in the path given
TFS_FILE_OPEN	User, itself or other, this file opened currently
TFS_NAME_EXIST	a file with same name exists already in this destination-directory
TFS_NOT_EXIST	File doesn't exist in this directory
TFS_PATH_ERR	an element of the path statement doesn't exist
TFS_LINKED	the file is linked from another entry
TFS_MEM_ERR	Mistakes in the memory management

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        returnOk=TFS_MoveFile("/test.src/main.c", "../version.src");
        .
        .
    }
}
```

TFS_AttribFile

```
U08 TFS_AttribFile(TFS_PATHNAME OS_HUGE *Name, TFS_ATTR Attribs)
```

Changes the attributes of the file in the current or handed over path. The path statement can absolutely or relatively take place on that occasion. The to changing file cannot be opened by any other user on that occasion.

Parameters

*Name	File-name [with path]
Attribs	ew File-attributs

Return Value

TFS_NO_ERR	File attributes changed
TFS_NO_USER	User unknown
TFS_NO_FILE	no file name in the path given
TFS_FILE_OPEN	User, itself or other, this file opened currently
TFS_NOT_EXIST	File doesn't exist in this directory
TFS_PATH_ERR	an element of the path statement doesn't exist
TFS_MEM_ERR	Mistakes in the memory management

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        returnOk=TFS_AttribFile("../TFS_err.log", TFS_ATTR_RO | TFS_ATTR_SYS);
        .
        .
    }
}
```

TFS_GetFileAttrib

```
TFS_ATTR TFS_GetFileAttrib(TFS_PATHNAME OS_HUGE *Name)
```

It returns the attributes of the file in the current or handed over path. The path statement can absolutely or relatively take place on that occasion.

Parameters

*Name	File-name [with path]

Return Value

If the returned attribs -1, so you get the error-codes with a following `TFS_GetErrNo()`.

TFS_NO_ERR	File attributes readed
TFS_NO_USER	User unknown
TFS_NO_FILE	no file name in the path given
TFS_NOT_EXIST	File doesn't exist in this directory
TFS_PATH_ERR	an element of the path statement doesn't exist
TFS_MEM_ERR	Mistakes in the memory management

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08       returnOk;
    TFS_ATTR  attribs;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        attribs=TFS_GetFileAttrib("../TFS_err.log");
        if(attribs==(TFS_ATTR)(-1))
            returnOk=TFS_GetErrNo();
        .
        .
    }
}
```

TFS_ResizeFile

```
U08 TFS_ResizeFile(TFS_PATHNAME OS_HUGE *Name, TFS_LONG newsize)
```

Changes the size of a file in the current or handed over path. The path statement can absolutely or relatively take place on that occasion. The to changing file cannot be opened by any other user on that occasion.

Parameters

*Name	File-name [with path]
newsize	new File-size

Return Value

TFS_NO_ERR	File-size changed
TFS_NO_USER	User unknown
TFS_NO_FILE	no file name in the path given
TFS_FILE_OPEN	User, itself or other, this file opened currently
TFS_NOT_EXIST	File doesn't exist in this directory
TFS_PATH_ERR	an element of the path statement doesn't exist
TFS_MEM_OVF	Drive full
TFS_MEM_ERR	Mistakes in the memory management

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        returnOk=TFS_ResizeFile("/Dir2/config.sys", 200);
        .
        .
    }
}
```

TFS_GetFileSize

```
TFS_LONG TFS_GetFileSize(TFS_PATHNAME OS_HUGE *Name)
```

It returns the size of the file in the current or handed over path. The path statement can absolutely or relatively take place on that occasion.

Parameters

*Name	File-name [with path]

Return Value

If the returned filesize zero, so you get the error-codes with a following `TFS_GetErrNo()`.

TFS_NO_ERR	filesize returned
TFS_NO_USER	User unknown
TFS_NO_FILE	no file name in the path given
TFS_NOT_EXIST	File doesn't exist in this directory
TFS_PATH_ERR	an element of the path statement doesn't exist
TFS_MEM_ERR	Mistakes in the memory management

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08       returnOk;
    TFS_LONG  filesize;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        filesize=TFS_GetFileSize("/Dir2/config.sys");
        if(!filesize)
            returnOk=TFS_GetErrNo();
        .
        .
    }
}
```

File-Access

TFS_OpenFile

```
TFS_HANDLE TFS_OpenFile(TFS_PATHNAME OS_HUGE *Name, U08 Mode)
```

Open a file in the current or handed over path. The path statement can absolutely or relatively take place on that occasion. The to opening file cannot be opened by any other user on that occasion. For all folloing accesses a handle will returned, under this this file-data can access. This handle is intern referenced with the User and is checked on every access.
If a NULL-handle returned, so get the Error-Code with `TFS_GetErrNo()`.

Access conditions:
If a file opened (more than once) to reading, an other User can't open this file to writing - if a file from one User opened to writing, no other User can open this file for reading or writing.

Parameters

*Name	File-name [with path]
mode	mode of access (ReadOnly/WriteOnly/ReadWrite)

Return Value

If a NULL-handle returned, so get the Error-Codes with a following `TFS_GetErrNo()`.

TFS_NO_ERR	File opened
TFS_NO_USER	User unknown
TFS_NO_FILE	no file name in the path given
TFS_FILE_OPEN	another User opened this file currently (see Access conditions)
TFS_FILE_RO	File is ReadOnly and cannot be opened "write"
TFS_FILE_WO	File is WriteOnly and cannot be opened "read"
TFS_NOT_EXIST	File doesn't exist in this directory
TFS_PATH_ERR	an element of the path statement doesn't exist
TFS_MEM_ERR	Mistakes in the memory management

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08         returnOk;
    TFS_HANDLE  handl;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        handl=TFS_OpenFile("/Dir2/config.sys", TFS_ACC_RO);
        if(!handl)
            returnOk=TFS_GetErrNo();
        .
        .
    }
```

}

TFS_CloseFile

```
U08 TFS_CloseFile(TFS_HANDLE Handl)
```

Close the currently opened file.

Parameters

Handl	File-Handle

Return Value

TFS_NO_ERR	File closed
TFS_NO_USER	User unknown
TFS_NO_FILE	Handle invalid

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08         returnOk;
    TFS_HANDLE  handl;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        handl=TFS_OpenFile("/Dir2/config.sys", TFS_ACC_RO);
        if(handl) {
            .
            returnOk=TFS_CloseFile(handl);
            .
        }
        .
    }
}
```

TFS_SeekFile

```
U08 TFS_SeekFile(TFS_LONG offset, TFS_HANDLE Handl)
```

Places the R/W-pointer within the opened file absolutely.

Parameters

offset	absolut pointer-position (in bytes), `0` for start of file, `TFS_SEEK_EOF` for end of file
Handl	File-Handle

Return Value

TFS_NO_ERR	Pointer in file placed
TFS_NO_HAND	handle invalid
TFS_NO_USER	User unknown
TFS_NO_FILE	no file is opened / Handle invalid
TFS_NO_DATA	File has zero-lenght
TFS_WRONG_PTR	offset greater file-size

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08         returnOk;
    TFS_HANDLE  handl;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        handl=TFS_OpenFile("/Dir2/config.sys", TFS_ACC_RO);
        if(handl) {
            .
            returnOk=TFS_SeekFile(100, handl);
        }
        .
    }
}
```

TFS_TellFile

```
TFS_LONG TFS_TellFile(TFS_HANDLE Handl)
```

It returns the R/W-pointer of opened file.

Parameters

Handl	File-Handle

Return Value

If the returned position is zero, so get the following Error-Codes with `TFS_GetErrNo()`.

TFS_NO_ERR	pointer returned (start of file)
TFS_NO_HAND	handle invalid
TFS_NO_USER	User unknown
TFS_NO_FILE	no file is opened / Handle invalid
TFS_NO_DATA	File has zero-lenght

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08         returnOk;
    TFS_HANDLE  handl;
    TFS_LONG    posit;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        handl=TFS_OpenFile("/Dir2/config.sys", TFS_ACC_RO);
        if(handl) {
            .
            posit=TFS_TellFile(handl);
            if(!posit)
                returnOk=TFS_GetErrNo();

        }
        .
    }
}
```

TFS_ExpandFile

```
U08 TFS_ExpandFile(TFS_LONG addsize, TFS_HANDLE Handl)
```

Expand the size of opened file.

Parameters

addsize	File-size to add
Handl	File-Handle

Return Value

TFS_NO_ERR	size added
TFS_NO_HAND	handle invalid
TFS_NO_USER	User unknown
TFS_NO_FILE	no file opened
TFS_MEM_OVF	Drive full
TFS_MEM_ERR	Mistakes in the memory management

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08         returnOk;
    TFS_HANDLE  handl;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        handl=TFS_OpenFile("/Dir2/config.sys", TFS_ACC_RO);
        if(handl) {
            .
            returnOk=TFS_ExpandFile(50, handl);
        }
        .
    }
}
```

TFS_SetEOF

```
U08 TFS_SetEOF(TFS_HANDLE Handl)
```

Set EndOfFile in opened file to actual R/W-pointer.

Parameters

Handl	File-Handle

Return Value

TFS_NO_ERR	EndOfFile set
TFS_NO_HAND	handle invalid
TFS_NO_USER	User unknown
TFS_NO_FILE	no file is open
TFS_NO_DATA	File has zero-lenght
TFS_FILE_RO	File or Open-mode is Read-Only

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08         returnOk;
    TFS_HANDLE  handl;
    U08         writebuffer[]={"TFS_Test_File R/W"};
    TFS_LONG    written;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        handl=TFS_OpenFile("/Dir2/config.sys", TFS_ACC_RW);
        if(handl) {
            written=TFS_WriteFile(writebuffer, strlen(writebuffer)-1, handl);
            if(written == strlen(writebuffer)-1)
                returnOk=TFS_SetEOF(handl);
            .
        }
        .
    }
}
```

TFS_ReadFile

```
TFS_LONG TFS_ReadFile(U08 OS_HUGE *dest, TFS_LONG size, TFS_HANDLE Handl)
```

Reads number of bytes from currently opened file from current position. After successful reading, the R/W-pointer stands behind the readed block.
The readed number of bytes will returned. If this not the same from the call, so call `TFS_GetErrNo()` to get the error-code.

Parameters

*dest	Pointer to buffer where the bytes must written in
size	Bytes to read
Handl	File-Handle

Return Value

If the returned number of readed bytes not the same from the call, so you get the following error-codes from `TFS_GetErrNo()`.

TFS_NO_ERR	Bytes from file readed
TFS_NO_HAND	handle invalid
TFS_NO_USER	User unknown
TFS_NO_FILE	no file is open
TFS_NO_DATA	File has zero-lenght
TFS_FILE_EOF	End-Of-File
TFS_FILE_WO	File or Open-mode is Write-Only
TFS_WRONG_PTR	offset and/or size greater file-size

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08         returnOk;
    TFS_HANDLE  handl;
    U08         readbuffer[100];
    TFS_LONG    readed;

    .
    ...
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        handl=TFS_OpenFile("/Dir2/config.sys", TFS_ACC_RO);
        if(handl) {
            returnOk=TFS_SeekFile(50, handl);
            .
            readed=TFS_ReadFile(&readbuffer[0], 80, handl);
            if(readed != 80)
                returnOk=TFS_GetErrNo();
            .
        }
        .
        .
    }
```

}

TFS_WriteFile

```
TFS_LONG TFS_WriteFile(U08 OS_HUGE *src, TFS_LONG size, TFS_HANDLE Handl)
```

Writes number of byte in currently opened file beginning on current position. After successful writing, the R/W-pointer stands behind the written block.
The written number of bytes will returned. If this not the same from the call, so call `TFS_GetErrNo()` to get the error-code.

Parameters

*src	Pointer to source-buffer
size	Bytes to write
Handl	File-Handle

Return Value

If the returned number of written bytes not the same from the call, so you get the following error-codes from `TFS_GetErrNo()`.

TFS_NO_ERR	Bytes in file written
TFS_NO_HAND	Handle invalid
TFS_NO_USER	User unknown
TFS_NO_FILE	no file is open
TFS_NO_DATA	File has zero-lenght
TFS_FILE_EOF	End-Of-File
TFS_FILE_RO	File or Open-mode is Read-Only
TFS_WRONG_PTR	actual offset plus size greater file-size

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08         returnOk;
    TFS_HANDLE  handl;
    U08         writebuffer[]={"TFS_Test_File R/W"};
    TFS_LONG    written;

    .
    ...
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        handl=TFS_OpenFile("/Dir2/config.sys", TFS_ACC_WO);
        if(handl) {
            written=TFS_WriteFile(writebuffer, strlen(writebuffer)-1, handl);
            if(written != strlen(writebuffer)-1)
                returnOk=TFS_GetErrNo();
            .
            .
            .
        }
        .
    }
```

}

TFS_WriteFileE

```
TFS_LONG TFS_WriteFileE(U08 OS_HUGE *src, TFS_LONG size, TFS_HANDLE Handl)
```

Writes number of byte in currently opened file beginning on current position. After successful writing, the R/W-pointer stands behind the written block.
If the file is too small, then this is increased automatically around the difference. To increase first tries the last fragment of the file. If this is not possible, is tried, to provide one new fragment. Only even if this fails, from the found free fragments the more memory is built up.
The written number of bytes will returned. If this not the same from the call, so call `TFS_GetErrNo()` to get the error-code.

Parameters

*src	Pointer to source-buffer
size	Bytes to write
Handl	File-Handle

Return Value

If the returned number of written bytes not the same from the call, so you get the following error-codes from `TFS_GetErrNo()`.

TFS_NO_ERR	Bytes in file written
TFS_NO_HAND	Handle invalid
TFS_NO_USER	User unknown
TFS_NO_FILE	no file is open
TFS_FILE_RO	File or Open-mode is Read-Only
TFS_MEM_OVF	Drive full
TFS_MEM_ERR	Mistakes in the memory management

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08         returnOk;
    TFS_HANDLE  handl;
    U08         writebuffer[]={"TFS_Test_File R/W(e)"};
    TFS_LONG    written;

    .
    ...
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        handl=TFS_OpenFile("/Dir2/config.sys", TFS_ACC_WO);
        if(handl) {
            written=TFS_WriteFileE(writebuffer, strlen(writebuffer)-1, handl);
            if(written != strlen(writebuffer)-1)
                returnOk=TFS_GetErrNo();
            .
            .
            .
        }
        .
```

```
        }
}
```

TFS_GetErrNo

```
U08 TFS_GetErrNo(void)
```

returns the error-code from Open, Read, Write ...

Parameters

none

Return Value

error-code	error-code from last called and failed function-call without error-code return in API

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08         returnOk;
    TFS_HANDLE  handl;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        handl=TFS_OpenFile("/Dir2/config.sys", TFS_ACC_RO);
        if(!handl)
            returnOk=TFS_GetErrNo();
        .
        .
    }
}
```

Link-Handling

TFS_CreateLink

```
U08 TFS_CreateLink(TFS_PATHNAME OS_HUGE *OrgName, TFS_PATHNAME OS_HUGE *Name)
```

Creates a new link to a file or directory. The original entry can located in an other tree-part. As attribs of this link are the attribs of the linked entry valid.

Parameters

*OrgName	File/Directory-name to link [with path]
*Name	Link-name [with path]

Return Value

TFS_NO_ERR	Link created
TFS_NO_USER	User unknown
TFS_PATH_ERR	an element of the path statement doesn't exist
TFS_NAME_EXIST	a entry with same name exists already in this directory
TFS_MEM_OVF	Drive full
TFS_MEM_ERR	Mistakes in the memory management
TFS_MAX_LINK	*OrgName is TFS_MAX_Links linked
TFS_LINK_ERR	Mistakes in the linker

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;

    .
    ...
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        returnOk=TFS_CreateLink("/Dir2/config.sys", "../test.dir/linked.conf");
        .
        .
    }
}
```

TFS_RemoveLink

```
U08 TFS_RemoveLink(TFS_PATHNAME OS_HUGE *LinkName)
```

Deletes the link in the current or handed over path. The path statement can absolutely or relatively take place on that occasion. The link to be deleted cannot be linked by another entry.
--- TO DELETING DIR / FILE: ---
Linked Dir / Files can't be removed while one link to this entry is valid !

Instead of `TFS_RemoveLink()` you can use the according (DIR/File) appendant primitive `TFS_RemoveFile()` or `TFS_RemoveDir()`.

Parameters

*Name	Link-name [with path]

Return Value

TFS_NO_ERR	Link removed
TFS_NO_USER	User unknown
TFS_PATH_ERR	an element of the path statement doesn't exist
TFS_MEM_ERR	Mistakes in the memory management
TFS_LINKED	*Name is oneself linked
TFS_NO_LINK	the entry is'nt a link / the entry is a link but points to no entry or this linked entry doesn't know this
TFS_LINK_ERR	Mistakes in the linker

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;

    .
    ...
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        returnOk=TFS_CreateLink("/Dir2/config.sys", "/test.dir/linked.conf");
        .
        .
        returnOk=TFS_RemoveLink("../test.dir/linked.conf");
    }
}
```

TFS_RenameLink

```
U08 TFS_RenameLink(TFS_PATHNAME OS_HUGE *OldName, TFS_PATHNAME OS_HUGE *NewName)
```

Changes the name of link in the current or handed over path. The path statement can absolutely or relatively take place on that occasion.
A Path in the new name will be cut, so the link can't be moved !

Parameters

*OldName	Link-name [with path]
*NewName	new Link-name (a path will ignored)

Return Value

TFS_NO_ERR	Linkname changed
TFS_NO_USER	User unknown
TFS_NAME_EXIST	a entry with same name exists already in this directory
TFS_NOT_EXIST	the link doesn't exist
TFS_PATH_ERR	an element of the path statement doesn't exist
TFS_MEM_ERR	Mistakes in the memory management

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        returnOk=TFS_RenameLink("/test.dir/linked.conf", "config.link");
        .
        .
    }
}
```

TFS_ReadLink

```
U08 TFS_ReadLink(TFS_PATHNAME OS_HUGE *Name, U08 OS_HUGE *LinkName)
```

Returns the name of the entry that is linked from *Name. It will return only the entry name on that occasion without path.

Parameters

*Name	Link-name [with path]
*LinkName	pointer to array for linked Entry-name

Return Value

TFS_NO_ERR	name readed
TFS_NO_USER	User unknown
TFS_PATH_ERR	an element of the path statement doesn't exist
TFS_MEM_ERR	Mistakes in the memory management
TFS_NO_LINK	the entry is'nt a link / the entry is a link but points to no entry or this linked entry doesn't know this
TFS_LINK_ERR	Mistakes in the linker

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;
    U08  Entry[TFS_Name_SIZE];

    .
    ...
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        returnOk=TFS_CreateLink("/Dir2/config.sys", "../test.dir/linked.conf");
        .
        returnOk=TFS_ReadLink("../test.dir/linked.conf", Entry);
        .
        returnOk=TFS_RemoveLink("../test.dir/linked.conf");
    }
}
```

Entries

TFS_GetFirstName

```
U08 TFS_GetFirstName(U08 OS_HUGE *Name)
```

Returns the name of first entry in the current directory and sets the "next"-counter of the user to begin of this directory. It will return only the directory or file name on that occasion without path.

Parameters

*Name	pointer to array for Entry-name

Return Value

TFS_NO_ERR	Entry read
TFS_NOT_EXIST	no entry exist
TFS_NO_USER	User unknown

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;
    U08  Name[TFS_Name_SIZE];

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        returnOk=TFS_GetFirstName(Name);
        .
        .
        .
    }
}
```

TFS_GetNextName

```
U08 TFS_GetNextName(U08 OS_HUGE *Name)
```

Returns the name of next entry in the current directory and steps the "next"-counter of the user one entry forward. It will return only the directory or file name on that occasion without path.

Parameters

*Name	pointer to array for Entry-name

Return Value

TFS_NO_ERR	Entry read
TFS_NOT_EXIST	no further entry existing
TFS_NO_USER	User unknown

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08  returnOk;
    U08  Name[TFS_Name_SIZE];

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        returnOk=TFS_GetFirstName(Name);
        .
        do {
            .
            returnOk=TFS_GetNextName(Name);
            .
            .
        } while(returnOk == TFS_NO_ERR);
        .
        .
    }
}
```

TFS_GetFirst

```
U08 TFS_GetFirst(TFS_GET OS_HUGE *get)
```

Returns in the struct all relevant information of first entry in the current directory and sets the "next"-counter of the user to begin of this directory.
following informations are part of the structure:

```
U08       Name[TFS_Name_SIZE]; // name of this entry
TFS_ATTR  Attr;                // attributes of this entry
TFS_LONG  Length;              // used length of this entry (0 if its a directory)
U08       Linked;              // not 0, if this entry is linked by an other entry
```

Parameters

*get	pointer to GET struct

Return Value

TFS_NO_ERR	Entry read
TFS_NOT_EXIST	no entry exist
TFS_NO_USER	User unknown

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08      returnOk;
    TFS_GET  get;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        .
        returnOk=TFS_GetFirst(&get);
        .
        .
        .
    }
}
```

TFS_GetNext

```
U08 TFS_GetNext(TFS_GET OS_HUGE *get)
```

Returns in the struct all relevant information of next entry in the current directory and steps the "next"-counter of the user one entry forward.

Parameters

*get	pointer to GET struct

Return Value

TFS_NO_ERR	Entry read
TFS_NOT_EXIST	no further entry existing
TFS_NO_USER	User unknown

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08      returnOk;
    TFS_GET  get;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        returnOk=TFS_GetFirst(&get);
        .
        do {
            .
            returnOk=TFS_GetNext(&get);
            .
            .
        } while(returnOk == TFS_NO_ERR);
        .
        .
    }
}
```

TFS_GetEntry

```
U08 TFS_GetEntry(TFS_PATHNAME OS_HUGE *Name, TFS_GET OS_HUGE *get)
```

Returns in the struct all relevant information of the given entry.

Parameters

*Name	Entry-name (dir/file/link) [with path]
*get	pointer to GET struct

Return Value

TFS_NO_ERR	Entry read
TFS_NOT_EXIST	this entry doesn't exist
TFS_NO_USER	User unknown

Example

```
TFS_USER  TFS_User1;

void OS_FAR Task1(void *data)
{
    U08      returnOk;
    TFS_GET  get;

    .
    while(1)
    {
        .
        returnOk=TFS_BecomeUser(&TFS_User1);
        .
        returnOk=TFS_GetEntry("/Dir2/element", &get);
        .
        if (get.Attr & TFS_ATTR_DIR) {
            .
            .
        }
        .
        .
    }
}
```

Comments

Comments

Comments