FileID

BLOODROCK/SDC

COLLABORATORS			
	TITLE : FileID		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	BLOODROCK/SDC	March 1, 2023	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

FileID

1.1 FileID.Guide: Contents

1.2 FileID.library/FIAllocFileInfo

NAME
FIAllocFileInfo - Create a FI_FileInfo structure.
SYNOPSIS
Result = FIAllocFileInfo()
D0
LONG FIAllocFileInfo()
FUNCTION
Creates a valid FI_FileInfo structure. This function
must be called first if you want to use FIIdentify() or
FIIdentifyFromName().
Note that every FI_FileInfo structure must be removed using

FI_FreeFileInfo() before you may finish your program.

RESULT

FI_FileInfo - Address of the requested FI_FileInfo
structure or NULL if any error occured
(out of memory).

SEE ALSO

FIFreeFileInfo()

1.3 FileID.library/FIFreeFileInfo

NAME

FIFreeFileInfo - Free a FI_FileInfo structure.

SYNOPSIS

Result = FIFreeFileInfo(FI_FileInfo)
D0 A1

LONG FIFreeFileInfo(STRUCT FI_FileInfo)

FUNCTION

Removes a FI_FileInfo structure from the system. Do not
call this function more than once for the same FI_FileInfo
structure. This would be the same as trying to Exec/FreeMem()
the same memory area twice.

INPUT

FIFileInfo - A FIFileInfo structure allocated with
FIAllocFileInfo()

RESULT

NONE - no errors possible.

SEE ALSO

FI_FileInfo structure

1.4 FileID.library/FIIdentify

NAME

FIIdentify - Identify a file type

SYNOPSIS

Result = FIIdentify(Address, FI_FileInfo)
D0 A0 A1

LONG FIIdentify(APTR, STRUCT FI_FileInfo)

FUNCTION

Examines a file in memory. Only the first 1200 bytes are

used therefor.
Assembler writers should note that A1 is preserved after
each FIIdentify() call. You may directly examine FI_FileInfo
using A1 after calling this function.
INPUTS
Address - The start address of a file already read.
This address MUST be even.
For a complete file analysis, you need
to read at least 1200 bytes of the file
(from Start, of course).
Note that if the file is smaller than 1200
bytes, you should fill up your file buffer
with zero bytes. Otherwise, FIIdentify()
may return *wrong* file IDs.
The file buffer must always be at least
1200 bytes ! Never pass smaller buffers!
FI_FileInfo - A valid FI_FileInfo structure.
Use FIAllocFileInfo() to create it.
RESULT
NONE - The passed FI_FileInfo structure will be
filled.
SEE ALSO
FIAllocFileInfo()
include/libraries/FileID.i
include/libraries/FileID_IDDefs.i

1.5 FileID.library/FIIdentifyFromName

NAME
FIIdentifyFromName - Identify a file type
SYNOPSIS
Result = FIIdentifyFromName(FI_FileInfo, Name)
D0 A1 D1
LONG FIIdentifyFromName(STRUCT FI_FileInfo, APTR)
FUNCTION
Loads the first 1200 bytes (the whole file, if smaller)
of the file you passed the name of, and tries to identi-
fy it.
Assembler writers should note that A1 is preserved after

each FIIdentify() call. You may directly examine FI_FileInfo
using A1 after calling this function.
INPUTS
FI_FileInfo - A valid FI_FileInfo structure.
Use FIAllocFileInfo() to create it.
Name - A null-terminated standard DOS string.
RESULT
ERR - ERROR or NULL if everything went ok.
The passed FI_FileInfo structure will be
filled. If an error occured, FI_Description
points to an error message instead of a
file description.
SEE ALSO
FIAllocFileInfo()
include/libraries/FileID.i
include/libraries/FileID_IDDefs.i

1.6 FileID.library/FIGetHighID

NAME
FIGetHighID - Get the highest FileID number.
SYNOPSIS
Result = FIGetHighID()
D0
LONG FIGetHighID()
FUNCTION
Returns the highest file ID number supported by the
currently installed library version.
All CPU registers are preserved by this function.
Only D0 is used for the result.
RESULT
HighID (LONG)
SEE ALSO
-

1.7 FileID.library/FIGetIDString

NAME
FIGetIDString - Get the FileID string belonging to a
specific ID number.
SYNOPSIS
Result = FIGetIDString(IDNumber)
D0 D0
APTR FIGetIDString(LONG)
FUNCTION
Returns the FI_Description belonging to the given ID number.
RESULT
An APTR to a FI_Description string or NULL if the given
Number was out of range.
SEE ALSO
-

1.8 FileID.Guide: Notes

Please don't open FileID.library with VERSION_ANY or VERSION == 1, because
V 1.0 has got a hard bug ! You should always use VERSION == 2 or higher.
For better performance on 680x0 Amigas, the start address passed to
FIIdentify() should be longword-aligned.
FIIdentify() doesn't check if the passed start address is valid.
If you pass e.g. a pointer to write-only hardware registers your program
will crash.
The maximum length of a file description string equals 35 bytes. Note this
if you want to write these strings e.g. into an own window.
When using FIIdentify(), you don't always need to read 1200 bytes of the file
to examine. If you want to save some time, you may read only 1024 bytes of
the file into the (cleared!) buffer. This prevents the heads of the Floppy/HD
moving to the next data block.
But in this case, Pro/NoiseTracker-Modules and StarTrekker modules can't be
identified, because this would take too much time.
The speedup for reading 1024 instead of 1200 bytes is very small. Checking
a complete directory containing 400 files on a Quantum LPS 240 harddisk took
only one second longer than the same test with reading only 1024 bytes of
every file.
This speed gain would be lost, if the library had to identify e.g. a Pro-

Tracker module from a buffer containing only the first 1024 data bytes.
The buffer you pass must be at least 1200 bytes in size, no matter how
many bytes you have read. The remaining buffer space must be filled with
NULL bytes to avoid wrong file identifications.
Note that FIIdentifyFromName() was made for easy single file analysis.
If you wish to examine lots of files, you should manage file I/O yourself
and then use FIIdentify(), because FIIdentifyFromName() has to allocate
and deallocate the needed memory for every single file.
Re-using the same file buffer for several files will save much time.
You may re-use an FI_FileInfo structure as often as you want. If you want
to examine lots of files, you don't have to FIAllocFileInfo() for every
single file.
Please do not write any data into the FI_FileInfo structure. Also, do not
try to create or free it yourself, because it's size may change in future
library versions.
Use FIAllocFileInfo() and FIFreeFileInfo() instead. It's also easier.
Do not misunderstand "FiIdentifyFromName". Currently, this function checks
only the file contents. The filename itself is not examined for any
extensions, making assumptions about what file might be. This is much too
unsure.
FIIdentify() doesn't change the passed file (-part) anyway.
By the way: opening FileID.library will fail if the library couldn't open
dos.library (any version). Rare, of course, but if you wonder about what's
the matter... :^)