FManC

1.0.0

Generated on Tue Feb 28 2023 00:07:23 for FManC by Doxygen 1.9.6

Tue Feb 28 2023 00:07:23

Chapter 1

Welcome to the FManC documentation website !

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

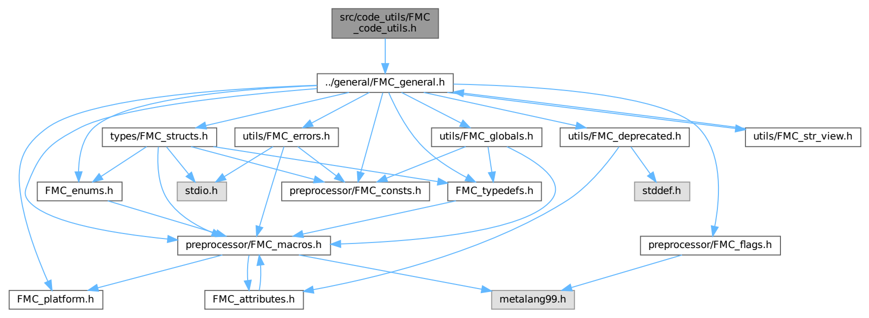
Chapter 3

File Documentation

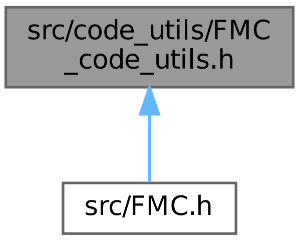
3.1 docs/documentation_pages/main_page.dox File Reference

3.2 src/code_utils/FMC_code_utils.h File Reference

Include dependency graph for FMC_code_utils.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define FMC_CODE_UTILS_H

3.2.1 Macro Definition Documentation

3.2.1.1 FMC_CODE_UTILS_H

```
#define FMC_CODE_UTILS_H
```

Definition at line 30 of file FMC_code_utils.h.

3.3 FMC_code_utils.h

Go to the documentation of this file.

```
00001 /*
00002
00003 MIT License
00004
00005 Copyright (c) 2022 Axel PASCON
00006
00007 Permission is hereby granted, free of charge, to any person obtaining a copy
00008 of this software and associated documentation files (the "Software"), to deal
00009 in the Software without restriction, including without limitation the rights
00010 to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
00011 copies of the Software, and to permit persons to whom the Software is
00012 furnished to do so, subject to the following conditions:
00013
00014 The above copyright notice and this permission notice shall be included in all
00015 copies or substantial portions of the Software.
00016
00017 THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
00018 IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
00019 FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
00020 AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
00021 LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
00022 OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
00023 SOFTWARE.
00024
00025 */
00026
00027 #pragma once
00028
00029 #ifndef FMC_CODE_UTILS_H
00030 #define FMC_CODE_UTILS_H
00031
00032 #include "../general/FMC_general.h"
00033
00034 #endif // FMC_CODE_UTILS_H
```

3.4 src/code_utils/FMC_codeUtils.c File Reference

3.5 FMC_codeUtils.c

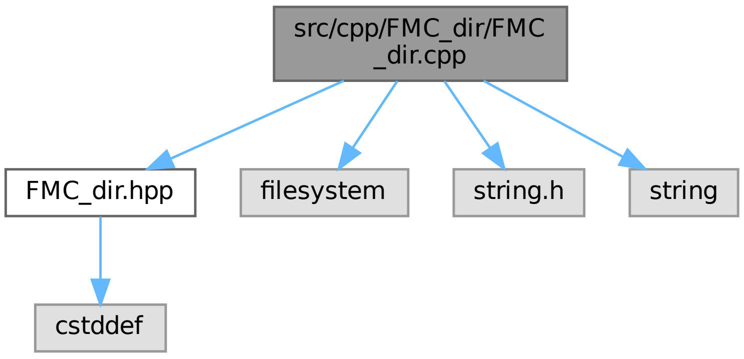
Go to the documentation of this file.

```
00001 /*
00002
00003 MIT License
00004
```

```
Copyright (c) 2022 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/
```

3.6 src/cpp/FMC_dir/FMC_dir.cpp File Reference

Include dependency graph for FMC_dir.cpp:



Functions

- int FMC_dirExists_ (const char ∗path)
- char ∗ FMC_getAbsolutePath_ (char ∗path, char ∗buffer, const size_t size)
- char ∗ FMC_getCurrentPath_ (char ∗path, const size_t size)
- int FMC_isBlock_ (const char ∗path)
- int FMC_isCharFile_ (const char ∗path)
- int FMC_isDir_ (const char ∗path)
- int FMC_isEmpty_ (const char ∗path)
- int FMC_isFIFO_ (const char ∗path)
- int FMC_isOther_ (const char ∗path)
- int FMC_isRegFile_ (const char ∗path)
- int FMC_isSocket_ (const char ∗path)
- int FMC_isSymLink_ (const char ∗path)

3.6.1 Function Documentation

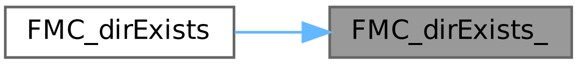
3.6.1.1 FMC_dirExists_()

```
int FMC_dirExists_ (
             const char * path )
```

Definition at line 38 of file FMC_dir.cpp.

Referenced by FMC_dirExists().

Here is the caller graph for this function:



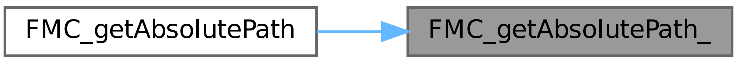
3.6.1.2 FMC_getAbsolutePath_()

```
char * FMC_getAbsolutePath_ (
             char * path,
             char * buffer,
             const size_t size )
```

Definition at line 157 of file FMC_dir.cpp.

Referenced by FMC_getAbsolutePath().

Here is the caller graph for this function:



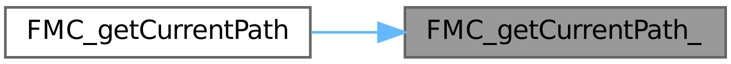
3.6.1.3 FMC_getCurrentPath_()

```
char * FMC_getCurrentPath_ (
             char * path,
             const size_t size )
```

Definition at line 142 of file FMC_dir.cpp.

Referenced by FMC_getCurrentPath().

Here is the caller graph for this function:



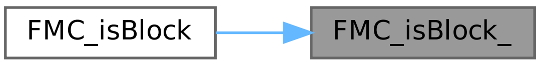
3.6.1.4 FMC_isBlock_()

```
int FMC_isBlock_ (
             const char * path )
```

Definition at line 70 of file FMC_dir.cpp.

Referenced by FMC_isBlock().

Here is the caller graph for this function:



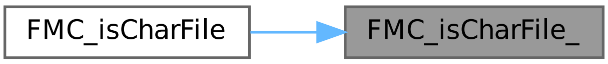
3.6.1.5 FMC_isCharFile_()

```
int FMC_isCharFile_ (
            const char * path )
```

Definition at line 79 of file FMC_dir.cpp.

Referenced by FMC_isCharFile().

Here is the caller graph for this function:



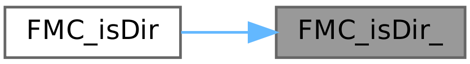
3.6.1.6 FMC_isDir_()

```
int FMC_isDir_ (
            const char * path )
```

Definition at line 43 of file FMC_dir.cpp.

Referenced by FMC_isDir().

Here is the caller graph for this function:



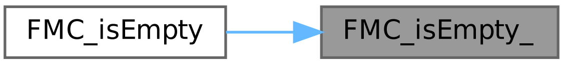
3.6.1.7 FMC_isEmpty_()

```
int FMC_isEmpty_ (
            const char * path )
```

Definition at line 128 of file FMC_dir.cpp.

Referenced by FMC_isEmpty().

Here is the caller graph for this function:



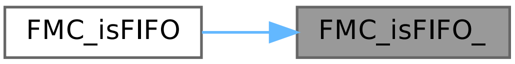
3.6.1.8 FMC_isFIFO_()

```
int FMC_isFIFO_ (
            const char * path )
```

Definition at line 97 of file FMC_dir.cpp.

Referenced by FMC_isFIFO().

Here is the caller graph for this function:



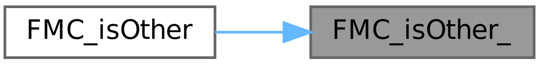
3.6.1.9 FMC_isOther_()

```
int FMC_isOther_ (
              const char * path )
```

Definition at line 106 of file FMC_dir.cpp.

Referenced by FMC_isOther().

Here is the caller graph for this function:



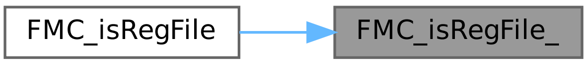
3.6.1.10 FMC_isRegFile_()

```
int FMC_isRegFile_ (
              const char * path )
```

Definition at line 52 of file FMC_dir.cpp.

Referenced by FMC_isRegFile().

Here is the caller graph for this function:



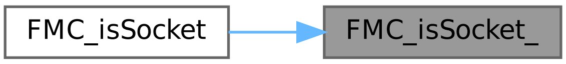
3.6.1.11 FMC_isSocket_()

```
int FMC_isSocket_ (
             const char * path )
```

Definition at line 88 of file FMC_dir.cpp.

Referenced by FMC_isSocket().

Here is the caller graph for this function:



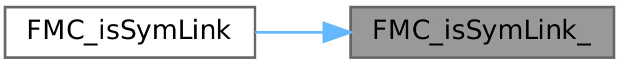
3.6.1.12 FMC_isSymLink_()

```
int FMC_isSymLink_ (
             const char * path )
```

Definition at line 61 of file FMC_dir.cpp.

Referenced by FMC_isSymLink().

Here is the caller graph for this function:



3.7 FMC_dir.cpp

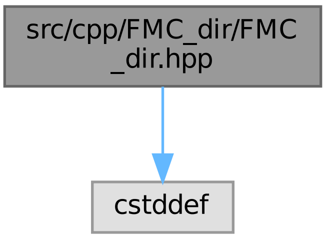
Go to the documentation of this file.

```
/*

MIT License

Copyright (c) 2022-2023 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/

#if __cplusplus < 201703L
    #error "The contents of <filesystem> are available only with C++17 or later."
#endif

#include "FMC_dir.hpp"
#include <filesystem>
#include <string.h>
#include <string>

namespace fs = std::filesystem;

int FMC_dirExists_(const char *path)
{
    return fs::exists(path);
}

int FMC_isDir_(const char *path)
{
    if(fs::exists(path))
    {
        return fs::is_directory(path);
    }
    else return -1;
}

int FMC_isRegFile_(const char *path)
{
    if(fs::exists(path))
    {
        return fs::is_regular_file(path);
    }
    else return -1;
}

int FMC_isSymLink_(const char *path)
{
    if (fs::exists(path))
    {
        return fs::is_symlink(path);
    }
    else return -1;
}

int FMC_isBlock_(const char* path)
{
    if (fs::exists(path))
    {
        return fs::is_block_file(path);
    }
    else return -1;
}

int FMC_isCharFile_(const char* path)
{
    if (fs::exists(path))
    {
        return fs::is_character_file(path);
    }
    else return -1;
}

int FMC_isSocket_(const char* path)
{
    if (fs::exists(path))
    {
        return fs::is_socket(path);
    }
    else return -1;
}

int FMC_isFIFO_(const char* path)
{
    if (fs::exists(path))
    {
        return fs::is_fifo(path);
    }
    else return -1;
}

int FMC_isOther_(const char* path)
{
    if (fs::exists(path))
    {
        return fs::is_other(path);
    }
    else return -1;
}

/*
char *FMC_readSymlink_(char *path_sym, const char * path, const int size)
{
    memset(path_sym, 0, size);
    fs::path p(path);
    if (is_symlink(p) && exists(p) && size) >= fs::read_symlink(p).string().size()) // to be changed
    {
        fs::path target = fs::read_symlink(p);
        strcpy(path_sym, target.c_str());
    }
    return path_sym;
} */

int FMC_isEmpty_(const char *path)
{
    if(fs::exists(path))
    {
        return fs::is_empty(path);
    }
    else return -1;
}

/*int FMC_createDir_(const char *path)
{
    return fs::create_directory(path);
}*/

char *FMC_getCurrentPath_(char *path, const size_t size)
{
    std::string s = fs::current_path().string();
    if (size >= s.length()+1)
    {
        memset(path, 0, size);
        strncpy(path, fs::current_path().string().c_str(), fs::current_path().string().length());
        if (strrchr(path, '/') != NULL) strcat(path, "/");
        else if (strrchr(path, '\\') != NULL) strcat(path, "\\");
        else return NULL;
        return path;
    }
    else return NULL;
}

char *FMC_getAbsolutePath_(char *path, char *buffer, const size_t size)
{

    if(fs::exists(path) && size > fs::absolute(path).string().length())
    {
        memset(buffer, 0, size);
        strncpy(buffer, fs::absolute(path).string().c_str(), fs::absolute(path).string().length());
        if (strrchr(path, '/') != NULL) strcat(buffer, "/");
        else if (strrchr(path, '\\') != NULL) strcat(buffer, "\\");
        else return NULL;
        return buffer;
    }
    else return NULL;
```

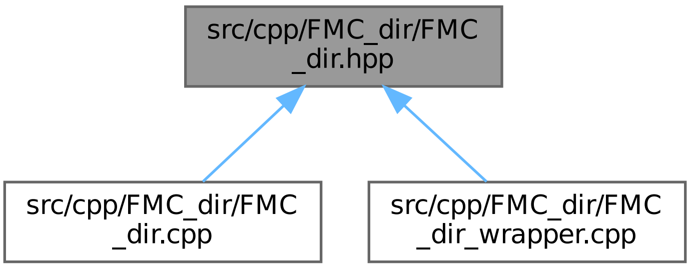
```
00170 }
```

3.8 src/cpp/FMC_dir/FMC_dir.hpp File Reference

Include dependency graph for FMC_dir.hpp:



This graph shows which files directly or indirectly include this file:



Functions

- int FMC_dirExists_ (const char ∗path)
- char ∗ FMC_getAbsolutePath_ (char ∗path, char ∗buffer, const size_t size)
- char ∗ FMC_getCurrentPath_ (char ∗path, const size_t size)
- int FMC_isBlock_ (const char ∗path)
- int FMC_isCharFile_ (const char ∗path)
- int FMC_isDir_ (const char ∗path)
- int FMC_isEmpty_ (const char ∗path)
- int FMC_isFIFO_ (const char ∗path)
- int FMC_isOther_ (const char ∗path)
- int FMC_isRegFile_ (const char ∗path)
- int FMC_isSocket_ (const char ∗path)
- int FMC_isSymLink_ (const char ∗path)

3.8.1 Function Documentation

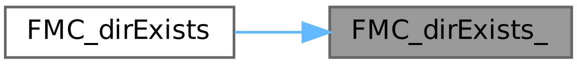
3.8.1.1 FMC_dirExists_()

```
int FMC_dirExists_ (
             const char * path )
```

Definition at line 38 of file FMC_dir.cpp.

Referenced by FMC_dirExists().

Here is the caller graph for this function:



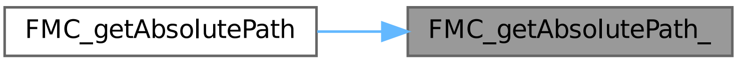
3.8.1.2 FMC_getAbsolutePath_()

```
char * FMC_getAbsolutePath_ (
             char * path,
             char * buffer,
             const size_t size )
```

Definition at line 157 of file FMC_dir.cpp.

Referenced by FMC_getAbsolutePath().

Here is the caller graph for this function:



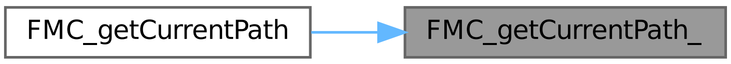
3.8.1.3 FMC_getCurrentPath_()

```
char * FMC_getCurrentPath_ (
            char * path,
            const size_t size )
```

Definition at line 142 of file FMC_dir.cpp.

Referenced by FMC_getCurrentPath().

Here is the caller graph for this function:



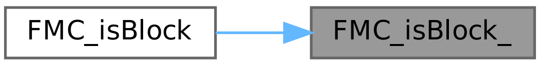
3.8.1.4 FMC_isBlock_()

```
int FMC_isBlock_ (
            const char * path )
```

Definition at line 70 of file FMC_dir.cpp.

Referenced by FMC_isBlock().

Here is the caller graph for this function:



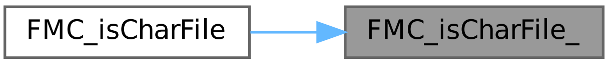
3.8.1.5 FMC_isCharFile_()

```
int FMC_isCharFile_ (
             const char * path )
```

Definition at line 79 of file FMC_dir.cpp.

Referenced by FMC_isCharFile().

Here is the caller graph for this function:



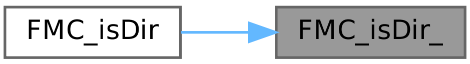
3.8.1.6 FMC_isDir_()

```
int FMC_isDir_ (
             const char * path )
```

Definition at line 43 of file FMC_dir.cpp.

Referenced by FMC_isDir().

Here is the caller graph for this function:



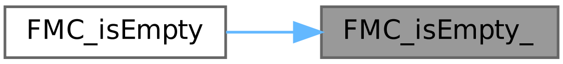
3.8.1.7 FMC_isEmpty_()

```
int FMC_isEmpty_ (
             const char * path )
```

Definition at line 128 of file FMC_dir.cpp.

Referenced by FMC_isEmpty().

Here is the caller graph for this function:



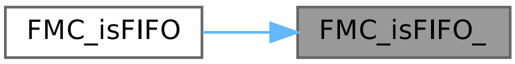
3.8.1.8 FMC_isFIFO_()

```
int FMC_isFIFO_ (
             const char * path )
```

Definition at line 97 of file FMC_dir.cpp.

Referenced by FMC_isFIFO().

Here is the caller graph for this function:



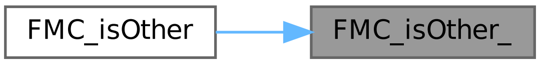
3.8.1.9 FMC_isOther_()

```
int FMC_isOther_ (
             const char * path )
```

Definition at line 106 of file FMC_dir.cpp.

Referenced by FMC_isOther().

Here is the caller graph for this function:



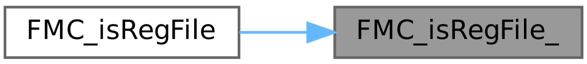
3.8.1.10 FMC_isRegFile_()

```
int FMC_isRegFile_ (
             const char * path )
```

Definition at line 52 of file FMC_dir.cpp.

Referenced by FMC_isRegFile().

Here is the caller graph for this function:



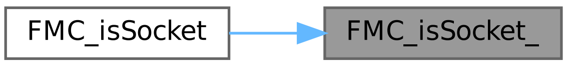
3.8.1.11 FMC_isSocket_()

```
int FMC_isSocket_ (
             const char * path )
```

Definition at line 88 of file FMC_dir.cpp.

Referenced by FMC_isSocket().

Here is the caller graph for this function:



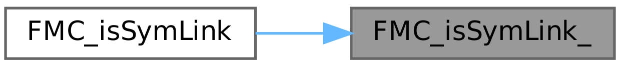
3.8.1.12 FMC_isSymLink_()

```
int FMC_isSymLink_ (
             const char * path )
```

Definition at line 61 of file FMC_dir.cpp.

Referenced by FMC_isSymLink().

Here is the caller graph for this function:



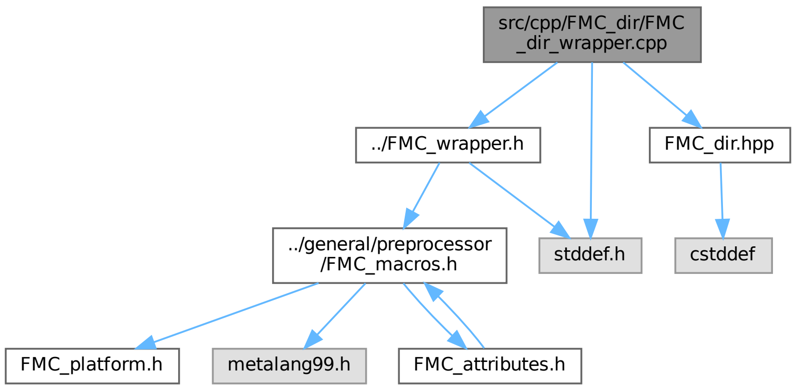
3.9 FMC_dir.hpp

Go to the documentation of this file.

```
/*

MIT License

Copyright (c) 2022-2023 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/

#ifndef FMC_DIR_HPP
#define FMC_DIR_HPP

#include <cstddef>

int FMC_dirExists_(const char *path);
int FMC_isDir_(const char *path);
int FMC_isRegFile_(const char *path);
int FMC_isSymLink_(const char *path);
int FMC_isBlock_(const char* path);
int FMC_isCharFile_(const char* path);
int FMC_isSocket_(const char* path);
int FMC_isFIFO_(const char* path);
int FMC_isOther_(const char* path);
//char *FMC_readSymlink_(char *path_sym, const char * path);
int FMC_isEmpty_(const char *path);
//int FMC_createDir_(const char *path);
char *FMC_getCurrentPath_(char *path, const size_t size);
char *FMC_getAbsolutePath_(char *path, char *buffer, const size_t size);

#endif // FMC_DIR_HPP
```

3.10 src/cpp/FMC_dir/FMC_dir_wrapper.cpp File Reference

Include dependency graph for FMC_dir_wrapper.cpp:



Functions

- int FMC_dirExists (const char ∗path)

  Checks if a directory exists.
- char ∗ FMC_getAbsolutePath (char ∗path, char ∗buffer, const size_t size)

  This function converts a relative path into an absolute one.
- char ∗ FMC_getCurrentPath (char ∗path, const size_t size)

  This function is equivalent to $PWD in bash.
- int FMC_isBlock (const char ∗path)

  Checks if a path is a block device.
- int FMC_isCharFile (const char ∗path)

  Checks if a path is a character device.
- int FMC_isDir (const char ∗path)

  Checks if a path is a directory.
- int FMC_isEmpty (const char ∗path)

  Checks if a directory is empty.
- int FMC_isFIFO (const char ∗path)

  Checks if a path is a FIFO.
- int FMC_isOther (const char ∗path)

  Checks if a path is of an unknown type.
- int FMC_isRegFile (const char ∗path)

  Checks if a path is a regular file.
- int FMC_isSocket (const char ∗path)

  Checks if a path is a socket.
- int FMC_isSymLink (const char ∗path)

  Checks if a path is a symbolic link.

3.10.1 Function Documentation

3.10.1.1 FMC_dirExists()

```
int FMC_dirExists (
             const char * path )
```

Checks if a directory exists.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

in	*path*	The path whose existence is to be checked.

Returns

An integer value.

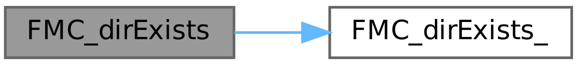
Return values

1	if the directory exists.
0	if the directory does not exist.

Definition at line 35 of file FMC_dir_wrapper.cpp.

References FMC_dirExists_().

Here is the call graph for this function:



3.10.1.2 FMC_getAbsolutePath()

```
char * FMC_getAbsolutePath (
             char * path,
             char * buffer,
             const size_t size )
```

This function converts a relative path into an absolute one.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

`in`	*path*	The path to convert.
`out`	*buffer*	The memory buffer to store the absolute path.
`in`	*size*	The size of the memory buffer.

Returns

A pointer to the memory buffer.

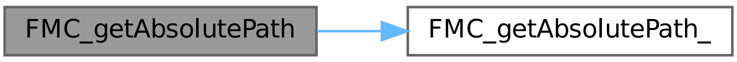
Return values

NULL	if an error occured.
buffer	The pointer to the buffer after the call if the function succeeded.

Definition at line 92 of file FMC_dir_wrapper.cpp.

References FMC_getAbsolutePath_().

Here is the call graph for this function:



3.10.1.3 FMC_getCurrentPath()

```
char * FMC_getCurrentPath (
             char * path,
             const size_t size )
```

This function is equivalent to $PWD in bash.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

out	*path*	The memory buffer to store the current path.
in	*size*	The size of the memory buffer.

Returns

A pointer to the memory buffer.

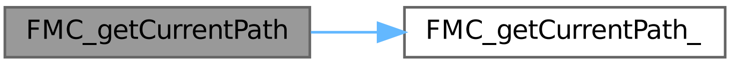
Return values

NULL	if an error occured.
path	The pointer to path after the call if the function succeeded.

Definition at line 87 of file FMC_dir_wrapper.cpp.

References FMC_getCurrentPath_().

Here is the call graph for this function:



3.10.1.4 FMC_isBlock()

```
int FMC_isBlock (
            const char * path )
```

Checks if a path is a block device.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

`in`	*path*	The path to check.

Returns

An integer value.

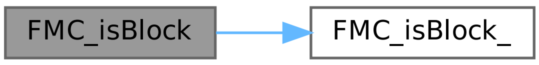
Return values

1	if the path is a block device.
0	if the path is not a block device.
-1	if an error occured.

Definition at line 55 of file FMC_dir_wrapper.cpp.

References FMC_isBlock_().

Here is the call graph for this function:



3.10.1.5 FMC_isCharFile()

```
int FMC_isCharFile (
            const char * path )
```

Checks if a path is a character device.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

`in`	*path*	The path to check.

Returns

An integer value.

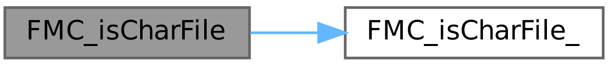
Return values

1	if the path is a character device.
0	if the path is not a character device.
-1	if an error occured.

Definition at line 60 of file FMC_dir_wrapper.cpp.

References FMC_isCharFile_().

Here is the call graph for this function:



3.10.1.6 FMC_isDir()

```
int FMC_isDir (
             const char * path )
```

Checks if a path is a directory.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

in	*path*	The path to check.

Returns

An integer value.

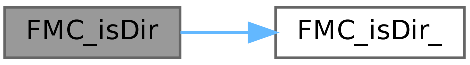
Return values

1	if the path is a directory.
0	if the path is not a directory.
-1	if an error occured.

Definition at line 40 of file FMC_dir_wrapper.cpp.

References FMC_isDir_().

Here is the call graph for this function:



3.10.1.7 FMC_isEmpty()

```
int FMC_isEmpty (
             const char * path )
```

Checks if a directory is empty.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

in	*path*	The path to check.

Returns

An integer value.

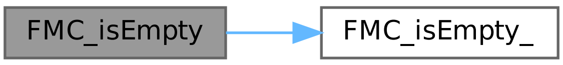
Return values

1	if the directory is empty.
0	if the directory is not empty.
-1	if an error occured.

Definition at line 80 of file FMC_dir_wrapper.cpp.

References FMC_isEmpty_().

Here is the call graph for this function:



3.10.1.8 FMC_isFIFO()

```
int FMC_isFIFO (
            const char * path )
```

Checks if a path is a FIFO.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

in	*path*	The path to check.

Returns

An integer value.

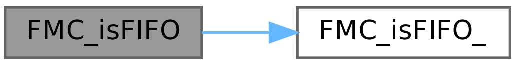
Return values

1	if the path is a FIFO.
0	if the path is not a FIFO.
-1	if an error occured.

Definition at line 70 of file FMC_dir_wrapper.cpp.

References FMC_isFIFO_().

Here is the call graph for this function:



3.10.1.9 FMC_isOther()

```
int FMC_isOther (
             const char * path )
```

Checks if a path is of an unknown type.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

in	*path*	The path to check.

Returns

An integer value.

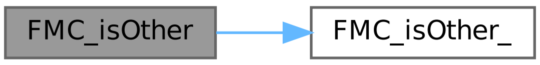
Return values

1	if the path is of an unknown type.
0	if the path is not of an unknown type.
-1	if an error occured.

Definition at line 75 of file FMC_dir_wrapper.cpp.

References FMC_isOther_().

Here is the call graph for this function:



3.10.1.10 FMC_isRegFile()

```
int FMC_isRegFile (
            const char * path )
```

Checks if a path is a regular file.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

`in`	*path*	The path to check.

Returns

An integer value.

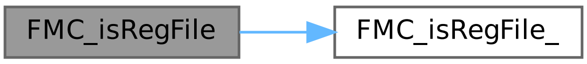
Return values

1	if the path is a regular file.
0	if the path is not a regular file.
-1	if an error occured.

Definition at line 45 of file FMC_dir_wrapper.cpp.

References FMC_isRegFile_().

Here is the call graph for this function:



3.10.1.11 FMC_isSocket()

```
int FMC_isSocket (
             const char * path )
```

Checks if a path is a socket.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

`in`	*path*	The path to check.

Returns

An integer value.

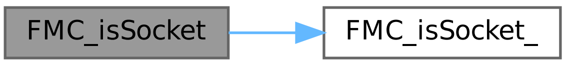
Return values

1	if the path is a socket.
0	if the path is not a socket.
-1	if an error occured.

Definition at line 65 of file FMC_dir_wrapper.cpp.

References FMC_isSocket_().

Here is the call graph for this function:



3.10.1.12 FMC_isSymLink()

```
int FMC_isSymLink (
             const char * path )
```

Checks if a path is a symbolic link.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

in	*path*	The path to check.

Returns

An integer value.

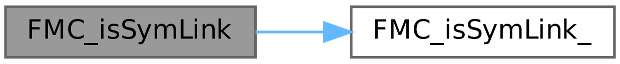
Return values

1	if the path is a symbolic link.
0	if the path is not a symbolic link.
-1	if an error occured.

Definition at line 50 of file FMC_dir_wrapper.cpp.

References FMC_isSymLink_().

Here is the call graph for this function:



3.11 FMC_dir_wrapper.cpp

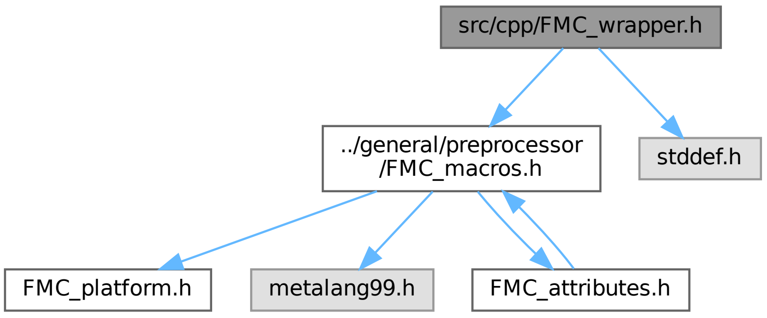
Go to the documentation of this file.

```
/*

MIT License

Copyright (c) 2022-2023 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/

#include "../FMC_wrapper.h"
#include "FMC_dir.hpp"
```

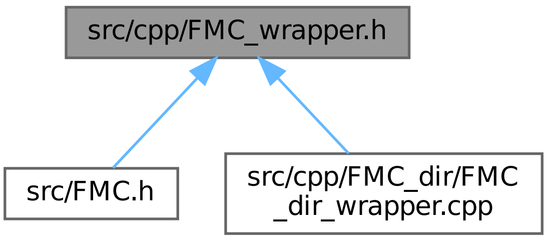
```
#include <stddef.h>

#ifdef __cplusplus
extern "C" {
#endif

FMC_SHARED int FMC_dirExists(const char *path)
{
    return FMC_dirExists_(path);
}

FMC_SHARED int FMC_isDir(const char *path)
{
    return FMC_isDir_(path);
}

FMC_SHARED int FMC_isRegFile(const char *path)
{
    return FMC_isRegFile_(path);
}

FMC_SHARED int FMC_isSymLink(const char *path)
{
    return FMC_isSymLink_(path);
}

FMC_SHARED int FMC_isBlock(const char* path)
{
    return FMC_isBlock_(path);
}

FMC_SHARED int FMC_isCharFile(const char* path)
{
    return FMC_isCharFile_(path);
}

FMC_SHARED int FMC_isSocket(const char* path)
{
    return FMC_isSocket_(path);
}

FMC_SHARED int FMC_isFIFO(const char* path)
{
    return FMC_isFIFO_(path);
}

FMC_SHARED int FMC_isOther(const char* path)
{
    return FMC_isOther_(path);
}

FMC_SHARED int FMC_isEmpty(const char *path)
{
    return FMC_isEmpty_(path);
}

//FMC_SHARED int FMC_createDir_(const char *path);

FMC_SHARED char *FMC_getCurrentPath(char *path, const size_t size)
{
    return FMC_getCurrentPath_(path, size);
}

FMC_SHARED char *FMC_getAbsolutePath(char *path, char *buffer, const size_t size)
{
    return FMC_getAbsolutePath_(path, buffer, size);
}

#ifdef __cplusplus
}
#endif
```

3.12 src/cpp/FMC_wrapper.h File Reference

Include dependency graph for FMC_wrapper.h:



This graph shows which files directly or indirectly include this file:



Functions

- int FMC_dirExists (const char ∗path)

  Checks if a directory exists.
- char ∗ FMC_getAbsolutePath (char ∗path, char ∗buffer, const size_t size)

  This function converts a relative path into an absolute one.
- char ∗ FMC_getCurrentPath (char ∗path, const size_t size)

  This function is equivalent to $PWD in bash.
- int FMC_isBlock (const char ∗path)

  Checks if a path is a block device.
- int FMC_isCharFile (const char ∗path)

  Checks if a path is a character device.
- int FMC_isDir (const char ∗path)

  Checks if a path is a directory.

- int FMC_isEmpty (const char ∗path)

  Checks if a directory is empty.
- int FMC_isFIFO (const char ∗path)

  Checks if a path is a FIFO.
- int FMC_isOther (const char ∗path)

  Checks if a path is of an unknown type.
- int FMC_isRegFile (const char ∗path)

  Checks if a path is a regular file.
- int FMC_isSocket (const char ∗path)

  Checks if a path is a socket.
- int FMC_isSymLink (const char ∗path)

  Checks if a path is a symbolic link.

3.12.1 Function Documentation

3.12.1.1 FMC_dirExists()

```
int FMC_dirExists (
            const char * path )
```

Checks if a directory exists.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

in	*path*	The path whose existence is to be checked.

Returns

An integer value.

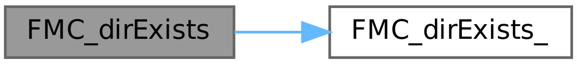
Return values

1	if the directory exists.
0	if the directory does not exist.

Definition at line 35 of file FMC_dir_wrapper.cpp.

References FMC_dirExists_().

Here is the call graph for this function:



3.12.1.2 FMC_getAbsolutePath()

```
char * FMC_getAbsolutePath (
            char * path,
            char * buffer,
            const size_t size )
```

This function converts a relative path into an absolute one.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

in	*path*	The path to convert.
out	*buffer*	The memory buffer to store the absolute path.
in	*size*	The size of the memory buffer.

Returns

A pointer to the memory buffer.

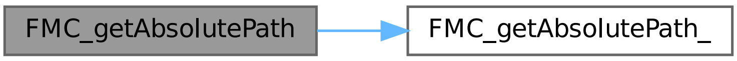
Return values

NULL	if an error occured.
buffer	The pointer to the buffer after the call if the function succeeded.

Definition at line 92 of file FMC_dir_wrapper.cpp.

References FMC_getAbsolutePath_().

Here is the call graph for this function:



3.12.1.3 FMC_getCurrentPath()

```
char * FMC_getCurrentPath (
             char * path,
             const size_t size )
```

This function is equivalent to $PWD in bash.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

out	*path*	The memory buffer to store the current path.
in	*size*	The size of the memory buffer.

Returns

A pointer to the memory buffer.

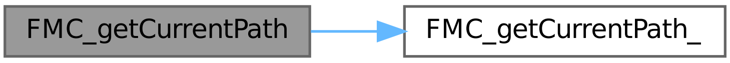
Return values

NULL	if an error occured.
path	The pointer to path after the call if the function succeeded.

Definition at line 87 of file FMC_dir_wrapper.cpp.

References FMC_getCurrentPath_().

Here is the call graph for this function:



3.12.1.4 FMC_isBlock()

```
int FMC_isBlock (
             const char * path )
```

Checks if a path is a block device.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

in	*path*	The path to check.

Returns

An integer value.

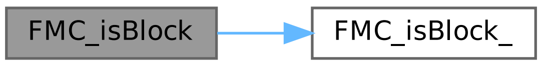
Return values

1	if the path is a block device.
0	if the path is not a block device.
-1	if an error occured.

Definition at line 55 of file FMC_dir_wrapper.cpp.

References FMC_isBlock_().

Here is the call graph for this function:



3.12.1.5 FMC_isCharFile()

```
int FMC_isCharFile (
            const char * path )
```

Checks if a path is a character device.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

in	*path*	The path to check.

Returns

An integer value.

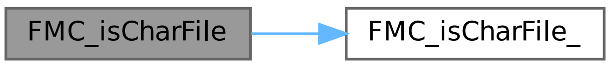
Return values

1	if the path is a character device.
0	if the path is not a character device.
-1	if an error occured.

Definition at line 60 of file FMC_dir_wrapper.cpp.

References FMC_isCharFile_().

Here is the call graph for this function:



3.12.1.6 FMC_isDir()

```
int FMC_isDir (
            const char * path )
```

Checks if a path is a directory.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

in	*path*	The path to check.

Returns

An integer value.

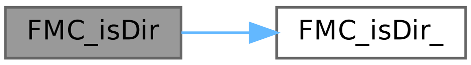
Return values

1	if the path is a directory.
0	if the path is not a directory.
-1	if an error occured.

Definition at line 40 of file FMC_dir_wrapper.cpp.

References FMC_isDir_().

Here is the call graph for this function:



3.12.1.7 FMC_isEmpty()

```
int FMC_isEmpty (
             const char * path )
```

Checks if a directory is empty.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

in	*path*	The path to check.

Returns

An integer value.

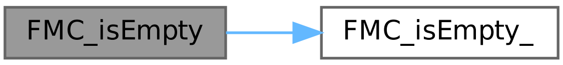
Return values

1	if the directory is empty.
0	if the directory is not empty.
-1	if an error occured.

Definition at line 80 of file FMC_dir_wrapper.cpp.

References FMC_isEmpty_().

Here is the call graph for this function:



3.12.1.8 FMC_isFIFO()

```
int FMC_isFIFO (
             const char * path )
```

Checks if a path is a FIFO.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

in	*path*	The path to check.

Returns

An integer value.

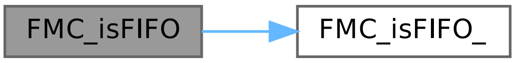
Return values

1	if the path is a FIFO.
0	if the path is not a FIFO.
-1	if an error occured.

Definition at line 70 of file FMC_dir_wrapper.cpp.

References FMC_isFIFO_().

Here is the call graph for this function:



3.12.1.9 FMC_isOther()

```
int FMC_isOther (
            const char * path )
```

Checks if a path is of an unknown type.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

in	*path*	The path to check.

Returns

An integer value.

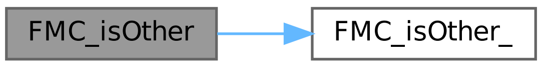
Return values

1	if the path is of an unknown type.
0	if the path is not of an unknown type.
-1	if an error occured.

Definition at line 75 of file FMC_dir_wrapper.cpp.

References FMC_isOther_().

Here is the call graph for this function:



3.12.1.10 FMC_isRegFile()

```
int FMC_isRegFile (
             const char * path )
```

Checks if a path is a regular file.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

in	*path*	The path to check.

Returns

An integer value.

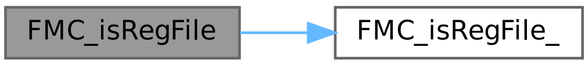
Return values

1	if the path is a regular file.
0	if the path is not a regular file.
-1	if an error occured.

Definition at line 45 of file FMC_dir_wrapper.cpp.

References FMC_isRegFile_().

Here is the call graph for this function:



3.12.1.11 FMC_isSocket()

```
int FMC_isSocket (
          const char * path )
```

Checks if a path is a socket.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

in	*path*	The path to check.

Returns

An integer value.

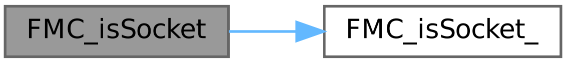
Return values

1	if the path is a socket.
0	if the path is not a socket.
-1	if an error occured.

Definition at line 65 of file FMC_dir_wrapper.cpp.

References FMC_isSocket_().

Here is the call graph for this function:



3.12.1.12 FMC_isSymLink()

```
int FMC_isSymLink (
             const char * path )
```

Checks if a path is a symbolic link.

Author

Axel PASCON

Date

2023

This function is a wrapper around the C++ filesystem library assciated function.

Parameters

in	*path*	The path to check.

Returns

An integer value.

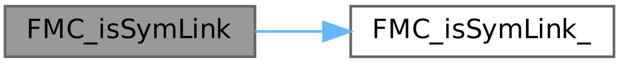
Return values

1	if the path is a symbolic link.
0	if the path is not a symbolic link.
-1	if an error occured.

Definition at line 50 of file FMC_dir_wrapper.cpp.

References FMC_isSymLink_().

Here is the call graph for this function:



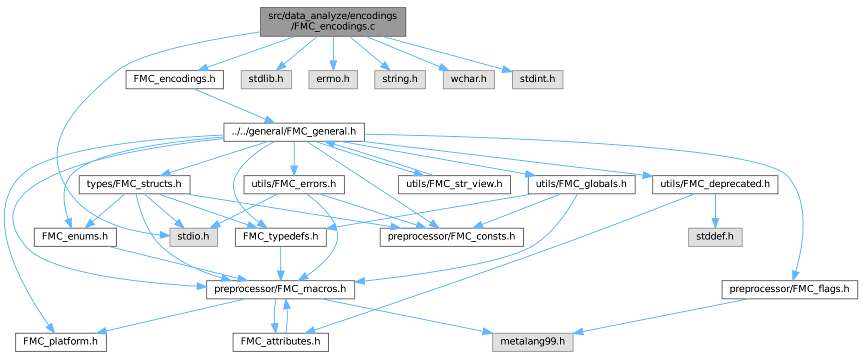
3.13 FMC_wrapper.h

Go to the documentation of this file.

```
/*

MIT License

Copyright (c) 2022-2023 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/

#ifndef FMC_WRAPPER_H
#define FMC_WRAPPER_H

#ifdef __cplusplus
extern "C" {
#endif

#include "../general/preprocessor/FMC_macros.h"

#include <stddef.h>

// FMC_dir
FMC_SHARED int FMC_dirExists(const char *path);
FMC_SHARED int FMC_isDir(const char *path);
FMC_SHARED int FMC_isRegFile(const char *path);
FMC_SHARED int FMC_isSymLink(const char *path);
FMC_SHARED int FMC_isBlock(const char* path);
FMC_SHARED int FMC_isCharFile(const char* path);
FMC_SHARED int FMC_isSocket(const char* path);
FMC_SHARED int FMC_isFIFO(const char* path);
FMC_SHARED int FMC_isOther(const char* path);
FMC_SHARED int FMC_isEmpty(const char *path);
FMC_SHARED char *FMC_getCurrentPath(char *path, const size_t size);
FMC_SHARED char *FMC_getAbsolutePath(char *path, char *buffer, const size_t size);
// !FMC_dir

#ifdef __cplusplus
}
#endif


#endif // FMC_WRAPPER_H
```

3.14 src/data_analyze/encodings/FMC_encodings.c File Reference

Include dependency graph for FMC_encodings.c:



Macros

- #define __STDC_WANT_LIB_EXT1__ 1

Functions

- FMC_FUNC_CONST FMC_FUNC_ALWAYS_INLINE FMC_Encodings FMC_checkEncodingFlag (unsigned int encoding)
- FMC_Encodings FMC_getEncoding (FILE *file)

3.14.1 Macro Definition Documentation

3.14.1.1 __STDC_WANT_LIB_EXT1__

```
#define __STDC_WANT_LIB_EXT1__ 1
```

Definition at line 26 of file FMC_encodings.c.

3.14.2 Function Documentation

3.14.2.1 FMC_checkEncodingFlag()

```
FMC_FUNC_CONST FMC_FUNC_ALWAYS_INLINE FMC_Encodings FMC_checkEncodingFlag (
             unsigned int encoding )  [inline]
```

Definition at line 204 of file FMC_encodings.c.

References ASCII, ascii, error, unknown, UTF16_BE, utf16_be, UTF16_LE, utf16_le, UTF32_BE, utf32_be, UTF32_LE, utf32_le, UTF8, utf8, UTF8_BOM, and utf8_bom.

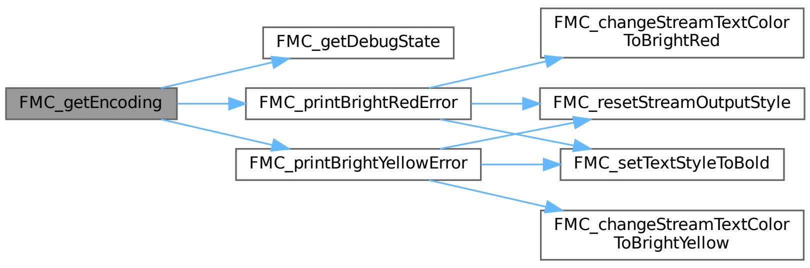
3.14.2.2 FMC_getEncoding()

```
FMC_Encodings FMC_getEncoding (
             FILE * file )
```

Definition at line 36 of file FMC_encodings.c.

References ascii, error, FMC_getDebugState(), FMC_makeMsg, FMC_printBrightRedError(), FMC_printBrightYellowError(), unknown, utf16_be, utf16_le, utf32_be, utf32_le, utf8, and utf8_bom.

Here is the call graph for this function:


FMC_getEncoding
FMC_getDebugState
FMC_printBrightRedError
FMC_printBrightYellowError
FMC_changeStreamTextColor ToBrightRed
FMC_resetStreamOutputStyle
FMC_setTextStyleToBold
FMC_changeStreamTextColor ToBrightYellow

3.15 FMC_encodings.c

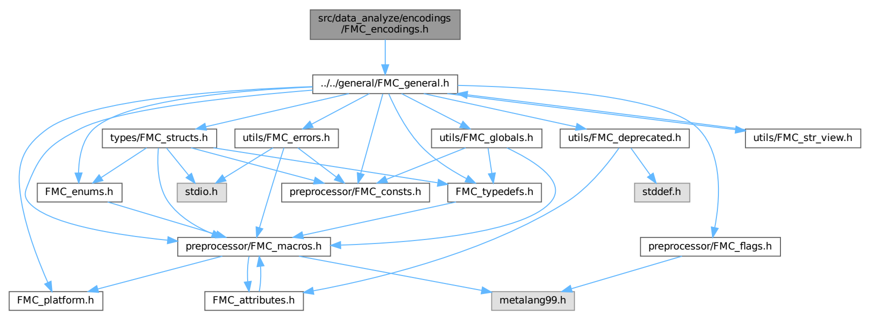
Go to the documentation of this file.

```
/*

MIT License

Copyright (c) 2023 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
```

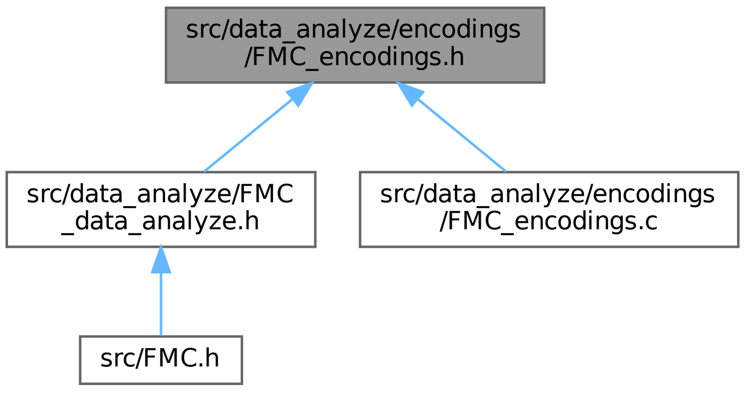
```
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/
#define __STDC_WANT_LIB_EXT1__ 1
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <string.h>
#include <wchar.h> // fwide
#include <stdint.h>

#include "FMC_encodings.h"

FMC_SHARED FMC_FUNC_WARN_UNUSED_RESULT FMC_FUNC_NONNULL(1) FMC_Encodings FMC_getEncoding(FILE *file)
{
    #pragma GCC diagnostic ignored "-Wnonnull-compare" // get an error at compile time without this (because of attribute nonnull)
    if (file == NULL)
    {
        if (FMC_getDebugState())
        {
            FMC_makeMsg(err_null, 4, "ERROR : ", "In function : ", __func__, ". The provided file must not be NULL.");
            FMC_printBrightRedError(stderr, err_null);
        }
        return error;
    }
    #pragma GCC diagnostic pop

    // check orientation
    if (fwide(file, -1) >= 0)
    {
        if (FMC_getDebugState())
        {
            FMC_makeMsg(err_wide, 4, "ERROR : ", "In function : ", __func__, ". The provided file must be opened with by orientation.");
            FMC_printBrightRedError(stderr, err_wide);
        }
        return error;
    }

    long long sizeOfFile = 0;
    if(fseek(file, 0, SEEK_END))
    {
        FMC_makeMsg(err_seek_1, 4, "FMC INTERNAL ERROR : ", "In function : ", __func__, ". fseek failure.");
        FMC_printBrightRedError(stderr, err_seek_1);
        return error;
    }
    errno = 0;
    sizeOfFile = ftell(file);
    if (errno || sizeOfFile == -1L)
    {
        FMC_makeMsg(err_tell, 5, "FMC INTERNAL ERROR : ", "In function : ", __func__, ". ftell failure.", strerror(errno));
        FMC_printBrightRedError(stderr, err_tell);
        return error;
    }

    rewind(file);
    char buff[4] = {0};
    // 1st if
    if(sizeOfFile < 0) // no error, must have overflowed
    {
        sizeOfFile = (typeof(sizeOfFile)) SIZE_MAX;
        size_t ret = fread(buff, 1, 4, file);
        if(ret != 4) goto check_error_type_1;
        else if (ret == 4) goto end_check_1;
        else return error;
    }

    // 2nd if
    else if (sizeOfFile <= 4 && sizeOfFile >= 0)
    {
        size_t ret = fread(buff, 1, (size_t)sizeOfFile, file); // harmless cast here because 0 <= sizeOfFile <= 4
        if(ret != (size_t) sizeOfFile) goto check_error_type_1;
        else if (ret == (size_t) sizeOfFile) goto end_check_1;
        else return error;

        check_error_type_1 :
        FMC_LABEL_COLD;
        if (feof(file))
        {
            FMC_makeMsg(err_feof, 4, "FMC INTERNAL ERROR : ", "In function : ", __func__, ". EOF indicator set.");
            FMC_printBrightRedError(stderr, err_feof);
            return error;
        }
        else if (ferror(file))
        {
            FMC_makeMsg(err_ferror, 5, "FMC INTERNAL ERROR : ", "In function : ", __func__, ". Error indicator set.", strerror(errno));
            FMC_printBrightRedError(stderr, err_ferror);
            return error;
        }
        else
        {
            FMC_makeMsg(err_fread, 4, "FMC INTERNAL ERROR : ", "In function : ", __func__, ". fread failure.");
            FMC_printBrightRedError(stderr, err_fread);
            return error;
        }

    }

    // 3rd if
    else if(fread(buff, 1, 4, file) != 4)
    {
        if (feof(file))
        {
            FMC_makeMsg(err_feof, 4, "FMC INTERNAL ERROR : ", "In function : ", __func__, ". EOF indicator set.");
            FMC_printBrightRedError(stderr, err_feof);
            return error;
        }
        else if (ferror(file))
        {
            FMC_makeMsg(err_ferror, 5, "FMC INTERNAL ERROR : ", "In function : ", __func__, ". Error indicator set.", strerror(errno));
            FMC_printBrightRedError(stderr, err_ferror);
            return error;
        }
        else
        {
            FMC_makeMsg(err_fread, 4, "FMC INTERNAL ERROR : ", "In function : ", __func__, ". fread failure.");
            FMC_printBrightRedError(stderr, err_fread);
            return error;
        }
    }

    end_check_1 :
    FMC_LABEL_HOT;
    if (sizeOfFile >= 3 && (unsigned char) buff[0] == 0xEF && (unsigned char) buff[1] == 0xBB && (unsigned char) buff[2] == 0xBF)
    {
        rewind(file);
        return utf8_bom;
    }
    else if (sizeOfFile >= 2 && (unsigned char) buff[0] == 0xFF && (unsigned char) buff[1] == 0xFE)
    {
        rewind(file);
        return utf16_le;
    }
    else if (sizeOfFile >= 2 && (unsigned char) buff[0] == 0xFE && (unsigned char) buff[1] == 0xFF)
    {
        rewind(file);
        return utf16_be;
    }
    else if (sizeOfFile >= 4 && (unsigned char) buff[0] == 0x00 && (unsigned char) buff[1] == 0x00 && (unsigned char) buff[2] == 0xFE && (unsigned char) buff[3] == 0xFF)
    {
        rewind(file);
        return utf32_be;
    }
    else if (sizeOfFile >= 4 && (unsigned char) buff[0] == 0xFF && (unsigned char) buff[1] == 0xFE && (unsigned char) buff[2] == 0x00 && (unsigned char) buff[3] == 0x00)
    {
        rewind(file);
        return utf32_le;
    }
    else
    {
        rewind(file);
        if (sizeOfFile == 0)
        {
            rewind(file);
            if (FMC_getDebugState())
            {
                FMC_makeMsg(err_empty, 4, "WARNING : ", "In function : ", __func__, ". The provided
    file is empty.");
                FMC_printBrightYellowError(stderr, err_empty);
            }
            return unknown;
        }

        char currentChar = 0;
        size_t cpt = 0;
        while((currentChar = (char)fgetc(file)) != EOF)
        {
            if (currentChar != EOF &&  (unsigned char) currentChar > 127)
            {
                rewind(file);
                return utf8;
            }
            cpt++;
            if ((long long) cpt >= sizeOfFile)
            {
                break;
            }
        }
        rewind(file);
        return ascii;
    }
}

FMC_SHARED FMC_FUNC_CONST FMC_FUNC_ALWAYS_INLINE inline FMC_Encodings FMC_checkEncodingFlag(unsigned
    int encoding)
{
    switch (encoding)
    {
        case ASCII:
            return ascii;
            break;
        case UTF8:
            return utf8;
            break;
        case UTF8_BOM:
            return utf8_bom;
            break;
        case UTF16_LE:
            return utf16_le;
            break;
        case UTF16_BE:
            return utf16_be;
            break;
        case UTF32_LE:
            return utf32_le;
            break;
        case UTF32_BE:
            return utf32_be;
            break;
        default: // TODO : add error in case of unknown encoding
            return unknown;
            break;
    }
    return error;
}

/*FMC_SHARED FMC_Char FMC_getc(FMC_File file)
{
    FMC_Char c = {.encoding = file.encoding, .comp = {.mostLeft = 0, .middleLeft = 0, .middleRight =
    0, .mostRight = 0}, .isNull = 0};
    if(file.file == NULL || file.encoding == error || file.encoding == unknown)
    {
        c.isNull = 1;
        return c;
    }
    else if (fwide(file.file, 0) > 0)
    {
        fprintf(stderr, "Error: file is wide oriented when trying to read with byte orientation\n");
        c.isNull = 1;
        return c;
    }
    else if (file.encoding == ascii)
    {
        if (!feof(file.file))
        {

        }

    }
}*/
```

3.16 src/data_analyze/encodings/FMC_encodings.h File Reference

Include dependency graph for FMC_encodings.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define FMC_ENCODINGS_H

Functions

- FMC_FUNC_CONST FMC_FUNC_ALWAYS_INLINE FMC_Encodings FMC_checkEncodingFlag (unsigned int encoding)
- FMC_Encodings FMC_getEncoding (FILE ∗file)

3.16.1 Macro Definition Documentation

3.16.1.1 FMC_ENCODINGS_H

```
#define FMC_ENCODINGS_H
```

Definition at line 30 of file FMC_encodings.h.

3.16.2 Function Documentation

3.16.2.1 FMC_checkEncodingFlag()

```
FMC_FUNC_CONST FMC_FUNC_ALWAYS_INLINE FMC_Encodings FMC_checkEncodingFlag (
             unsigned int encoding )  [inline]
```

Definition at line 204 of file FMC_encodings.c.

References ASCII, ascii, error, unknown, UTF16_BE, utf16_be, UTF16_LE, utf16_le, UTF32_BE, utf32_be, UTF32_LE, utf32_le, UTF8, utf8, UTF8_BOM, and utf8_bom.

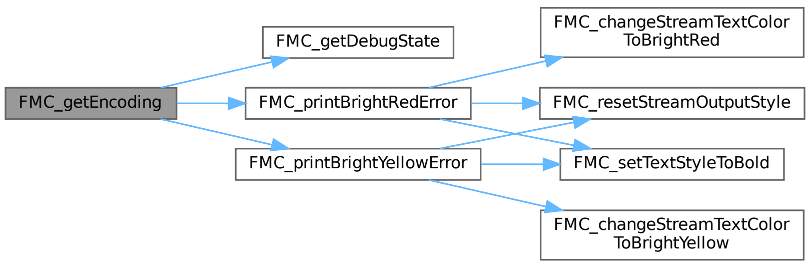
3.16.2.2 FMC_getEncoding()

```
FMC_Encodings FMC_getEncoding (
             FILE * file )
```

Definition at line 36 of file FMC_encodings.c.

References ascii, error, FMC_getDebugState(), FMC_makeMsg, FMC_printBrightRedError(), FMC_printBrightYellowError(), unknown, utf16_be, utf16_le, utf32_be, utf32_le, utf8, and utf8_bom.

Here is the call graph for this function:


FMC_getEncoding
FMC_getDebugState
FMC_printBrightRedError
FMC_printBrightYellowError
FMC_changeStreamTextColor ToBrightRed
FMC_resetStreamOutputStyle
FMC_setTextStyleToBold
FMC_changeStreamTextColor ToBrightYellow

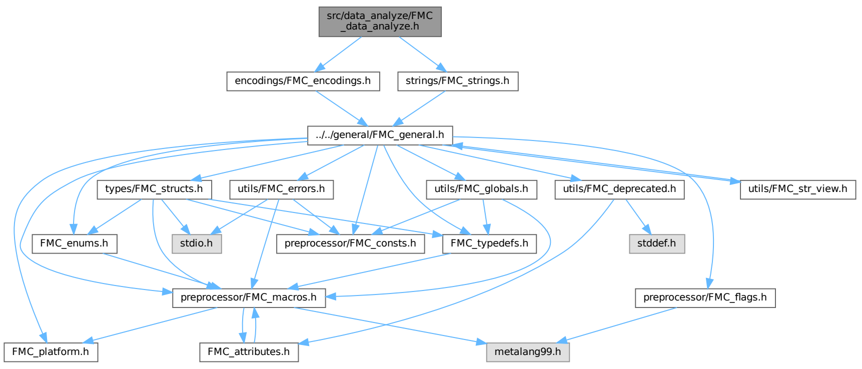
3.17 FMC_encodings.h

Go to the documentation of this file.

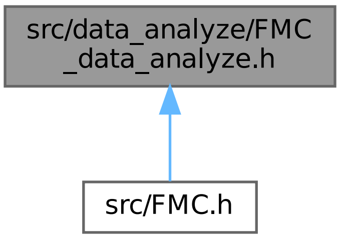
```
/*

MIT License

Copyright (c) 2023 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/

#pragma once

#ifndef FMC_ENCODINGS_H
#define FMC_ENCODINGS_H

#include "../../general/FMC_general.h"

FMC_SHARED FMC_FUNC_WARN_UNUSED_RESULT FMC_FUNC_NONNULL(1) FMC_Encodings FMC_getEncoding(FILE *file);
FMC_SHARED FMC_FUNC_CONST FMC_FUNC_ALWAYS_INLINE inline FMC_Encodings FMC_checkEncodingFlag(unsigned
int encoding);

#endif // FMC_ENCODINGS_H
```

3.18 src/data_analyze/FMC_data_analyze.h File Reference

Include dependency graph for FMC_data_analyze.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define FMC_DATA_ANALYZE_H

3.18.1 Macro Definition Documentation

3.18.1.1 FMC_DATA_ANALYZE_H

```
#define FMC_DATA_ANALYZE_H
```

Definition at line 30 of file FMC_data_analyze.h.

3.19 FMC_data_analyze.h

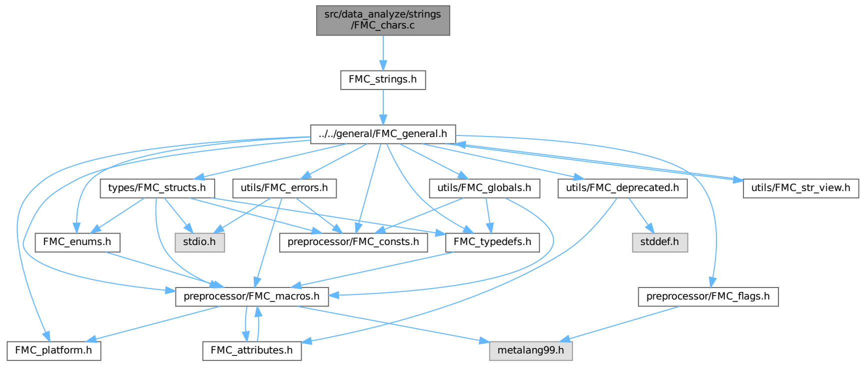
Go to the documentation of this file.

```
/*

MIT License

Copyright (c) 2022 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

```

```
*/

#pragma once

#ifndef FMC_DATA_ANALYZE_H
#define FMC_DATA_ANALYZE_H


#include "encodings/FMC_encodings.h"
#include "strings/FMC_strings.h"

#endif // FMC_DATA_ANALYZE_H
```

3.20 src/data_analyze/strings/FMC_chars.c File Reference

Include dependency graph for FMC_chars.c:



3.21 FMC_chars.c

Go to the documentation of this file.

```
/*

MIT License

Copyright (c) 2022-2023 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/

#include "FMC_strings.h"

/* FMC_SHARED FMC_FUNC_NONNULL(1) FMC_FUNC_HOT FMC_Char FMC_getChar(FMC_File *file)
{
    #pragma GCC diagnostic ignored "-Wnonnull-compare"
    if (file == NULL)
    {
        FMC_makeMsg(err_nullarg, 3, "ERROR : In function : ", __func__, " : the provided file pointer is NULL");
        FMC_printRedError(stderr, err_nullarg);
    }
    #pragma GCC diagnostic pop
    if (!file->isOpened || file->name)
    {

    }

} */
```

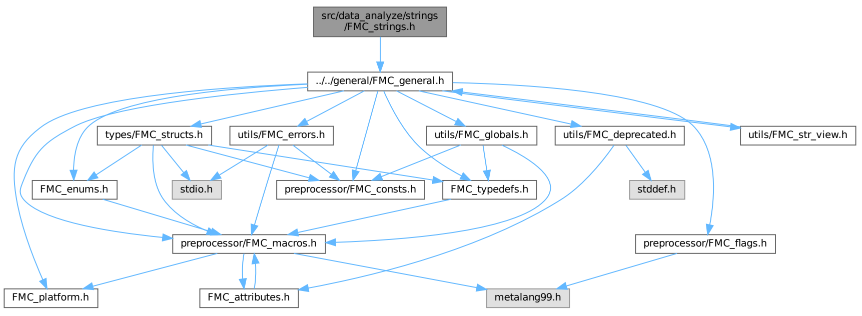
3.22 src/data_analyze/strings/FMC_strings.c File Reference

3.23 FMC_strings.c

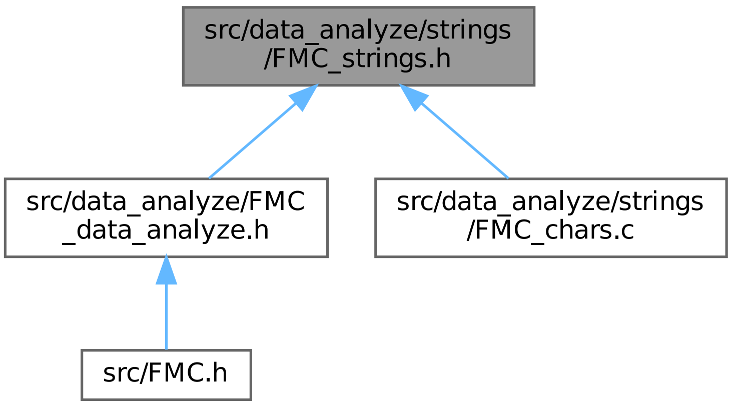
Go to the documentation of this file.

3.24 src/data_analyze/strings/FMC_strings.h File Reference

Include dependency graph for FMC_strings.h:



This graph shows which files directly or indirectly include this file:


src/data_analyze/strings/FMC_strings.h
src/data_analyze/FMC_data_analyze.h
src/data_analyze/strings/FMC_chars.c
src/FMC.h

Macros

- #define FMC_STRINGS_H

3.24.1 Macro Definition Documentation

3.24.1.1 FMC_STRINGS_H

```
#define FMC_STRINGS_H
```

Definition at line 30 of file FMC_strings.h.

3.25 FMC_strings.h

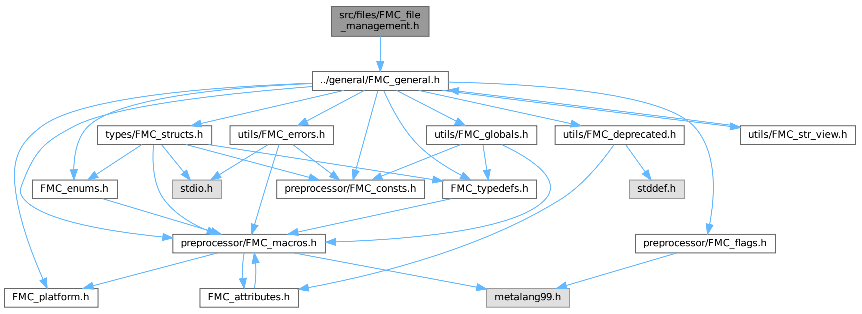
Go to the documentation of this file.

```
00001 /*
00002
00003 MIT License
00004
00005 Copyright (c) 2022 Axel PASCON
00006
00007 Permission is hereby granted, free of charge, to any person obtaining a copy
00008 of this software and associated documentation files (the "Software"), to deal
00009 in the Software without restriction, including without limitation the rights
00010 to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
00011 copies of the Software, and to permit persons to whom the Software is
00012 furnished to do so, subject to the following conditions:
00013
00014 The above copyright notice and this permission notice shall be included in all
00015 copies or substantial portions of the Software.
00016
```

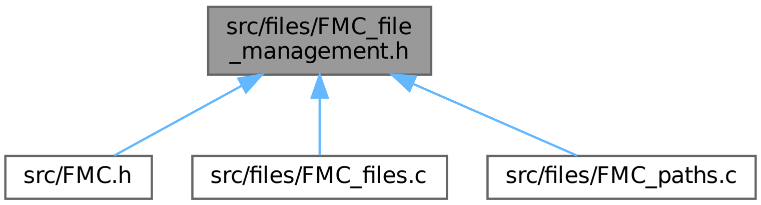
```
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/

#pragma once

#ifndef FMC_STRINGS_H
#define FMC_STRINGS_H

#include "../../general/FMC_general.h"


#endif // FMC_STRINGS_H
```

3.26 src/files/FMC_file_management.h File Reference

Include dependency graph for FMC_file_management.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define FMC_FILE_MANAGEMENT_H

Functions

- char ∗ FMC_cutFilename (const char ∗restrict const path, char ∗restrict dirs, const size_t dirs_size)
- char ∗ FMC_extractFilename (const char ∗restrict const path, char ∗restrict filename, const size_t filename←_size)

  Gets the filename from a complete path.
- char ∗ FMC_getExtension (const char ∗restrict const path, char ∗restrict ext, const size_t ext_size)

3.26.1 Macro Definition Documentation

3.26.1.1 FMC_FILE_MANAGEMENT_H

```
#define FMC_FILE_MANAGEMENT_H
```

Definition at line 30 of file FMC_file_management.h.

3.26.2 Function Documentation

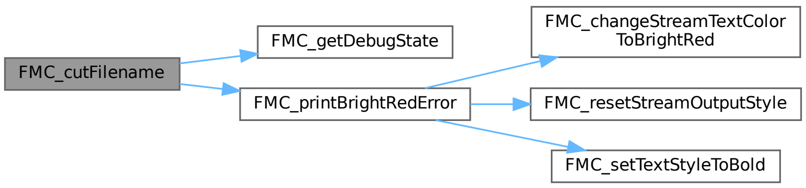
3.26.2.1 FMC_cutFilename()

```
char * FMC_cutFilename (
            const char *restrict const path,
            char *restrict dirs,
            const size_t dirs_size )
```

Definition at line 109 of file FMC_paths.c.

References FMC_getDebugState(), FMC_makeMsg, FMC_printBrightRedError(), MAX_FEXT_SIZE, MAX_FNAME_SIZE, and MAX_FPATH_SIZE.

Here is the call graph for this function:



3.26.2.2 FMC_extractFilename()

```
char * FMC_extractFilename (
            const char *restrict const path,
            char *restrict filename,
            const size_t filename_size )
```

Gets the filename from a complete path.

Author

Axel PASCON

Date

2023

Basically, this function only detects the last '/' or '\' character. For example, if the path is "C:\\Users\\someone\\←
Documents\\MyFile.txt", the function will return "MyFile.txt". If the path is "/home/someone/Desktop", then Desktop will be considered as the filename. This function is designed to only operate on strings, and do not check if the path is valid, is a file or a directory, etc.

Parameters

in	*path*	The path to extract the filename from.
out	*filename*	The buffer where the filename will be stored.
in	*filename_size*	The size of the filename buffer.

Returns

A pointer to the filename buffer.

Return values

NULL	If the path is NULL, if the filename buffer is NULL or if an error occured. The error can be viewed by setting FMC_ENABLE_DEBUG to True .

Warning

The filename buffer must be at least as big as the path.

Note

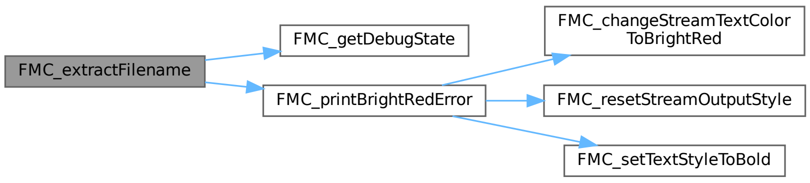
The maximum filename size is MAX_FNAME_SIZE . You can disable some warnings or error messages by defining FMC_ENABLE_DEBUG to False .

Definition at line 32 of file FMC_paths.c.

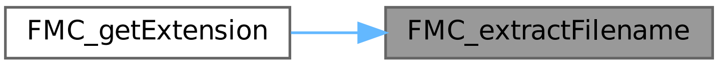
References FMC_getDebugState(), FMC_makeMsg, FMC_printBrightRedError(), MAX_FEXT_SIZE, MAX_FNAME_SIZE, and MAX_FPATH_SIZE.

Referenced by FMC_getExtension().

Here is the call graph for this function:



Here is the caller graph for this function:



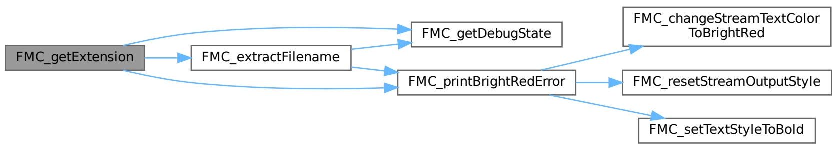
3.26.2.3 FMC_getExtension()

```
char * FMC_getExtension (
            const char *restrict const path,
            char *restrict ext,
            const size_t ext_size )
```

Definition at line 187 of file FMC_paths.c.

References FMC_extractFilename(), FMC_getDebugState(), FMC_makeMsg, FMC_printBrightRedError(), MAX_FEXT_SIZE, and MAX_FNAME_SIZE.

Here is the call graph for this function:



3.27 FMC_file_management.h

Go to the documentation of this file.

```
/*

MIT License

Copyright (c) 2022 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/

#pragma once

#ifndef FMC_FILE_MANAGEMENT_H
#define FMC_FILE_MANAGEMENT_H

#include "../general/FMC_general.h"

FMC_SHARED FMC_FUNC_HOT FMC_FUNC_WARN_UNUSED_RESULT FMC_FUNC_NONNULL(1, 2) char
*FMC_extractFilename(const char * restrict const path, char * restrict filename, const size_t
filename_size);
FMC_SHARED FMC_FUNC_HOT FMC_FUNC_WARN_UNUSED_RESULT FMC_FUNC_NONNULL(1, 2) char *FMC_cutFilename(const
char * restrict const path, char * restrict dirs, const size_t dirs_size);
FMC_SHARED FMC_FUNC_HOT FMC_FUNC_WARN_UNUSED_RESULT FMC_FUNC_NONNULL(1, 2) char
*FMC_getExtension(const char * restrict const path, char * restrict ext, const size_t ext_size);

#endif // FMC_FILE_MANAGEMENT_H
```

3.28 src/files/FMC_fileMan.c File Reference

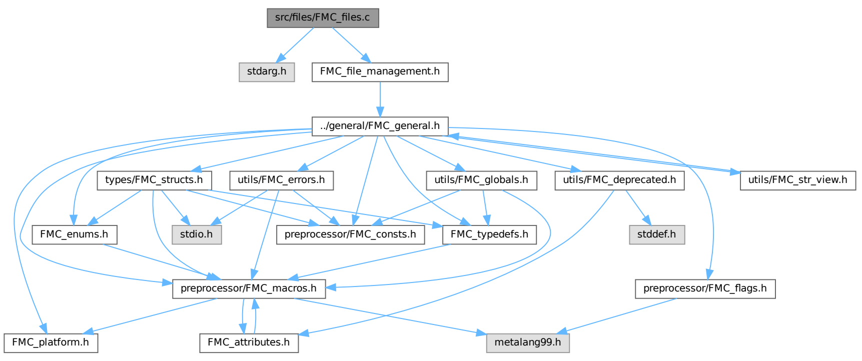
3.29 FMC_fileMan.c

Go to the documentation of this file.

```
/*

MIT License

Copyright (c) 2022-2023 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/

```

3.30 src/files/FMC_files.c File Reference

Include dependency graph for FMC_files.c:



3.31 FMC_files.c

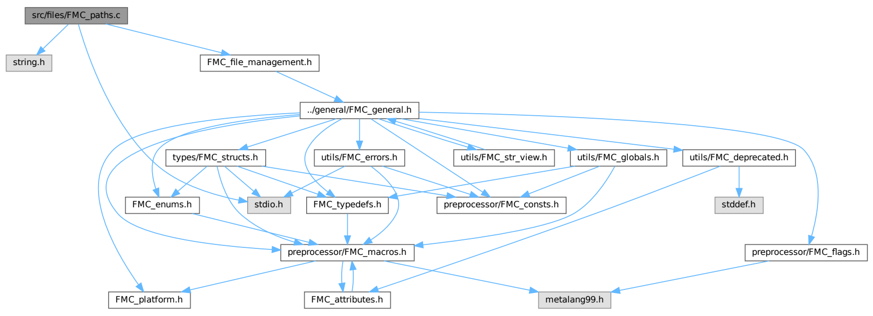
Go to the documentation of this file.

```
/*

MIT License

Copyright (c) 2022-2023 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/

#include <stdarg.h>
#include "FMC_file_management.h"


/* FMC_SHARED FMC_File *FMC_createFile_(unsigned int flags, ...)
{
    va_list args;
    check_in flags for_only_flags(FMC_C_STR_VIEW, TO_OPEN)
    {

    }
    else check_in flags for_only_flags(FMC_C_STR_VIEW, GET_ENCODING)
    {

    }
    else check_in flags for_only_flags(FMC_C_STR_VIEW, GET_SIZE)
    {

    }

    FMC_UNREACHABLE;
}
 */
```

```
/* #pragma GCC diagnostic ignored "-Wnonnull-compare"
    if (!path)
    {
        return NULL;
    }
    #pragma GCC diagnostic pop // -Wnonnull-compare */
```

3.32 src/files/FMC_paths.c File Reference

Include dependency graph for FMC_paths.c:



Functions

- char ∗ FMC_cutFilename (const char ∗restrict const path, char ∗restrict dirs, const size_t dirs_size)
- char ∗ FMC_extractFilename (const char ∗restrict const path, char ∗restrict filename, const size_t filename←_size)

    Gets the filename from a complete path.
- char ∗ FMC_getExtension (const char ∗restrict const path, char ∗restrict ext, const size_t ext_size)

3.32.1 Function Documentation

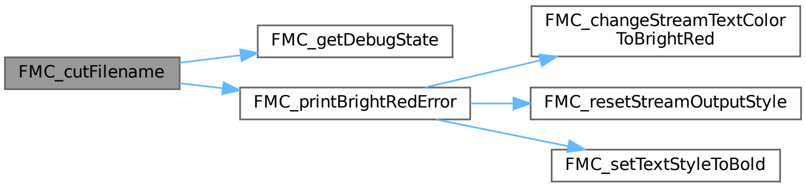
3.32.1.1 FMC_cutFilename()

```
char * FMC_cutFilename (
             const char *restrict const path,
             char *restrict dirs,
             const size_t dirs_size )
```

Definition at line 109 of file FMC_paths.c.

References FMC_getDebugState(), FMC_makeMsg, FMC_printBrightRedError(), MAX_FEXT_SIZE, MAX_FNAME_SIZE, and MAX_FPATH_SIZE.

Here is the call graph for this function:



3.32.1.2 FMC_extractFilename()

```
char * FMC_extractFilename (
             const char *restrict const path,
             char *restrict filename,
             const size_t filename_size )
```

Gets the filename from a complete path.

Author

Axel PASCON

Date

2023

Basically, this function only detects the last '/' or '\' character. For example, if the path is "C:\\Users\\someone\\↩ Documents\\MyFile.txt", the function will return "MyFile.txt". If the path is "/home/someone/Desktop", then Desktop will be considered as the filename. This function is designed to only operate on strings, and do not check if the path is valid, is a file or a directory, etc.

Parameters

in	*path*	The path to extract the filename from.
out	*filename*	The buffer where the filename will be stored.
in	*filename_size*	The size of the filename buffer.

Returns

A pointer to the filename buffer.

Return values

NULL	If the path is NULL, if the filename buffer is NULL or if an error occured. The error can be viewed by setting FMC_ENABLE_DEBUG to True .

Warning

The filename buffer must be at least as big as the path.

Note

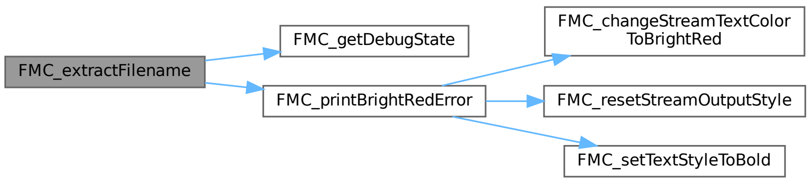
The maximum filename size is MAX_FNAME_SIZE . You can disable some warnings or error messages by defining FMC_ENABLE_DEBUG to False .

Definition at line 32 of file FMC_paths.c.

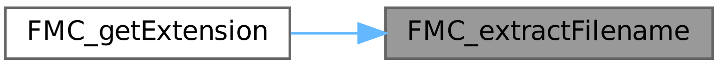
References FMC_getDebugState(), FMC_makeMsg, FMC_printBrightRedError(), MAX_FEXT_SIZE, MAX_FNAME_SIZE, and MAX_FPATH_SIZE.

Referenced by FMC_getExtension().

Here is the call graph for this function:



Here is the caller graph for this function:



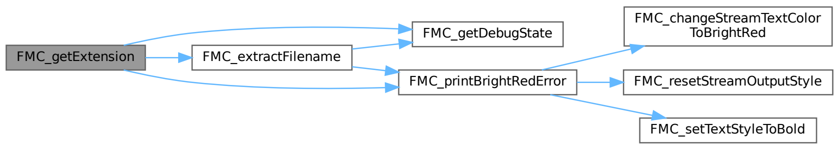
3.32.1.3 FMC_getExtension()

```
char * FMC_getExtension (
            const char *restrict const path,
            char *restrict ext,
            const size_t ext_size )
```

Definition at line 187 of file FMC_paths.c.

References FMC_extractFilename(), FMC_getDebugState(), FMC_makeMsg, FMC_printBrightRedError(), MAX_FEXT_SIZE, and MAX_FNAME_SIZE.

Here is the call graph for this function:



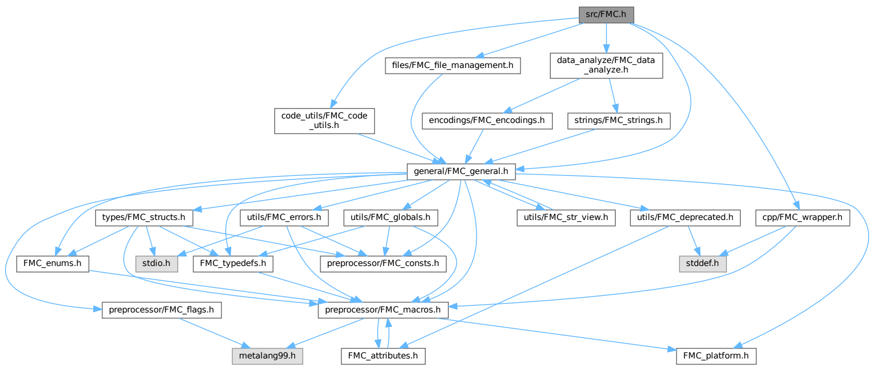
3.33 FMC_paths.c

Go to the documentation of this file.

```
/*

MIT License

Copyright (c) 2022-2023 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/

#include <string.h>
#include <stdio.h>

#include "FMC_file_management.h"

FMC_SHARED FMC_FUNC_HOT FMC_FUNC_WARN_UNUSED_RESULT FMC_FUNC_NONNULL(1, 2) char
*FMC_extractFilename(const char * restrict const path, char * restrict filename, const size_t
filename_size)
{
    #pragma GCC diagnostic ignored "-Wnonnull-compare" // get an error at compile time without this
(because of attribute nonnull)
    if (!path || !filename)
    {
        if (FMC_getDebugState())
        {
            FMC_makeMsg(err_null, 4, "ERROR : ", "In function : ", __func__, ". At least one of the
provided pointers is NULL.");
            FMC_printBrightRedError(stderr, err_null);
        }
        return NULL;
    }
    #pragma GCC diagnostic pop
    memset(filename, 0, filename_size);
    size_t path_len = 0;
    if ((path_len = strnlen(path, MAX_FEXT_SIZE + MAX_FNAME_SIZE + MAX_FPATH_SIZE)) >= MAX_FEXT_SIZE +
MAX_FNAME_SIZE + MAX_FPATH_SIZE)
    {
        FMC_makeMsg(err_path, 4, "ERROR : ", "In function : ", __func__, ". The provided path is too
long (or doesn't contain any nul-character).");
        FMC_printBrightRedError(stderr, err_path);
        return NULL;
    }
    char path_cpy[MAX_FEXT_SIZE + MAX_FNAME_SIZE + MAX_FPATH_SIZE];
    strncpy(path_cpy, path, path_len+1);
    if (strcmp(path_cpy, path) != 0)
    {
        FMC_makeMsg(err_path2, 4, "FMC INTERNAL ERROR : ", "In function : ", __func__, ". strncpy failure.");
        FMC_printBrightRedError(stderr, err_path2);
        return NULL;
    }

    char *last_sep = NULL;
    last_sep = strrchr(path_cpy, (int)'/');
    if (!strrchr(path_cpy, (int)'/') && !strrchr(path_cpy, (int)'\\'))
    {
        filename = strncpy(filename, path_cpy, path_len+1);
        if (strcmp(filename, path_cpy) != 0)
        {
            FMC_makeMsg(err_path3, 4, "FMC INTERNAL ERROR : ", "In function : ", __func__, ". strncpy failure.");
            FMC_printBrightRedError(stderr, err_path3);
            return NULL;
        }
        return filename;
    }
    else if (strrchr(path_cpy, (int)'\\') && strrchr(path_cpy, (int)'/'))
    {
        if (FMC_getDebugState())
        {
            FMC_makeMsg(err_path5, 4, "ERROR : ", "In function : ", __func__, ". The path contains both '/' and '\\'.");
            FMC_printBrightRedError(stderr, err_path5);
        }
        return NULL;
    }
    else if (last_sep)
    {
        filename = strncpy(filename, last_sep+1, path_len+1);
        if (strcmp(filename, last_sep+1) != 0)
        {
            FMC_makeMsg(err_path4, 4, "FMC INTERNAL ERROR : ", "In function : ", __func__, ". strncpy failure.");
            FMC_printBrightRedError(stderr, err_path4);
            return NULL;
        }
        return filename;
    }
    else
    {
        last_sep = strrchr(path_cpy, (int)'\\');
        filename = strncpy(filename, last_sep+1, path_len+1);
        if (strcmp(filename, last_sep+1) != 0)
        {
            FMC_makeMsg(err_path4, 4, "FMC INTERNAL ERROR : ", "In function : ", __func__, ". strncpy failure.");
            FMC_printBrightRedError(stderr, err_path4);
            return NULL;
        }
        return filename;
    }
}

FMC_SHARED FMC_FUNC_HOT FMC_FUNC_WARN_UNUSED_RESULT FMC_FUNC_NONNULL(1, 2) char *FMC_cutFilename(const char * restrict const path, char * restrict dirs, const size_t dirs_size)
{
    #pragma GCC diagnostic ignored "-Wnonnull-compare" // get an error at compile time without this (because of attribute nonnull defined on linux)
    if (!path || !dirs)
    {
        if (FMC_getDebugState())
        {
            FMC_makeMsg(err_null, 4, "ERROR : ", "In function : ", __func__, ". At least one of the provided pointers is NULL.");
            FMC_printBrightRedError(stderr, err_null);
        }
        return NULL;
    }
    #pragma GCC diagnostic pop
    memset(dirs, 0, dirs_size);
    size_t path_len = 0;
    if ((path_len = strnlen(path, MAX_FEXT_SIZE + MAX_FNAME_SIZE + MAX_FPATH_SIZE)) >= MAX_FEXT_SIZE + MAX_FNAME_SIZE + MAX_FPATH_SIZE)
    {
        FMC_makeMsg(err_path, 4, "ERROR : ", "In function : ", __func__, ". The provided path is too long (or doesn't contain any nul-character).");
        FMC_printBrightRedError(stderr, err_path);
        return NULL;
    }
    char path_cpy[MAX_FEXT_SIZE + MAX_FNAME_SIZE + MAX_FPATH_SIZE];
    strncpy(path_cpy, path, path_len+1);
    if (strcmp(path_cpy, path) != 0)
    {
        FMC_makeMsg(err_path2, 4, "FMC INTERNAL ERROR : ", "In function : ", __func__, ". strncpy
failure.");
        FMC_printBrightRedError(stderr, err_path2);
        return NULL;
    }
    char *last_sep = NULL;
    if ((last_sep = strrchr(path_cpy, (int)'/')) && (last_sep = strrchr(path_cpy, (int)'\\')))
    {
        if (FMC_getDebugState())
        {
            FMC_makeMsg(err_path5, 4, "ERROR : ", "In function : ", __func__, ". The path contains
both '/' and '\\'.");
            FMC_printBrightRedError(stderr, err_path5);
        }
        return NULL;
    }

    else if ((last_sep = strrchr(path_cpy, (int)'/')))
    {
        strncpy(dirs, path_cpy, strnlen(path_cpy, path_len) - strnlen(last_sep, path_len) + 1);
        dirs[strnlen(path_cpy, path_len) - strnlen(last_sep, path_len) + 1] = '\0';
        return dirs;
    }

    else if ((last_sep = strrchr(path_cpy, (int)'\\')))
    {
        strncpy(dirs, path_cpy, strnlen(path_cpy, path_len) - strnlen(last_sep, path_len) + 1);
        dirs[strnlen(path_cpy, path_len) - strnlen(last_sep, path_len) + 1] = '\0';
        return dirs;
    }
    else if ((last_sep = strrchr(path_cpy, (int)'~')))
    {
        strncpy(dirs, "~/", 4);
        if (strcmp(dirs, "~/") != 0)
        {
            FMC_makeMsg(err_path4, 4, "FMC INTERNAL ERROR : ", "In function : ", __func__, ". strncpy
failure.");
            FMC_printBrightRedError(stderr, err_path4);
            return NULL;
        }
        return dirs;
    }
    else
    {
        dirs = strncpy(dirs, "./", 4);
        if (strcmp(dirs, "./") != 0)
        {
            FMC_makeMsg(err_path3, 4, "FMC INTERNAL ERROR : ", "In function : ", __func__, ". strncpy
failure.");
            FMC_printBrightRedError(stderr, err_path3);
            return NULL;
        }
        return dirs;
    }

}

FMC_SHARED FMC_FUNC_HOT FMC_FUNC_WARN_UNUSED_RESULT FMC_FUNC_NONNULL(1, 2) char
*FMC_getExtension(const char * restrict const path, char * restrict ext, const size_t ext_size)
{
    #pragma GCC diagnostic ignored "-Wnonnull-compare" // get an error at compile time without this
(because of attribute nonnull)
    if (!path || !ext)
    {
        if (FMC_getDebugState())
        {
            FMC_makeMsg(err_null, 4, "ERROR : ", "In function : ", __func__, ". At least one of the
provided pointers is NULL.");
            FMC_printBrightRedError(stderr, err_null);
        }
        return NULL;
    }
    #pragma GCC diagnostic pop
    memset(ext, 0, ext_size);
    char name[MAX_FNAME_SIZE];
    if (!FMC_extractFilename(path, name, MAX_FNAME_SIZE))
    {
        FMC_makeMsg(err_path6, 4, "FMC INTERNAL ERROR : ", "In function : ", __func__, ".
FMC_extractFilename call failed.");
        FMC_printBrightRedError(stderr, err_path6);
        return NULL;
    }

    if (!strchr(name, (int)'.')) {strncpy(ext, "", 2); return ext;} // Could be modified (?)
    else
    {
        char *last_dot = NULL;
        if ((last_dot = strrchr(name, (int)'.')))
        {
            strncpy(ext, last_dot, strnlen(last_dot+1, MAX_FEXT_SIZE)+1);
            return ext;
        }
        else
        {
            FMC_makeMsg(err_path7, 4, "FMC INTERNAL ERROR : ", "In function : ", __func__, ". strrchr call failed.");
            FMC_printBrightRedError(stderr, err_path7);
            return NULL;
        }
    }
}
```

3.34 src/FMC.h File Reference

Include dependency graph for FMC.h:



Macros

- #define FMC_H

3.34.1 Macro Definition Documentation

3.34.1.1 FMC_H

```
#define FMC_H
```

Definition at line 30 of file FMC.h.

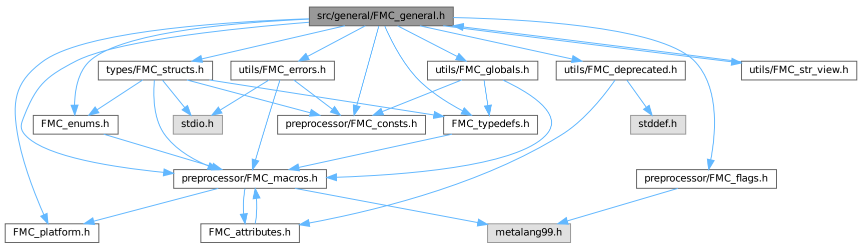
3.35 FMC.h

Go to the documentation of this file.

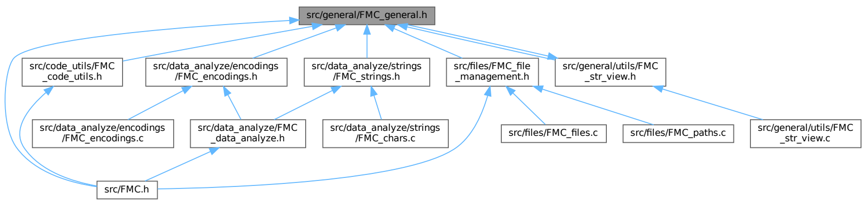
```
/*

MIT License

Copyright (c) 2022 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/

#pragma once

#ifndef FMC_H
#define FMC_H

// includes
#include "general/FMC_general.h"
#include "code_utils/FMC_code_utils.h"
#include "files/FMC_file_management.h"
#include "data_analyze/FMC_data_analyze.h"
#include "cpp/FMC_wrapper.h"




#endif // FMC_H
```

3.36 src/general/FMC_general.h File Reference

Include dependency graph for FMC_general.h:



This graph shows which files directly or indirectly include this file:


src/general/FMC_general.h
src/code_utils/FMC_code_utils.h
src/data_analyze/encodings/FMC_encodings.h
src/data_analyze/strings/FMC_strings.h
src/files/FMC_file_management.h
src/general/utils/FMC_str_view.h
src/data_analyze/encodings/FMC_encodings.c
src/data_analyze/FMC_data_analyze.h
src/data_analyze/strings/FMC_chars.c
src/files/FMC_files.c
src/files/FMC_paths.c
src/general/utils/FMC_str_view.c
src/FMC.h

Macros

- #define FMC_DATA_H

3.36.1 Macro Definition Documentation

3.36.1.1 FMC_DATA_H

```
#define FMC_DATA_H
```

Definition at line 30 of file FMC_general.h.

3.37 FMC_general.h

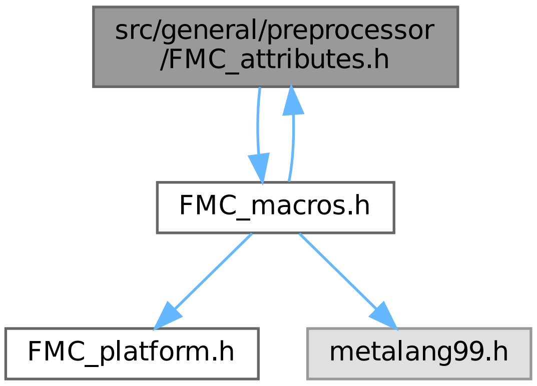
Go to the documentation of this file.

```
00001 /*
00002 
00003 MIT License
00004 
00005 Copyright (c) 2022 Axel PASCON
00006 
00007 Permission is hereby granted, free of charge, to any person obtaining a copy
00008 of this software and associated documentation files (the "Software"), to deal
00009 in the Software without restriction, including without limitation the rights
00010 to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
00011 copies of the Software, and to permit persons to whom the Software is
00012 furnished to do so, subject to the following conditions:
00013 
00014 The above copyright notice and this permission notice shall be included in all
00015 copies or substantial portions of the Software.
00016 
00017 THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
00018 IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
00019 FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
00020 AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
00021 LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
00022 OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
00023 SOFTWARE.
00024 
00025 */
00026 
00027 #pragma once
00028 
00029 #ifndef FMC_DATA_H
00030 #define FMC_DATA_H
00031 
00032 
```

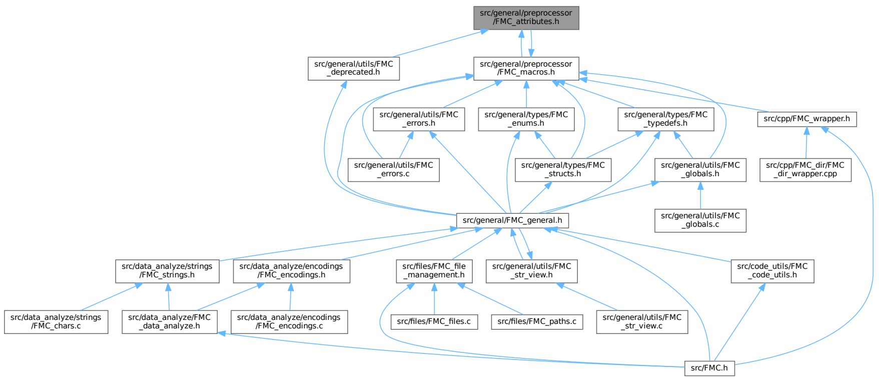
```

#include "preprocessor/FMC_macros.h"
#include "preprocessor/FMC_platform.h"
#include "preprocessor/FMC_consts.h"
#include "types/FMC_structs.h"
#include "preprocessor/FMC_flags.h"
#include "types/FMC_typedefs.h"
#include "types/FMC_enums.h"
#include "utils/FMC_errors.h"
#include "utils/FMC_globals.h"
#include "utils/FMC_deprecated.h"
#include "utils/FMC_str_view.h"

#endif /* FMC_DATA_H */
```

3.38 src/general/preprocessor/FMC_attributes.h File Reference

Include dependency graph for FMC_attributes.h:



This graph shows which files directly or indirectly include this file:



3.39 FMC_attributes.h

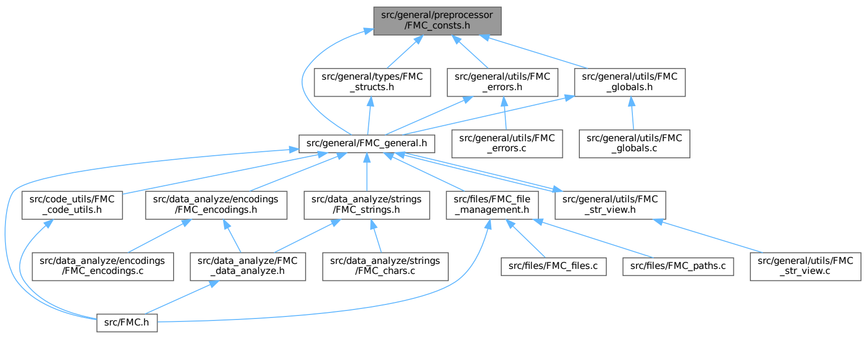
Go to the documentation of this file.

```
/*

MIT License

Copyright (c) 2023 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/

#ifndef FMC_ATTRIBUTES_H
#define FMC_ATTRIBUTES_H

#include "FMC_macros.h"


#ifndef FMC_FUNC_ATTRIBUTES
    #define FMC_FUNC_ATTRIBUTES 1

    #ifndef FMC_FUNC_ACCESS
        #define FMC_FUNC_ACCESS(access_type, ...) __attribute__((access(access_type, __VA_ARGS__)))
    #endif // FMC_FUNC_ACCESS

    #ifndef FMC_FUNC_ALIAS
        #define FMC_FUNC_ALIAS(aliased) __attribute__((alias(FMC_STRINGIZE(aliased))))
    #endif // FMC_FUNC_ALIAS

    #ifndef FMC_FUNC_ALWAYS_INLINE
        #define FMC_FUNC_ALWAYS_INLINE __attribute__((always_inline))
    #endif // FMC_FUNC_ALWAYS_INLINE

    #ifndef FMC_FUNC_COLD
        #define FMC_FUNC_COLD __attribute__((cold))
    #endif // FMC_FUNC_COLD

    #ifndef FMC_FUNC_CONST
        #define FMC_FUNC_CONST __attribute__((const))
    #endif // FMC_FUNC_CONST

    #ifndef FMC_FUNC_CONSTRUCTOR
        #define FMC_FUNC_CONSTRUCTOR(priority) __attribute__((constructor(priority)))
    #endif // FMC_FUNC_CONSTRUCTOR

    #ifndef FMC_FUNC_DESTRUCTOR
        #define FMC_FUNC_DESTRUCTOR(priority) __attribute__((destructor(priority)))
    #endif // FMC_FUNC_DESTRUCTOR

    #ifndef FMC_FUNC_COPY
        #define FMC_FUNC_COPY(func) __attribute__((copy(func)))
    #endif // FMC_FUNC_COPY

    #ifndef FMC_FUNC_DEPRECATED
        #define FMC_FUNC_DEPRECATED(msg) __attribute__((deprecated(FMC_STRINGIZE(msg))))
    #endif // FMC_FUNC_DEPRECATED

    #ifndef FMC_FUNC_UNAVAILABLE
        #define FMC_FUNC_UNAVAILABLE(msg) __attribute__((unavailable(FMC_STRINGIZE(msg))))
    #endif // FMC_FUNC_UNAVAILABLE

    #ifndef FMC_FUNC_ERROR
        #define FMC_FUNC_ERROR(msg) __attribute__((error(FMC_STRINGIZE(msg))))
    #endif // FMC_FUNC_ERROR

    #ifndef FMC_FUNC_WARNING
        #define FMC_FUNC_WARNING(msg) __attribute__((warning(FMC_STRINGIZE(msg))))
    #endif // FMC_FUNC_WARNING

    #ifndef FMC_FUNC_EXTERNALLY_VISIBLE
        #define FMC_FUNC_EXTERNALLY_VISIBLE __attribute__((externally_visible))
    #endif // FMC_FUNC_EXTERNALLY_VISIBLE

    #ifndef FMC_FUNC_FLATTEN
        #define FMC_FUNC_FLATTEN __attribute__((flatten))
    #endif // FMC_FUNC_FLATTEN

    #ifndef FMC_FUNC_FORMAT
        #define FMC_FUNC_FORMAT(func_fmt, fmt_pos, args_pos) __attribute__((format(func_fmt, fmt_pos,
args_pos)))
    #endif // FMC_FUNC_FORMAT

    #ifndef FMC_FUNC_HOT
        #define FMC_FUNC_HOT __attribute__((hot))
    #endif // FMC_FUNC_HOT

    #ifndef FMC_FUNC_MALLOC
        #define FMC_FUNC_MALLOC(...) __attribute__((malloc(__VA_ARGS__)))
    #endif // FMC_FUNC_MALLOC

    #ifndef FMC_FUNC_NONNULL
        #if !(defined(FMC_COMPILING_ON_WINDOWS) || defined(FMC_COMPILING_ON_MINGW))
            #define FMC_FUNC_NONNULL(...) __attribute__((nonnull(__VA_ARGS__)))
        #else
            #define FMC_FUNC_NONNULL(...)
        #endif
    #endif // FMC_FUNC_NONNULL

    #ifndef FMC_FUNC_NORETURN
        #define FMC_FUNC_NORETURN __attribute__((noreturn))
    #endif // FMC_FUNC_NORETURN

    #ifndef FMC_FUNC_OPTIMIZE
        #define FMC_FUNC_OPTIMIZE(level) __attribute__((optimize(FMC_STRINGIZE(level))))
    #endif // FMC_FUNC_OPTIMIZE

    #ifndef FMC_FUNC_PURE
        #define FMC_FUNC_PURE __attribute__((pure))
    #endif // FMC_FUNC_PURE

    #ifndef FMC_FUNC_RETURNS_NONNULL
        #define FMC_FUNC_RETURNS_NONNULL __attribute__((returns_nonnull))
    #endif // FMC_FUNC_RETURNS_NONNULL

    #ifndef FMC_FUNC_SECTION
        #define FMC_FUNC_SECTION(section_name) __attribute__((section(FMC_STRINGIZE(section_name))))
    #endif // FMC_FUNC_SECTION

    #ifndef FMC_FUNC_SENTINEL
        #define FMC_FUNC_SENTINEL(pos) __attribute__((sentinel(pos)))
    #endif // FMC_FUNC_SENTINEL

    #ifndef FMC_FUNC_STACK_PROTECT
        #define FMC_FUNC_STACK_PROTECT __attribute__((stack_protect))
    #endif // FMC_FUNC_STACK_PROTECT

    #ifndef FMC_FUNC_SYMVER
        #define FMC_FUNC_SYMVER(name, major, minor, patch)
__attribute__((symver(FMC_STRINGIZE(name@FMC_CONCAT_4(v,major,minor,patch)))))
    #endif // FMC_FUNC_SYMVER

    #ifndef FMC_FUNC_UNUSED
        #define FMC_FUNC_UNUSED __attribute__((unused))
    #endif // FMC_FUNC_UNUSED

    #ifndef FMC_FUNC_USED
        #define FMC_FUNC_USED __attribute__((used))
    #endif // FMC_FUNC_USED

    #ifndef FMC_FUNC_VISIBILITY
        #define FMC_FUNC_VISIBILITY(visibility_type)
__attribute__((visibility(FMC_STRINGIZE(visibility_type))))
    #endif // FMC_FUNC_VISIBILITY

    #ifndef FMC_FUNC_WARN_UNUSED_RESULT
        #define FMC_FUNC_WARN_UNUSED_RESULT __attribute__((warn_unused_result))
    #endif // FMC_FUNC_WARN_UNUSED_RESULT

    #ifndef FMC_FUNC_WEAK
        #define FMC_FUNC_WEAK __attribute__((weak))
    #endif // FMC_FUNC_WEAK

    #ifndef FMC_FUNC_WEAK_REF
        #define FMC_FUNC_WEAK_REF(...) __attribute__((weakref(FMC_STRINGIZE(__VA_ARGS__))))
    #endif // FMC_FUNC_WEAK_REF
    #ifndef FMC_FUNC_ZERO_REGISTERS
        #define FMC_FUNC_ZERO_REGISTERS(to_zero) __attribute__((zero_call_used_regs(FMC_STRINGIZE(to_zero))))
    #endif // FMC_FUNC_ZERO_REGISTERS



    #ifndef FMC_FUNC_STRONG_ALIAS
        #define FMC_FUNC_STRONG_ALIAS(func_name, aliased) FMC_FUNC_ALIAS(aliased) FMC_FUNC_COPY(aliased) __typeof__(aliased) func_name
    #endif // FMC_FUNC_STRONG_ALIAS

    #ifndef FMC_FUNC_INLINE
        #define FMC_FUNC_INLINE inline FMC_FUNC_ALWAYS_INLINE
    #endif // FMC_FUNC_INLINE

    #ifndef FMC_FUNC_PRINTF_FMT
        #define FMC_FUNC_PRINTF_FMT(fmt_pos, args_pos) FMC_FUNC_FORMAT(printf, fmt_pos, args_pos)
    #endif // FMC_FUNC_PRINTF_FMT

#endif //FMC_FUNC_ATTRIBUTES

#ifndef FMC_VAR_ATTRIBUTES
    #define FMC_VAR_ATTRIBUTES

    #ifndef FMC_VAR_ALIAS
        #define FMC_VAR_ALIAS(aliased) __attribute__((alias(FMC_STRINGIZE(aliased))))
    #endif // FMC_VAR_ALIAS

    #ifndef FMC_VAR_CLEANUP
        #define FMC_VAR_CLEANUP(func_name) __attribute__((cleanup(func_name)))
    #endif // FMC_VAR_CLEANUP

    #ifndef FMC_VAR_COMMON
        #define FMC_VAR_COMMON __attribute__((common))
    #endif // FMC_VAR_COMMON

    #ifndef FMC_VAR_NO_COMMON
        #define FMC_VAR_NO_COMMON __attribute__((nocommon))
    #endif // FMC_VAR_NO_COMMON

    #ifndef FMC_VAR_COPY
        #define FMC_VAR_COPY(var) __attribute__((copy(var)))
    #endif // FMC_VAR_COPY

    #ifndef FMC_VAR_DEPRECATED
        #define FMC_VAR_DEPRECATED(msg) __attribute__((deprecated(FMC_STRINGIZE(msg))))
    #endif // FMC_VAR_DEPRECATED

    #ifndef FMC_VAR_UNAVAILABLE
        #define FMC_VAR_UNAVAILABLE(msg) __attribute__((unavailable(FMC_STRINGIZE(msg))))
    #endif // FMC_VAR_UNAVAILABLE

    #ifndef FMC_VAR_MACH_MODE
        #define FMC_VAR_MACH_MODE(mode) __attribute__((mode(mode)))
    #endif // FMC_VAR_MACH_MODE

    #ifndef FMC_VAR_NON_STRING
        #define FMC_VAR_NON_STRING __attribute__((nonstring))
    #endif // FMC_VAR_NON_STRING

    #ifndef FMC_VAR_SECTION
        #define FMC_VAR_SECTION(section_name) __attribute__((section(FMC_STRINGIZE(section_name))))
    #endif // FMC_VAR_SECTION

    #ifndef FMC_VAR_UNUSED
        #define FMC_VAR_UNUSED __attribute__((unused))
    #endif // FMC_VAR_UNUSED

    #ifndef FMC_VAR_USED
        #define FMC_VAR_USED __attribute__((used))
    #endif // FMC_VAR_USED

    #ifndef FMC_VAR_UNINITIALIZED
        #define FMC_VAR_UNINITIALIZED __attribute__((uninitialized))
    #endif // FMC_VAR_UNINITIALIZED

    #ifndef FMC_VAR_VISIBILITY
        #define FMC_VAR_VISIBILITY(visibility_type) __attribute__((visibility(FMC_STRINGIZE(visibility_type))))
    #endif // FMC_VAR_VISIBILITY

    #ifndef FMC_VAR_WEAK
        #define FMC_VAR_WEAK __attribute__((weak))
    #endif // FMC_VAR_WEAK

```

```
#endif // FMC_VAR_ATTRIBUTES

#ifndef FMC_TYPE_ATTRIBUTES
    #define FMC_TYPE_ATTRIBUTES

    #ifndef FMC_TYPE_DEPRECATED
        #define FMC_TYPE_DEPRECATED(msg) __attribute__((deprecated(FMC_STRINGIZE(msg))))
    #endif // FMC_TYPE_DEPRECATED

    #ifndef FMC_TYPE_UNAVAILABLE
        #define FMC_TYPE_UNAVAILABLE(msg) __attribute__((unavailable(FMC_STRINGIZE(msg))))
    #endif // FMC_TYPE_UNAVAILABLE

    #ifndef FMC_TYPE_MACH_MODE
        #define FMC_TYPE_MACH_MODE(mode) __attribute__((mode(mode)))
    #endif // FMC_TYPE_MACH_MODE

    #ifndef FMC_TYPE_UNUSED
        #define FMC_TYPE_UNUSED __attribute__((unused))
    #endif // FMC_TYPE_UNUSED

    #ifndef FMC_TYPE_VISIBILITY
        #define FMC_TYPE_VISIBILITY(visibility_type)
__attribute__((visibility(FMC_STRINGIZE(visibility_type))))
    #endif // FMC_TYPE_VISIBILITY

#endif // FMC_TYPE_ATTRIBUTES

#ifndef FMC_LABEL_ATTRIBUTES
    #define FMC_LABEL_ATTRIBUTES

    #ifndef FMC_LABEL_UNUSED
        #define FMC_LABEL_UNUSED __attribute__((unused))
    #endif // FMC_LABEL_UNUSED

    #ifndef FMC_LABEL_HOT
        #define FMC_LABEL_HOT __attribute__((hot))
    #endif // FMC_LABEL_HOT

    #ifndef FMC_LABEL_COLD
        #define FMC_LABEL_COLD __attribute__((cold))
    #endif // FMC_LABEL_COLD

#endif // FMC_LABEL_ATTRIBUTES

#ifndef FMC_ENUM_ATTRIBUTES
    #define FMC_ENUM_ATTRIBUTES

    #ifndef FMC_ENUM_DEPRECATED
        #define FMC_ENUM_DEPRECATED(msg) __attribute__((deprecated(FMC_STRINGIZE(msg))))
    #endif // FMC_ENUM_DEPRECATED

    #ifndef FMC_ENUM_UNAVAILABLE
        #define FMC_ENUM_UNAVAILABLE(msg) __attribute__((unavailable(FMC_STRINGIZE(msg))))
    #endif // FMC_ENUM_UNAVAILABLE

#endif // FMC_ENUM_ATTRIBUTES

#ifndef FMC_STMT_ATTRIBUTES
    #define FMC_STMT_ATTRIBUTES

    #ifndef FMC_STMT_FALLTHROUGH
        #define FMC_STMT_FALLTHROUGH __attribute__((fallthrough))
    #endif // FM_STMT_FALLTHROUGH

#endif // FMC_STMT_ATTRIBUTES

#endif // FMC_ATTRIBUTES_H
```

3.40 src/general/preprocessor/FMC_consts.h File Reference

This graph shows which files directly or indirectly include this file:



Macros

- #define BG_BLACK "\x1b[40m"
- #define BG_BLUE "\x1b[44m"
- #define BG_BRIGHT_BLACK "\x1b[100m"
- #define BG_BRIGHT_BLUE "\x1b[104m"
- #define BG_BRIGHT_CYAN "\x1b[106m"
- #define BG_BRIGHT_GREEN "\x1b[102m"
- #define BG_BRIGHT_MAGENTA "\x1b[105m"
- #define BG_BRIGHT_RED "\x1b[101m"
- #define BG_BRIGHT_WHITE "\x1b[107m"
- #define BG_BRIGHT_YELLOW "\x1b[103m"
- #define BG_CYAN "\x1b[46m"
- #define BG_GREEN "\x1b[42m"
- #define BG_MAGENTA "\x1b[45m"
- #define BG_RED "\x1b[41m"
- #define BG_WHITE "\x1b[47m"
- #define BG_YELLOW "\x1b[43m"
- #define False 0
- #define FG_BLACK "\x1b[30m"
- #define FG_BLUE "\x1b[34m"
- #define FG_BRIGHT_BLACK "\x1b[90m"
- #define FG_BRIGHT_BLUE "\x1b[94m"
- #define FG_BRIGHT_CYAN "\x1b[96m"
- #define FG_BRIGHT_GREEN "\x1b[92m"
- #define FG_BRIGHT_MAGENTA "\x1b[95m"
- #define FG_BRIGHT_RED "\x1b[91m"
- #define FG_BRIGHT_WHITE "\x1b[97m"
- #define FG_BRIGHT_YELLOW "\x1b[93m"
- #define FG_CYAN "\x1b[36m"
- #define FG_GREEN "\x1b[32m"
- #define FG_MAGENTA "\x1b[35m"
- #define FG_RED "\x1b[31m"
- #define FG_WHITE "\x1b[37m"
- #define FG_YELLOW "\x1b[33m"

- #define FMC_BOOLEANS
- #define FMC_CONSTS_H
- #define FMC_MAX_PATH_COMPONENTS_SIZE
- #define FMC_STYLES
- #define MAX_FEXT_SIZE 50
- #define MAX_FNAME_SIZE 256
- #define MAX_FPATH_SIZE 512
- #define RESET "\x1b[0m"
- #define True 1
- #define TXT_BLINK "\x1b[5m"
- #define TXT_BOLD "\x1b[1m"
- #define TXT_DIM "\x1b[2m"
- #define TXT_HIDDEN "\x1b[8m"
- #define TXT_REVERSE "\x1b[7m"
- #define TXT_UNDERLINED "\x1b[4m"

3.40.1 Macro Definition Documentation

3.40.1.1 BG_BLACK

```
#define BG_BLACK "\x1b[40m"
```

Definition at line 66 of file FMC_consts.h.

3.40.1.2 BG_BLUE

```
#define BG_BLUE "\x1b[44m"
```

Definition at line 70 of file FMC_consts.h.

3.40.1.3 BG_BRIGHT_BLACK

```
#define BG_BRIGHT_BLACK "\x1b[100m"
```

Definition at line 74 of file FMC_consts.h.

3.40.1.4 BG_BRIGHT_BLUE

```
#define BG_BRIGHT_BLUE "\x1b[104m"
```

Definition at line 78 of file FMC_consts.h.

3.40.1.5 BG_BRIGHT_CYAN

```
#define BG_BRIGHT_CYAN "\x1b[106m"
```

Definition at line 80 of file FMC_consts.h.

3.40.1.6 BG_BRIGHT_GREEN

```
#define BG_BRIGHT_GREEN "\x1b[102m"
```

Definition at line 76 of file FMC_consts.h.

3.40.1.7 BG_BRIGHT_MAGENTA

```
#define BG_BRIGHT_MAGENTA "\x1b[105m"
```

Definition at line 79 of file FMC_consts.h.

3.40.1.8 BG_BRIGHT_RED

```
#define BG_BRIGHT_RED "\x1b[101m"
```

Definition at line 75 of file FMC_consts.h.

3.40.1.9 BG_BRIGHT_WHITE

```
#define BG_BRIGHT_WHITE "\x1b[107m"
```

Definition at line 81 of file FMC_consts.h.

3.40.1.10 BG_BRIGHT_YELLOW

```
#define BG_BRIGHT_YELLOW "\x1b[103m"
```

Definition at line 77 of file FMC_consts.h.

3.40.1.11 BG_CYAN

```
#define BG_CYAN "\x1b[46m"
```

Definition at line 72 of file FMC_consts.h.

3.40.1.12 BG_GREEN

```
#define BG_GREEN "\x1b[42m"
```

Definition at line 68 of file FMC_consts.h.

3.40.1.13 BG_MAGENTA

```
#define BG_MAGENTA "\x1b[45m"
```

Definition at line 71 of file FMC_consts.h.

3.40.1.14 BG_RED

```
#define BG_RED "\x1b[41m"
```

Definition at line 67 of file FMC_consts.h.

3.40.1.15 BG_WHITE

```
#define BG_WHITE "\x1b[47m"
```

Definition at line 73 of file FMC_consts.h.

3.40.1.16 BG_YELLOW

```
#define BG_YELLOW "\x1b[43m"
```

Definition at line 69 of file FMC_consts.h.

3.40.1.17 False

```
#define False 0
```

Definition at line 99 of file FMC_consts.h.

3.40.1.18 FG_BLACK

```
#define FG_BLACK "\x1b[30m"
```

Definition at line 49 of file FMC_consts.h.

3.40.1.19 FG_BLUE

```
#define FG_BLUE "\x1b[34m"
```

Definition at line 53 of file FMC_consts.h.

3.40.1.20 FG_BRIGHT_BLACK

```
#define FG_BRIGHT_BLACK "\x1b[90m"
```

Definition at line 57 of file FMC_consts.h.

3.40.1.21 FG_BRIGHT_BLUE

```
#define FG_BRIGHT_BLUE "\x1b[94m"
```

Definition at line 61 of file FMC_consts.h.

3.40.1.22 FG_BRIGHT_CYAN

```
#define FG_BRIGHT_CYAN "\x1b[96m"
```

Definition at line 63 of file FMC_consts.h.

3.40.1.23 FG_BRIGHT_GREEN

```
#define FG_BRIGHT_GREEN "\x1b[92m"
```

Definition at line 59 of file FMC_consts.h.

3.40.1.24 FG_BRIGHT_MAGENTA

```
#define FG_BRIGHT_MAGENTA "\x1b[95m"
```

Definition at line 62 of file FMC_consts.h.

3.40.1.25 FG_BRIGHT_RED

```
#define FG_BRIGHT_RED "\x1b[91m"
```

Definition at line 58 of file FMC_consts.h.

3.40.1.26 FG_BRIGHT_WHITE

```
#define FG_BRIGHT_WHITE "\x1b[97m"
```

Definition at line 64 of file FMC_consts.h.

3.40.1.27 FG_BRIGHT_YELLOW

```
#define FG_BRIGHT_YELLOW "\x1b[93m"
```

Definition at line 60 of file FMC_consts.h.

3.40.1.28 FG_CYAN

```
#define FG_CYAN "\x1b[36m"
```

Definition at line 55 of file FMC_consts.h.

3.40.1.29 FG_GREEN

```
#define FG_GREEN "\x1b[32m"
```

Definition at line 51 of file FMC_consts.h.

3.40.1.30 FG_MAGENTA

```
#define FG_MAGENTA "\x1b[35m"
```

Definition at line 54 of file FMC_consts.h.

3.40.1.31 FG_RED

```
#define FG_RED "\x1b[31m"
```

Definition at line 50 of file FMC_consts.h.

3.40.1.32 FG_WHITE

```
#define FG_WHITE "\x1b[37m"
```

Definition at line 56 of file FMC_consts.h.

3.40.1.33 FG_YELLOW

```
#define FG_YELLOW "\x1b[33m"
```

Definition at line 52 of file FMC_consts.h.

3.40.1.34 FMC_BOOLEANS

```
#define FMC_BOOLEANS
```

Definition at line 97 of file FMC_consts.h.

3.40.1.35 FMC_CONSTS_H

```
#define FMC_CONSTS_H
```

Definition at line 30 of file FMC_consts.h.

3.40.1.36 FMC_MAX_PATH_COMPONENTS_SIZE

```
#define FMC_MAX_PATH_COMPONENTS_SIZE
```

Definition at line 38 of file FMC_consts.h.

3.40.1.37 FMC_STYLES

```
#define FMC_STYLES
```

Definition at line 45 of file FMC_consts.h.

3.40.1.38 MAX_FEXT_SIZE

```
#define MAX_FEXT_SIZE 50
```

Definition at line 39 of file FMC_consts.h.

3.40.1.39 MAX_FNAME_SIZE

```
#define MAX_FNAME_SIZE 256
```

Definition at line 40 of file FMC_consts.h.

3.40.1.40 MAX_FPATH_SIZE

```
#define MAX_FPATH_SIZE 512
```

Definition at line 41 of file FMC_consts.h.

3.40.1.41 RESET

```
#define RESET "\x1b[0m"
```

Definition at line 47 of file FMC_consts.h.

3.40.1.42 True

```
#define True 1
```

Definition at line 98 of file FMC_consts.h.

3.40.1.43 TXT_BLINK

```
#define TXT_BLINK "\x1b[5m"
```

Definition at line 86 of file FMC_consts.h.

3.40.1.44 TXT_BOLD

```
#define TXT_BOLD "\x1b[1m"
```

Definition at line 83 of file FMC_consts.h.

3.40.1.45 TXT_DIM

```
#define TXT_DIM "\x1b[2m"
```

Definition at line 84 of file FMC_consts.h.

3.40.1.46 TXT_HIDDEN

```
#define TXT_HIDDEN "\x1b[8m"
```

Definition at line 88 of file FMC_consts.h.

3.40.1.47 TXT_REVERSE

```
#define TXT_REVERSE "\x1b[7m"
```

Definition at line 87 of file FMC_consts.h.

3.40.1.48 TXT_UNDERLINED

```
#define TXT_UNDERLINED "\x1b[4m"
```

Definition at line 85 of file FMC_consts.h.

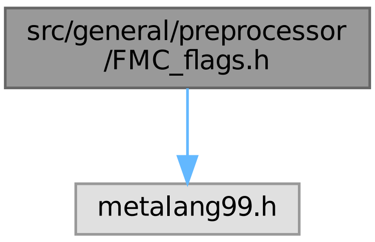
3.41 FMC_consts.h

Go to the documentation of this file.

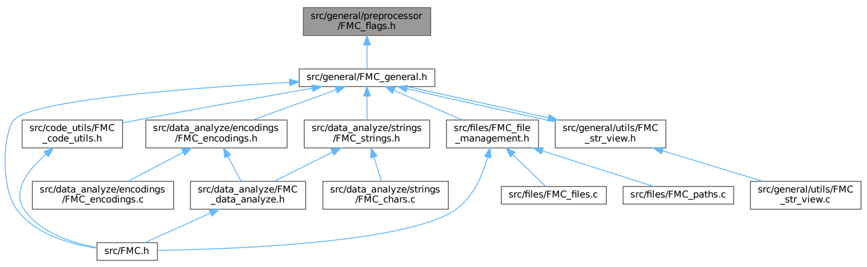
```
/*

MIT License

Copyright (c) 2022 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/

#pragma once

#ifndef FMC_CONSTS_H
#define FMC_CONSTS_H

#if defined(FMC_MAX_PATH_COMPONENTS_SIZE) || defined(MAX_FEXT_SIZE) || defined(MAX_FNAME_SIZE) || defined(MAX_FPATH_SIZE)
    #undef FMC_MAX_PATH_COMPONENTS_SIZE
    #undef MAX_FEXT_SIZE
    #undef MAX_FNAME_SIZE
    #undef MAX_FPATH_SIZE
#endif
#define FMC_MAX_PATH_COMPONENTS_SIZE
#define MAX_FEXT_SIZE 50
#define MAX_FNAME_SIZE 256
#define MAX_FPATH_SIZE 512


#ifndef FMC_STYLES
    #define FMC_STYLES

    #define RESET "\x1b[0m"

    #define FG_BLACK "\x1b[30m"
    #define FG_RED "\x1b[31m"
    #define FG_GREEN "\x1b[32m"
    #define FG_YELLOW "\x1b[33m"
    #define FG_BLUE "\x1b[34m"
    #define FG_MAGENTA "\x1b[35m"
    #define FG_CYAN "\x1b[36m"
    #define FG_WHITE "\x1b[37m"
    #define FG_BRIGHT_BLACK "\x1b[90m"
    #define FG_BRIGHT_RED "\x1b[91m"
    #define FG_BRIGHT_GREEN "\x1b[92m"
    #define FG_BRIGHT_YELLOW "\x1b[93m"
    #define FG_BRIGHT_BLUE "\x1b[94m"
    #define FG_BRIGHT_MAGENTA "\x1b[95m"
    #define FG_BRIGHT_CYAN "\x1b[96m"
    #define FG_BRIGHT_WHITE "\x1b[97m"

    #define BG_BLACK "\x1b[40m"
    #define BG_RED "\x1b[41m"
    #define BG_GREEN "\x1b[42m"
    #define BG_YELLOW "\x1b[43m"
    #define BG_BLUE "\x1b[44m"
    #define BG_MAGENTA "\x1b[45m"
    #define BG_CYAN "\x1b[46m"
    #define BG_WHITE "\x1b[47m"
    #define BG_BRIGHT_BLACK "\x1b[100m"
    #define BG_BRIGHT_RED "\x1b[101m"
    #define BG_BRIGHT_GREEN "\x1b[102m"
    #define BG_BRIGHT_YELLOW "\x1b[103m"
    #define BG_BRIGHT_BLUE "\x1b[104m"
    #define BG_BRIGHT_MAGENTA "\x1b[105m"
    #define BG_BRIGHT_CYAN "\x1b[106m"
    #define BG_BRIGHT_WHITE "\x1b[107m"

    #define TXT_BOLD "\x1b[1m"
    #define TXT_DIM "\x1b[2m"
    #define TXT_UNDERLINED "\x1b[4m"
    #define TXT_BLINK "\x1b[5m"
    #define TXT_REVERSE "\x1b[7m"
    #define TXT_HIDDEN "\x1b[8m"

#endif // FMC_STYLES

#if defined(FMC_BOOLEANS) || defined(True) || defined(False)
    #undef FMC_BOOLEANS
    #undef True
    #undef False
#endif // FMC_BOOLEANS
#define FMC_BOOLEANS
#define True 1
#define False 0

#endif // FMC_CONSTS_H
```

3.42 src/general/preprocessor/FMC_flags.h File Reference

Include dependency graph for FMC_flags.h:



This graph shows which files directly or indirectly include this file:


src/general/preprocessor/FMC_flags.h
src/general/FMC_general.h
src/code_utils/FMC_code_utils.h
src/data_analyze/encodings/FMC_encodings.h
src/data_analyze/strings/FMC_strings.h
src/files/FMC_file_management.h
src/general/utils/FMC_str_view.h
src/data_analyze/encodings/FMC_encodings.c
src/data_analyze/FMC_data_analyze.h
src/data_analyze/strings/FMC_chars.c
src/files/FMC_files.c
src/files/FMC_paths.c
src/general/utils/FMC_str_view.c
src/FMC.h

Macros

- #define ASCII 64U
- #define C_STR 2U
- #define C_STR_PTR 8U
- #define check_in if(((
- #define FMC_C_STR_VIEW 1U
- #define FMC_C_STR_VIEW_PTR 4U
- #define FMC_ENCODING_FLAGS
- #define FMC_FLAGS_H
- #define for_at_least_flags(...)) & (ML99_LIST_EVAL(ML99_call(ML99_listIntersperse, v(|), ML99_list(v(__VA_ARGS__)))))) == (ML99_LIST_EVAL(ML99_call(ML99_listIntersperse, v(|), ML99_list(v(__VA_ARGS__))))))
- #define for_only_flags(...)) | (ML99_LIST_EVAL(ML99_call(ML99_listIntersperse, v(|), ML99_list(v(__VA_ARGS__)))))) == (ML99_LIST_EVAL(ML99_call(ML99_listIntersperse, v(|), ML99_list(v(__VA_ARGS__))))))
- #define GET_ENCODING 32U
- #define TO_OPEN 16U
- #define UNKNOWN 128U
- #define UTF16_BE 8U
- #define UTF16_LE 4U
- #define UTF32_BE 32U
- #define UTF32_LE 16U
- #define UTF8 1U
- #define UTF8_BOM 2U

3.42.1 Macro Definition Documentation

3.42.1.1 ASCII

```
#define ASCII 64U
```

Definition at line 61 of file FMC_flags.h.

3.42.1.2 C_STR

```
#define C_STR 2U
```

Definition at line 73 of file FMC_flags.h.

3.42.1.3 C_STR_PTR

```
#define C_STR_PTR 8U
```

Definition at line 75 of file FMC_flags.h.

3.42.1.4 check_in

```
#define check_in if(((
```

Definition at line 38 of file FMC_flags.h.

3.42.1.5 FMC_C_STR_VIEW

```
#define FMC_C_STR_VIEW 1U
```

Definition at line 72 of file FMC_flags.h.

3.42.1.6 FMC_C_STR_VIEW_PTR

```
#define FMC_C_STR_VIEW_PTR 4U
```

Definition at line 74 of file FMC_flags.h.

3.42.1.7 FMC_ENCODING_FLAGS

```
#define FMC_ENCODING_FLAGS
```

Definition at line 54 of file FMC_flags.h.

3.42.1.8 FMC_FLAGS_H

```
#define FMC_FLAGS_H
```

Definition at line 30 of file FMC_flags.h.

3.42.1.9 for_at_least_flags

```
#define for_at_least_flags(
              ... ) ) & (ML99_LIST_EVAL(ML99_call(ML99_listIntersperse, v(|), ML99_list(v(←
__VA_ARGS__)))))) == (ML99_LIST_EVAL(ML99_call(ML99_listIntersperse, v(|), ML99_list(v(__VA_←
ARGS__))))))
```

Definition at line 40 of file FMC_flags.h.

3.42.1.10 for_only_flags

```
#define for_only_flags(
              ... ) ) | (ML99_LIST_EVAL(ML99_call(ML99_listIntersperse, v(|), ML99_list(v(←
__VA_ARGS__)))))) == (ML99_LIST_EVAL(ML99_call(ML99_listIntersperse, v(|), ML99_list(v(__VA_←
ARGS__))))))
```

Definition at line 39 of file FMC_flags.h.

3.42.1.11 GET_ENCODING

```
#define GET_ENCODING 32U
```

Definition at line 77 of file FMC_flags.h.

3.42.1.12 TO_OPEN

```
#define TO_OPEN 16U
```

Definition at line 76 of file FMC_flags.h.

3.42.1.13 UNKNOWN

```
#define UNKNOWN 128U
```

Definition at line 62 of file FMC_flags.h.

3.42.1.14 UTF16_BE

```
#define UTF16_BE 8U
```

Definition at line 58 of file FMC_flags.h.

3.42.1.15 UTF16_LE

```
#define UTF16_LE 4U
```

Definition at line 57 of file FMC_flags.h.

3.42.1.16 UTF32_BE

```
#define UTF32_BE 32U
```

Definition at line 60 of file FMC_flags.h.

3.42.1.17 UTF32_LE

```
#define UTF32_LE 16U
```

Definition at line 59 of file FMC_flags.h.

3.42.1.18 UTF8

```
#define UTF8 1U
```

Definition at line 55 of file FMC_flags.h.

3.42.1.19 UTF8_BOM

```
#define UTF8_BOM 2U
```

Definition at line 56 of file FMC_flags.h.

3.43 FMC_flags.h

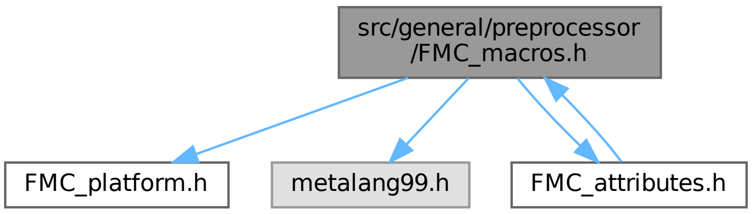
Go to the documentation of this file.

```
00001 /*
00002 
00003 MIT License
00004 
00005 Copyright (c) 2022 Axel PASCON
00006 
00007 Permission is hereby granted, free of charge, to any person obtaining a copy
00008 of this software and associated documentation files (the "Software"), to deal
00009 in the Software without restriction, including without limitation the rights
00010 to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
00011 copies of the Software, and to permit persons to whom the Software is
00012 furnished to do so, subject to the following conditions:
00013 
00014 The above copyright notice and this permission notice shall be included in all
00015 copies or substantial portions of the Software.
00016 
00017 THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
00018 IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
00019 FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
00020 AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
00021 LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
00022 OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
00023 SOFTWARE.
00024 
00025 */
00026 
00027 #pragma once
00028 
00029 #ifndef FMC_FLAGS_H
00030 #define FMC_FLAGS_H
00031 
00032 #include <metalang99.h>
00033 
00034 #if defined(check_in) || defined(for_only_flags) || defined(for_at_least_flags)
00035     #undef check_in
00036     #undef for_only_flags
00037 #endif // check_in || for_only_flags
00038 #define check_in if(((
00039 #define for_only_flags(...) ) | (ML99_LIST_EVAL(ML99_call(ML99_listIntersperse, v(|),
      ML99_list(v(__VA_ARGS__)))))) == (ML99_LIST_EVAL(ML99_call(ML99_listIntersperse, v(|),
      ML99_list(v(__VA_ARGS__))))))
00040 #define for_at_least_flags(...) ) & (ML99_LIST_EVAL(ML99_call(ML99_listIntersperse, v(|),
      ML99_list(v(__VA_ARGS__)))))) == (ML99_LIST_EVAL(ML99_call(ML99_listIntersperse, v(|),
      ML99_list(v(__VA_ARGS__))))))
00041 
00042 
00043 #if defined(FMC_ENCODING_FLAGS) || defined(UTF8) || defined(UTF8_BOM) || defined(UTF16_LE) ||
      defined(UTF16_BE) || defined(UTF32_LE) || defined(UTF32_BE) || defined(ASCII) || defined(UNKNOWN)
00044     #undef FMC_ENCODING_FLAGS
00045     #undef UTF8
00046     #undef UTF8_BOM
00047     #undef UTF16_LE
00048     #undef UTF16_BE
00049     #undef UTF32_LE
00050     #undef UTF32_BE
00051     #undef ASCII
00052     #undef UNKNOWN
00053 #endif
00054 #define FMC_ENCODING_FLAGS
00055 #define UTF8 1U
00056 #define UTF8_BOM 2U
00057 #define UTF16_LE 4U
00058 #define UTF16_BE 8U
00059 #define UTF32_LE 16U
00060 #define UTF32_BE 32U
00061 #define ASCII 64U
00062 #define UNKNOWN 128U
00063 
00064 #if defined(FMC_C_STR_VIEW) || defined(C_STR) || defined(FMC_C_STR_VIEW_PTR) || defined(C_STR_PTR) ||
      defined(TO_OPEN) || defined(GET_ENCODING)
```

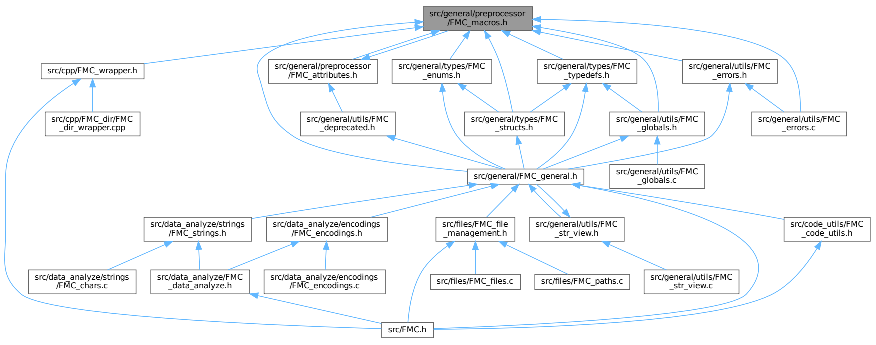
```
    #undef FMC_C_STR_VIEW_PTR
    #undef C_STR_PTR
    #undef TO_OPEN
    #undef GET_ENCODING
    #undef C_STR
    #undef FMC_C_STR_VIEW
#endif // FMC_CSTR || C_STR || FMC_C_STR_VIEW_PTR || C_STR_PTR || TO_OPEN || GET_ENCODING
#define FMC_C_STR_VIEW 1U
#define C_STR 2U
#define FMC_C_STR_VIEW_PTR 4U
#define C_STR_PTR 8U
#define TO_OPEN 16U
#define GET_ENCODING 32U

#endif // FMC_FLAGS_H
```

3.44 src/general/preprocessor/FMC_macros.h File Reference

Include dependency graph for FMC_macros.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define defer(stmt, body) do body while (0); stmt
- #define FMC_BEGIN_DECLS
- #define FMC_COMPILE_TIME_ERROR(msg) _Pragma(STRINGIZE(GCC error STRINGIZE(msg)))
- #define FMC_DECR_BY(x, y) ML99_EVAL(ML99_call(ML99_sub, v(x), v(y)))
- #define FMC_END_DECLS

- #define FMC_ERROR_CHECK(cond, todo_stmt, enable_debug, todo_before)
- #define FMC_ID(x) FMC_ID2(x)
- #define FMC_ID2(x) FMC_ID3(x)
- #define FMC_ID3(x) FMC_ID4(x)
- #define FMC_ID4(x) FMC_ID5(x)
- #define FMC_ID5(x) FMC_ID6(x)
- #define FMC_ID6(x) FMC_ID7(x)
- #define FMC_ID7(x) FMC_ID8(x)
- #define FMC_ID8(x) FMC_ID9(x)
- #define FMC_ID9(x) x
- #define FMC_MACROS_H
- #define FMC_MAJOR_VERSION 1
- #define FMC_MINOR_VERSION 0
- #define FMC_PATCH_VERSION 0
- #define FMC_VERSION FMC_CONCAT_5(FMC_MAJOR_VERSION, FMC_PP_POINT(), FMC_MINOR_VERSION, FMC_PP_POINT(), FMC_PATCH_VERSION)
- #define FMC_VERSION_NUMBER FMC_CONCAT_2(FMC_MAJOR_VERSION∗10000 + FMC_MINOR_VERSION∗100 + FMC_PATCH_VERSION, L)
- #define FMC_VERSION_STRING FMC_STRINGIZE_5(FMC_MAJOR_VERSION, FMC_PP_POINT(), FMC_MINOR_VERSION, FMC_PP_POINT(), FMC_PATCH_VERSION)
- #define foreach(elem, array, start, stop_index_cond) size_t foreach_counter(0) = start; for(typeof(array[foreach_counter(0)]) elem = array[foreach_counter(0)]; foreach_stop_cond(stop_index_cond) ; foreach_counter(0)++, elem = array[foreach_counter(0)])
- #define foreach_counter(lines_after_foreach) FMC_CONCAT_2(base_index, FMC_DECR_BY(__LINE__←↩, lines_after_foreach))
- #define foreach_stop_cond(x) ML99_EVAL(ML99_EVAL(ML99_call(ML99_if, ML99_isNothing(x), v(ML99←↩_id(ML99_id(v(foreach_counter(0) < sizeof(array)/sizeof(array[0]))))), v(ML99_maybeUnwrap(x)))))
- #define LOOP_TO_THE_END ML99_nothing()
- #define LOOP_WHILE(x) ML99_just(v(x))

3.44.1 Macro Definition Documentation

3.44.1.1 defer

```
#define defer(
              stmt,
              body ) do body while (0); stmt
```

Definition at line 125 of file FMC_macros.h.

3.44.1.2 FMC_BEGIN_DECLS

```
#define FMC_BEGIN_DECLS
```

Definition at line 196 of file FMC_macros.h.

3.44.1.3 FMC_COMPILE_TIME_ERROR

```
#define FMC_COMPILE_TIME_ERROR(
              msg ) _Pragma(STRINGIZE(GCC error STRINGIZE(msg)))
```

Definition at line 225 of file FMC_macros.h.

3.44.1.4 FMC_DECR_BY

```
#define FMC_DECR_BY(
              x,
              y ) ML99_EVAL(ML99_call(ML99_sub, v(x), v(y)))
```

Definition at line 120 of file FMC_macros.h.

3.44.1.5 FMC_END_DECLS

```
#define FMC_END_DECLS
```

Definition at line 197 of file FMC_macros.h.

3.44.1.6 FMC_ERROR_CHECK

```
#define FMC_ERROR_CHECK(
              cond,
              todo_stmt,
              enable_debug,
              todo_before )
```

Value:

```
    if (cond)                                                    \
    {   if(enable_debug) todo_before                             \
        todo_stmt;                                               \
    }
```

Definition at line 232 of file FMC_macros.h.

3.44.1.7 FMC_ID

```
#define FMC_ID(
              x ) FMC_ID2(x)
```

Definition at line 115 of file FMC_macros.h.

3.44.1.8 FMC_ID2

```
#define FMC_ID2(
              x ) FMC_ID3(x)
```

Definition at line 114 of file FMC_macros.h.

3.44.1.9 FMC_ID3

```
#define FMC_ID3(
              x ) FMC_ID4(x)
```

Definition at line 113 of file FMC_macros.h.

3.44.1.10 FMC_ID4

```
#define FMC_ID4(
              x ) FMC_ID5(x)
```

Definition at line 112 of file FMC_macros.h.

3.44.1.11 FMC_ID5

```
#define FMC_ID5(
              x ) FMC_ID6(x)
```

Definition at line 111 of file FMC_macros.h.

3.44.1.12 FMC_ID6

```
#define FMC_ID6(
              x ) FMC_ID7(x)
```

Definition at line 110 of file FMC_macros.h.

3.44.1.13 FMC_ID7

```
#define FMC_ID7(
              x ) FMC_ID8(x)
```

Definition at line 109 of file FMC_macros.h.

3.44.1.14 FMC_ID8

```
#define FMC_ID8(
              x ) FMC_ID9(x)
```

Definition at line 108 of file FMC_macros.h.

3.44.1.15 FMC_ID9

```
#define FMC_ID9(
              x ) x
```

Definition at line 107 of file FMC_macros.h.

3.44.1.16 FMC_MACROS_H

```
#define FMC_MACROS_H
```

Definition at line 31 of file FMC_macros.h.

3.44.1.17 FMC_MAJOR_VERSION

```
#define FMC_MAJOR_VERSION 1
```

Definition at line 165 of file FMC_macros.h.

3.44.1.18 FMC_MINOR_VERSION

```
#define FMC_MINOR_VERSION 0
```

Definition at line 166 of file FMC_macros.h.

3.44.1.19 FMC_PATCH_VERSION

```
#define FMC_PATCH_VERSION 0
```

Definition at line 167 of file FMC_macros.h.

3.44.1.20 FMC_VERSION

```
#define FMC_VERSION FMC_CONCAT_5(FMC_MAJOR_VERSION, FMC_PP_POINT(), FMC_MINOR_VERSION, FMC_↩
PP_POINT(), FMC_PATCH_VERSION)
```

Definition at line 168 of file FMC_macros.h.

3.44.1.21 FMC_VERSION_NUMBER

```
#define FMC_VERSION_NUMBER FMC_CONCAT_2(FMC_MAJOR_VERSION*10000 + FMC_MINOR_VERSION*100 +
FMC_PATCH_VERSION, L)
```

Definition at line 170 of file FMC_macros.h.

3.44.1.22 FMC_VERSION_STRING

```
#define FMC_VERSION_STRING FMC_STRINGIZE_5(FMC_MAJOR_VERSION, FMC_PP_POINT(), FMC_MINOR_VERSION,
FMC_PP_POINT(), FMC_PATCH_VERSION)
```

Definition at line 169 of file FMC_macros.h.

3.44.1.23 foreach

```
#define foreach(
              elem,
              array,
              start,
              stop_index_cond ) size_t foreach_counter(0) = start; for(typeof(array[foreach_counter(0)])
elem = array[foreach_counter(0)]; foreach_stop_cond(stop_index_cond) ; foreach_counter(0)++,
elem = array[foreach_counter(0)])
```

Definition at line 138 of file FMC_macros.h.

3.44.1.24 foreach_counter

```
#define foreach_counter(
              lines_after_foreach ) FMC_CONCAT_2(base_index, FMC_DECR_BY(__LINE__, lines_↩
after_foreach))
```

Definition at line 136 of file FMC_macros.h.

3.44.1.25 foreach_stop_cond

```
#define foreach_stop_cond(
              x ) ML99_EVAL(ML99_EVAL(ML99_call(ML99_if, ML99_isNothing(x), v(ML99_id(ML99_↩
id(v(foreach_counter(0) < sizeof(array)/sizeof(array[0]))))), v(ML99_maybeUnwrap(x)))))
```

Definition at line 137 of file FMC_macros.h.

3.44.1.26 LOOP_TO_THE_END

```
#define LOOP_TO_THE_END ML99_nothing()
```

Definition at line 134 of file FMC_macros.h.

3.44.1.27 LOOP_WHILE

```
#define LOOP_WHILE(
              x ) ML99_just(v(x))
```

Definition at line 135 of file FMC_macros.h.

3.45 FMC_macros.h

Go to the documentation of this file.

```
/*

MIT License

Copyright (c) 2022-2023 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/


#pragma once

#ifndef FMC_MACROS_H
#define FMC_MACROS_H

#include "FMC_platform.h"
#include <metalang99.h>
#include "FMC_attributes.h"

```

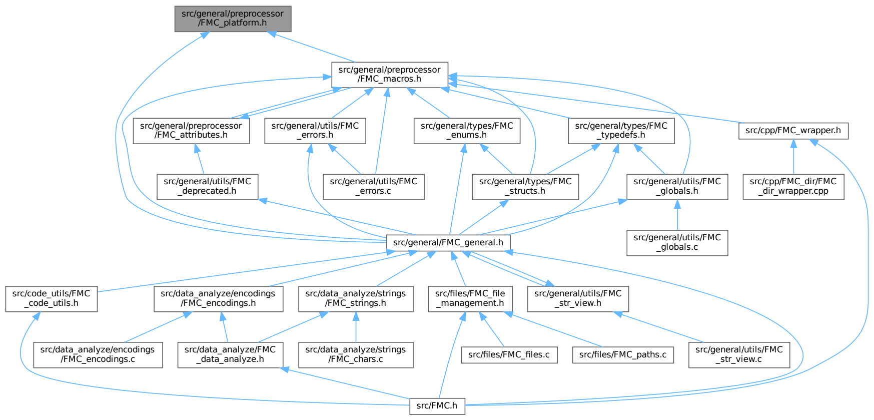
```

/* Used to avoid false warnings (for example "attribute destructor/constructor does not take argument", when it actually can) */
#if defined(__INTELLISENSE__ )
    #pragma diag_suppress 1094
#endif

#ifndef FMC_PP_POINT
    #define FMC_PP_POINT() .
#endif

#ifndef FMC_CONCAT_MACROS
    #define FMC_CONCAT_MACROS
    #define FMC_CONCAT10(x, y) x##y
    #define FMC_CONCAT9(x, y) FMC_CONCAT10(x, y)
    #define FMC_CONCAT8(x, y) FMC_CONCAT9(x, y)
    #define FMC_CONCAT7(x, y) FMC_CONCAT8(x, y)
    #define FMC_CONCAT6(x, y) FMC_CONCAT7(x, y)
    #define FMC_CONCAT5(x, y) FMC_CONCAT6(x, y)
    #define FMC_CONCAT4(x, y) FMC_CONCAT5(x, y)
    #define FMC_CONCAT3(x, y) FMC_CONCAT4(x, y)
    #define FMC_CONCAT2(x, y) FMC_CONCAT3(x, y)
    #define FMC_CONCAT(x, y) FMC_CONCAT2(x, y)

    #define FMC_CONCAT_2(x, y) FMC_CONCAT(x, y)
    #define FMC_CONCAT_3(x, y, z) FMC_CONCAT(FMC_CONCAT(x, y), z)
    #define FMC_CONCAT_4(x, y, z, w) FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(x, y), z), w)
    #define FMC_CONCAT_5(x, y, z, w, v) FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(x, y), z), w), v)
    #define FMC_CONCAT_6(x, y, z, w, v, u) FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(x, y), z), w), v), u)
    #define FMC_CONCAT_7(x, y, z, w, v, u, t) FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(x, y), z), w), v), u), t)
    #define FMC_CONCAT_8(x, y, z, w, v, u, t, s) FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(x, y), z), w), v), u), t), s)
    #define FMC_CONCAT_9(x, y, z, w, v, u, t, s, r) FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(x, y), z), w), v), u), t), s), r)
#endif

#ifndef FMC_STRINGIZE_MACROS
    #define FMC_STRINGIZE_MACROS
    #define FMC_STRINGIZE10(x) #x
    #define FMC_STRINGIZE9(x) FMC_STRINGIZE10(x)
    #define FMC_STRINGIZE8(x) FMC_STRINGIZE9(x)
    #define FMC_STRINGIZE7(x) FMC_STRINGIZE8(x)
    #define FMC_STRINGIZE6(x) FMC_STRINGIZE7(x)
    #define FMC_STRINGIZE5(x) FMC_STRINGIZE6(x)
    #define FMC_STRINGIZE4(x) FMC_STRINGIZE5(x)
    #define FMC_STRINGIZE3(x) FMC_STRINGIZE4(x)
    #define FMC_STRINGIZE2(x) FMC_STRINGIZE3(x)
    #define FMC_STRINGIZE(x) FMC_STRINGIZE2(x)
#endif

#ifndef FMC_STRINGIZE_X
    #define FMC_STRINGIZE_X
    #define FMC_STRINGIZE_2(x, y) FMC_STRINGIZE(FMC_CONCAT(x, y))
    #define FMC_STRINGIZE_3(x, y, z) FMC_STRINGIZE(FMC_CONCAT(FMC_CONCAT(x, y), z))
    #define FMC_STRINGIZE_4(x, y, z, w) FMC_STRINGIZE(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(x, y), z), w))
    #define FMC_STRINGIZE_5(x, y, z, w, v) FMC_STRINGIZE(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(x, y), z), w), v))
    #define FMC_STRINGIZE_6(x, y, z, w, v, u) FMC_STRINGIZE(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(x, y), z), w), v), u))
    #define FMC_STRINGIZE_7(x, y, z, w, v, u, t) FMC_STRINGIZE(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(x, y), z), w), v), u), t))
    #define FMC_STRINGIZE_8(x, y, z, w, v, u, t, s) FMC_STRINGIZE(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(x, y), z), w), v), u), t), s))
    #define FMC_STRINGIZE_9(x, y, z, w, v, u, t, s, r) FMC_STRINGIZE(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(FMC_CONCAT(x, y), z), w), v), u), t), s), r))
#endif

#if defined(FMC_ID) || defined(FMC_ID2) || defined(FMC_ID3) || defined(FMC_ID4) || defined(FMC_ID5) || defined(FMC_ID6) || defined(FMC_ID7) || defined(FMC_ID8) || defined(FMC_ID9)
    #undef FMC_ID
    #undef FMC_ID2
    #undef FMC_ID3
    #undef FMC_ID4
    #undef FMC_ID5
    #undef FMC_ID6
    #undef FMC_ID7
    #undef FMC_ID8
    #undef FMC_ID9
#endif
#define FMC_ID9(x) x
```

```
#define FMC_ID8(x) FMC_ID9(x)
#define FMC_ID7(x) FMC_ID8(x)
#define FMC_ID6(x) FMC_ID7(x)
#define FMC_ID5(x) FMC_ID6(x)
#define FMC_ID4(x) FMC_ID5(x)
#define FMC_ID3(x) FMC_ID4(x)
#define FMC_ID2(x) FMC_ID3(x)
#define FMC_ID(x) FMC_ID2(x)

#if defined(FMC_DECR_BY)
    #undef FMC_DECR_BY
#endif
#define FMC_DECR_BY(x, y) ML99_EVAL(ML99_call(ML99_sub, v(x), v(y)))

#ifdef defer
    #undef defer
#endif
#define defer(stmt, body) do body while (0); stmt

#if defined(foreach) || defined(foreach_counter) || defined(foreach_stop_cond) || defined(LOOP_TO_THE_END) || defined(LOOP_WHILE)
    #undef foreach
    #undef foreach_counter
    #undef foreach_stop_cond
    #undef LOOP_TO_THE_END
    #undef LOOP_WHILE
#endif
#define LOOP_TO_THE_END ML99_nothing()
#define LOOP_WHILE(x) ML99_just(v(x))
#define foreach_counter(lines_after_foreach) FMC_CONCAT_2(base_index, FMC_DECR_BY(__LINE__, lines_after_foreach))
#define foreach_stop_cond(x) ML99_EVAL(ML99_EVAL(ML99_call(ML99_if, ML99_isNothing(x), v(ML99_id(ML99_id(v(foreach_counter(0) < sizeof(array)/sizeof(array[0])))))), v(ML99_maybeUnwrap(x)))))
#define foreach(elem, array, start, stop_index_cond) size_t foreach_counter(0) = start; for(typeof(array[foreach_counter(0)]) elem = array[foreach_counter(0)]; foreach_stop_cond(stop_index_cond) ; foreach_counter(0)++, elem = array[foreach_counter(0)])


#ifndef FMC_METHODS
    #define FMC_METHODS

    #define DECL_METHOD(name, ret, ...) \
        ret (*name)(__VA_ARGS__)

    #define INIT_STRUCT_METHOD(method, associated_function) \
        .method = associated_function

#endif // FMC_METHODS

/*#ifndef FMC_OVERLOAD
    #define FMC_OVERLOAD(func)
*/

#ifdef FMC_VERSION
    #undef FMC_VERSION
    #undef FMC_VERSION_STRING
    #undef FMC_VERSION_NUMBER
    #undef FMC_MAJOR_VERSION
    #undef FMC_MINOR_VERSION
    #undef FMC_PATCH_VERSION
#endif // FMC_VERSION

#define FMC_MAJOR_VERSION 1
#define FMC_MINOR_VERSION 0
#define FMC_PATCH_VERSION 0
#define FMC_VERSION FMC_CONCAT_5(FMC_MAJOR_VERSION, FMC_PP_POINT(), FMC_MINOR_VERSION, FMC_PP_POINT(), FMC_PATCH_VERSION)
#define FMC_VERSION_STRING FMC_STRINGIZE_5(FMC_MAJOR_VERSION, FMC_PP_POINT(), FMC_MINOR_VERSION, FMC_PP_POINT(), FMC_PATCH_VERSION)
#define FMC_VERSION_NUMBER FMC_CONCAT_2(FMC_MAJOR_VERSION*10000 + FMC_MINOR_VERSION*100 + FMC_PATCH_VERSION, L)

#ifndef FMC_alloca
    #define FMC_alloca(size) __builtin_alloca(size)
#endif

#ifndef FMC_PROB
    #define FMC_PROB(true_expr, prob) __builtin_expect_with_probability(true_expr, 1, prob)
#endif

#ifndef FMC_UNREACHABLE
    #define FMC_UNREACHABLE __builtin_unreachable()
#endif

#ifndef FMC_MAKE_VOID
    #define FMC_MAKE_VOID(expr) do { (void)(expr); } while (0)
#endif
```

```

#if defined(FMC_BEGIN_DECLS) || defined(FMC_END_DECLS)
    #undef FMC_BEGIN_DECLS
    #undef FMC_END_DECLS
#endif
#ifdef __cplusplus
    #define FMC_BEGIN_DECLS extern "C" {
    #define FMC_END_DECLS }
#else
    #define FMC_BEGIN_DECLS
    #define FMC_END_DECLS
#endif

/* Maybe I'll have to modify this, even though it sounds fine to me now. */
#ifndef FMC_SHARED
    #if FMC_COMPILING_ON_WINDOWS && !defined(FMC_STATIC)
        #if defined(FMC_BUILD_DLL)
            #define FMC_SHARED __declspec(dllexport)
        #elif defined(USE_FMC_DLL)
            #define FMC_SHARED __declspec(dllimport)
        #else
            #error "You must define FMC_BUILD_DLL to build the DLL or USE_FMC_DLL to use the built DLL. To use or build the static library, please define FMC_STATIC."
        #endif
    #elif FMC_COMPILING_ON_WINDOWS && defined(FMC_STATIC)
        #define FMC_SHARED
    #elif FMC_COMPILING_ON_LINUX || FMC_COMPILING_ON_MACOS
        #if defined(FMC_STATIC) || defined(USE_FMC_DLL) || defined(FMC_BUILD_DLL)
            #warning "You don't have to specify FMC_STATIC, USE_FMC_DLL or FMC_BUILD_DLL on Linux, Unix or Mac OS X. These are ignored on your system."
        #endif
        #define FMC_SHARED
    #else
        #error "Unsupported OS"
    #endif // PLATFORMS
#endif // FMC_SHARED

#ifdef FMC_COMPILE_TIME_ERROR
    #undef FMC_COMPILE_TIME_ERROR
#endif // FMC_COMPILE_TIME_ERROR
#define FMC_COMPILE_TIME_ERROR(msg) _Pragma(STRINGIZE(GCC error STRINGIZE(msg)))


#ifdef FMC_ERROR_CHECK
    #undef FMC_ERROR_CHECK
#endif // FMC_ERROR_CHECK
// thought about this for lisibility, not sure if I'll use it though
#define FMC_ERROR_CHECK(cond, todo_stmt, enable_debug, todo_before) \
    if (cond)                                                       \
    {   if(enable_debug) todo_before                                \
        todo_stmt;                                                  \
    }

#endif // FMC_MACROS_H
```

3.46 src/general/preprocessor/FMC_platform.h File Reference

This graph shows which files directly or indirectly include this file:


src/general/preprocessor/FMC_platform.h
src/general/preprocessor/FMC_macros.h
src/general/preprocessor/FMC_attributes.h
src/general/utils/FMC_errors.h
src/general/types/FMC_enums.h
src/general/types/FMC_typedefs.h
src/cpp/FMC_wrapper.h
src/general/utils/FMC_deprecated.h
src/general/utils/FMC_errors.c
src/general/types/FMC_structs.h
src/general/utils/FMC_globals.h
src/cpp/FMC_dir/FMC_dir_wrapper.cpp
src/general/utils/FMC_globals.c
src/general/FMC_general.h
src/code_utils/FMC_code_utils.h
src/data_analyze/encodings/FMC_encodings.h
src/data_analyze/strings/FMC_strings.h
src/files/FMC_file_management.h
src/general/utils/FMC_str_view.h
src/data_analyze/encodings/FMC_encodings.c
src/data_analyze/FMC_data_analyze.h
src/data_analyze/strings/FMC_chars.c
src/files/FMC_files.c
src/files/FMC_paths.c
src/general/utils/FMC_str_view.c
src/FMC.h

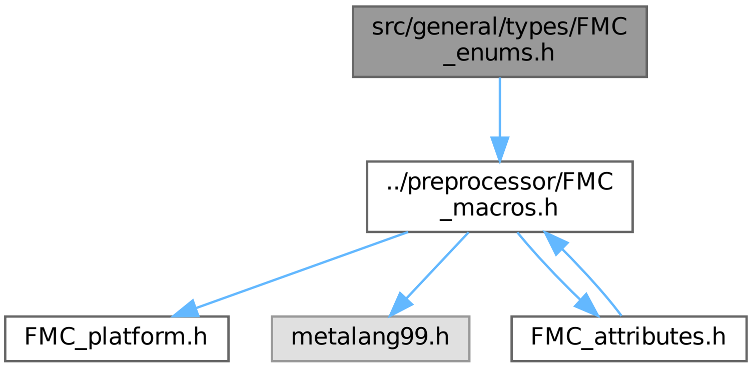
3.47 FMC_platform.h

Go to the documentation of this file.

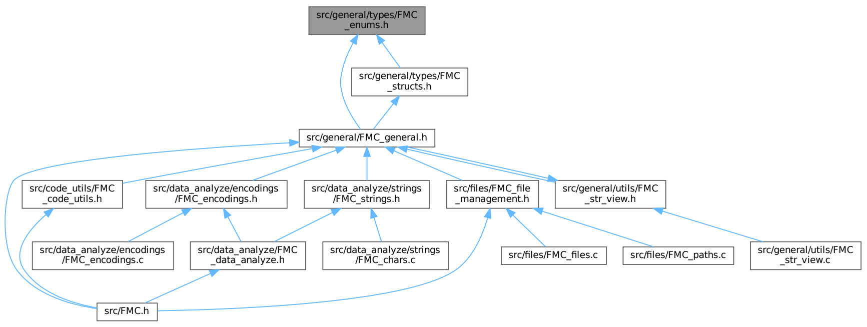
```
/*

MIT License

Copyright (c) 2022-2023 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/

#ifndef FMC_PLATFORM_H
#define FMC_PLATFORM_H


#if defined(FMC_COMPILING_ON_WINDOWS)
    #undef FMC_COMPILING_ON_WINDOWS
#elif defined(FMC_COMPILING_ON_LINUX)
    #undef FMC_COMPILING_ON_LINUX
#endif // OS detection
#if defined(_WIN32) || defined(_WIN64) || defined(__WIN32__) || defined(__TOS_WIN__) ||
defined(__WINDOWS__)
    #define FMC_COMPILING_ON_WINDOWS 1
#elif defined(__linux__) || defined(__linux) || defined(linux) || defined(__gnu_linux__)
    #define FMC_COMPILING_ON_LINUX 1
#else
    #warning "This library hasn't been tested on this OS."
#endif // OS management

#if defined(FMC_COMPILING_ON_MINGW)
    #undef FMC_COMPILING_ON_MINGW
#elif defined(FMC_COMPILING_WITH_GCC)
    #undef FM_COMPILING_WITH_GCC
#endif // Compiler and environment detection
#if defined(__MINGW32__) || defined(__MINGW64__) || defined(__MINGW32) || defined(__MINGW64) ||
defined(__MINGW__)
    #define FMC_COMPILING_ON_MINGW 1
#elif defined(__GNUC__) || defined(__GNUG__)
    #define FMC_COMPILING_WITH_GCC 1
#else
    #warning "This library hasn't been tested on your compiler."
#endif // Compiler and environment management

// check C17 standard
#ifndef __cplusplus
    #if __STDC_VERSION__ < 201710L
        #error "FManC requires C17 standard or higher."
    #endif
#else
    #if __cplusplus < 201703L
        #error "FManC requires C++17 standard or higher."
    #endif
#endif

#endif /* FMC_PLATFORM_H */
```

3.48 src/general/types/FMC_enums.h File Reference

Include dependency graph for FMC_enums.h:



This graph shows which files directly or indirectly include this file:


src/general/types/FMC_enums.h
src/general/types/FMC_structs.h
src/general/FMC_general.h
src/code_utils/FMC_code_utils.h
src/data_analyze/encodings/FMC_encodings.h
src/data_analyze/strings/FMC_strings.h
src/files/FMC_file_management.h
src/general/utils/FMC_str_view.h
src/data_analyze/encodings/FMC_encodings.c
src/data_analyze/FMC_data_analyze.h
src/data_analyze/strings/FMC_chars.c
src/files/FMC_files.c
src/files/FMC_paths.c
src/general/utils/FMC_str_view.c
src/FMC.h

Macros

- #define FMC_ENUMS_H

Typedefs

- typedef enum FManC_Encodings FMC_Encodings

Enumerations

- enum FManC_Encodings {
  utf8 = 1 , utf8_bom = 2 , utf16_le = 4 , utf16_be = 8 ,
  utf32_le = 16 , utf32_be = 32 , ascii = 64 , unknown = 128 ,
  error = 256 }

3.48.1 Macro Definition Documentation

3.48.1.1 FMC_ENUMS_H

```
#define FMC_ENUMS_H
```

Definition at line 30 of file FMC_enums.h.

3.48.2 Typedef Documentation

3.48.2.1 FMC_Encodings

```
typedef enum FManC_Encodings FMC_Encodings
```

Definition at line 47 of file FMC_enums.h.

3.48.3 Enumeration Type Documentation

3.48.3.1 FManC_Encodings

```
enum FManC_Encodings
```

Enumerator

utf8	
utf8_bom	
utf16_le	
utf16_be	
utf32_le	
utf32_be	
ascii	
unknown	
error	

Definition at line 34 of file FMC_enums.h.

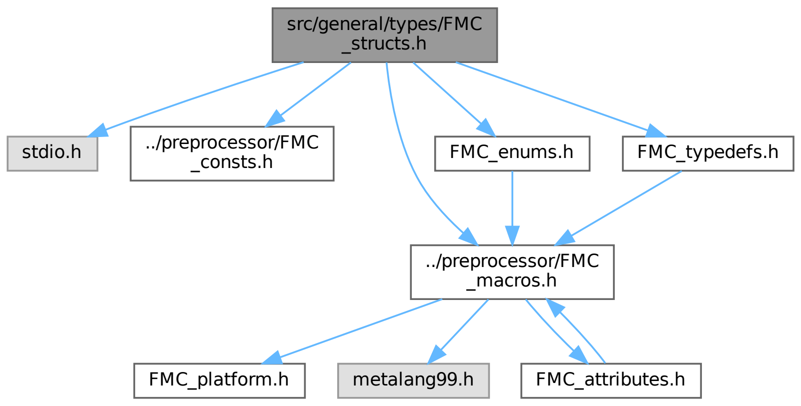
3.49 FMC_enums.h

Go to the documentation of this file.

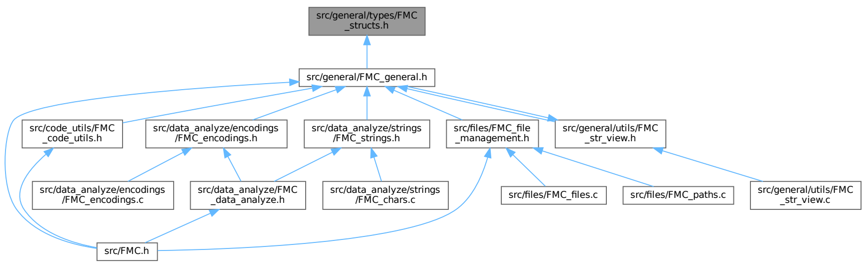
```
/*

MIT License

Copyright (c) 2022-2023 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/

#pragma once

#ifndef FMC_ENUMS_H
#define FMC_ENUMS_H

#include "../preprocessor/FMC_macros.h"

FMC_SHARED enum FManC_Encodings
{
    utf8 = 1,
    utf8_bom = 2,
    utf16_le = 4,
    utf16_be = 8,
    utf32_le = 16,
    utf32_be = 32,
    ascii = 64,
    unknown = 128,
    error = 256
};

typedef enum FManC_Encodings FMC_Encodings;

#endif // FMC_ENUMS_H
```

3.50 src/general/types/FMC_structs.h File Reference

Include dependency graph for FMC_structs.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct FManC_Char
- struct FManC_CharComp
- struct FManC_CStrView
- struct FManC_File
- struct FManC_String
- struct FManC_StrOcc

Macros

- #define FMC_STRUCTS_H

Typedefs

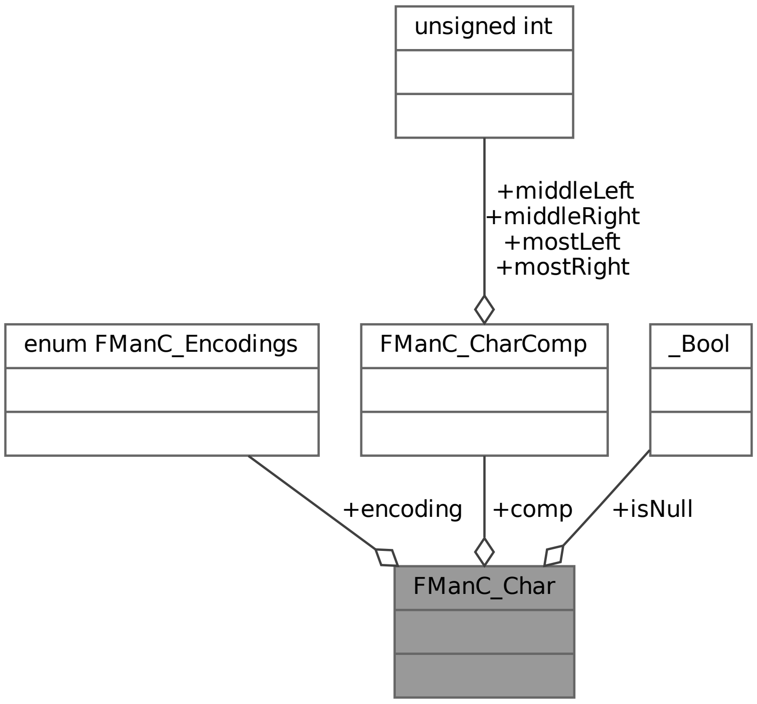
- typedef struct FManC_Char FMC_Char
- typedef struct FManC_CharComp FMC_CharComp
- typedef struct FManC_CStrView FMC_CStrView
- typedef struct FManC_File FMC_File
- typedef struct FManC_String FMC_String
- typedef struct FManC_StrOcc FMC_StrOcc

3.50.1 Data Structure Documentation

3.50.1.1 struct FManC_Char

Definition at line 69 of file FMC_structs.h.

Collaboration diagram for FManC_Char:



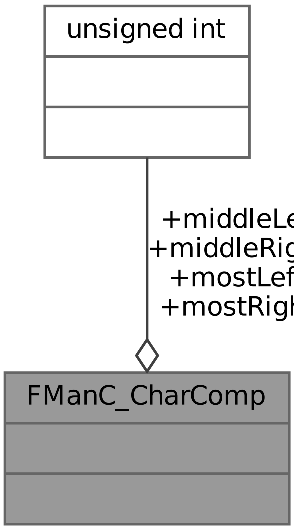
Data Fields

FMC_CharComp	comp	
FMC_Encodings	encoding	
FMC_CharControl	isNull	

3.50.1.2 struct FManC_CharComp

Definition at line 59 of file FMC_structs.h.

Collaboration diagram for FManC_CharComp:



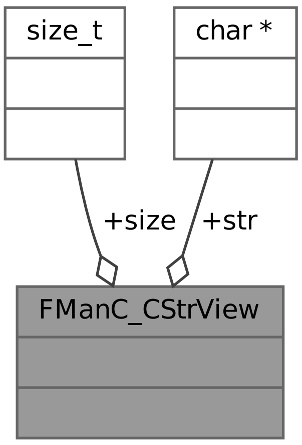
Data Fields

unsigned int	middleLeft: 8	
unsigned int	middleRight: 8	
unsigned int	mostLeft: 8	
unsigned int	mostRight: 8	

3.50.1.3 struct FManC_CStrView

Definition at line 87 of file FMC_structs.h.

Collaboration diagram for FManC_CStrView:



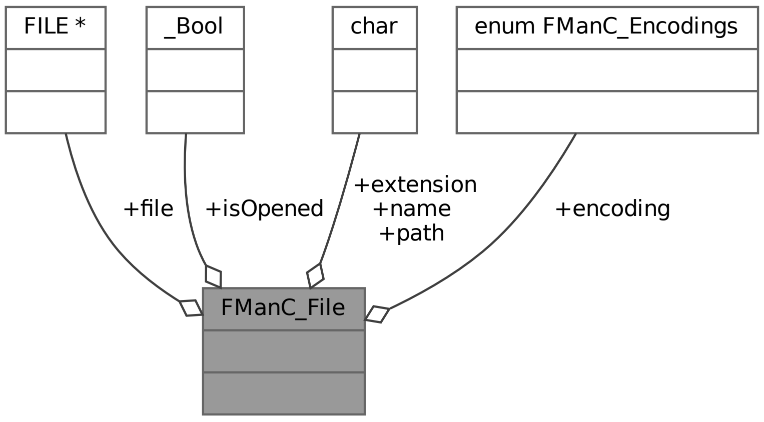
Data Fields

size_t	size	
char ∗	str	

3.50.1.4 struct FManC_File

Definition at line 39 of file FMC_structs.h.

Collaboration diagram for FManC_File:



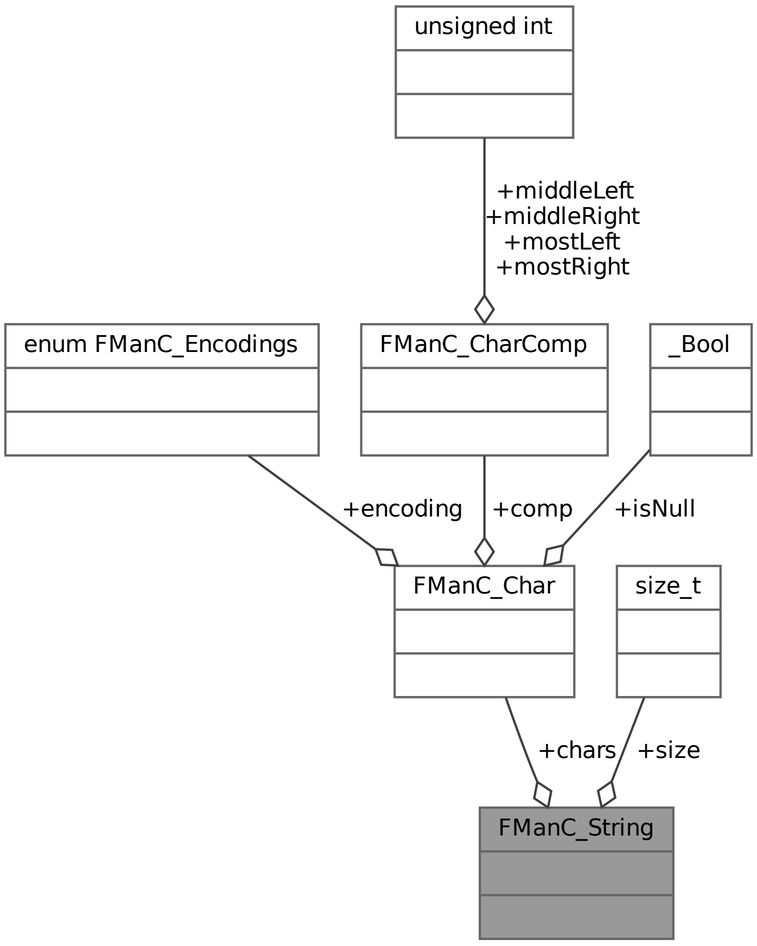
Data Fields

FMC_Encodings	encoding	
char	extension[MAX_FEXT_SIZE]	
FILE ∗	file	
FMC_FileState	isOpened	
char	name[MAX_FNAME_SIZE]	
char	path[MAX_FPATH_SIZE]	

3.50.1.5 struct FManC_String

Definition at line 79 of file FMC_structs.h.

Collaboration diagram for FManC_String:



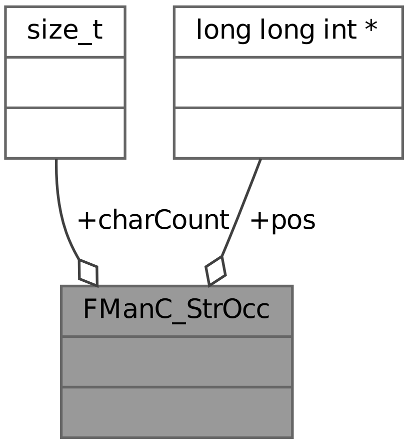
Data Fields

FMC_Char ∗	chars	
size_t	size	

3.50.1.6 struct FManC_StrOcc

Definition at line 51 of file FMC_structs.h.

Collaboration diagram for FManC_StrOcc:



Data Fields

size_t	charCount	
long long int ∗	pos	

3.50.2 Macro Definition Documentation

3.50.2.1 FMC_STRUCTS_H

```
#define FMC_STRUCTS_H
```

Definition at line 30 of file FMC_structs.h.

3.50.3 Typedef Documentation

3.50.3.1 FMC_Char

```
typedef struct FManC_Char FMC_Char
```

Definition at line 76 of file FMC_structs.h.

3.50.3.2 FMC_CharComp

```
typedef struct FManC_CharComp FMC_CharComp
```

Definition at line 67 of file FMC_structs.h.

3.50.3.3 FMC_CStrView

```
typedef struct FManC_CStrView FMC_CStrView
```

Definition at line 93 of file FMC_structs.h.

3.50.3.4 FMC_File

```
typedef struct FManC_File FMC_File
```

Definition at line 49 of file FMC_structs.h.

3.50.3.5 FMC_String

```
typedef struct FManC_String FMC_String
```

Definition at line 85 of file FMC_structs.h.

3.50.3.6 FMC_StrOcc

```
typedef struct FManC_StrOcc FMC_StrOcc
```

Definition at line 57 of file FMC_structs.h.

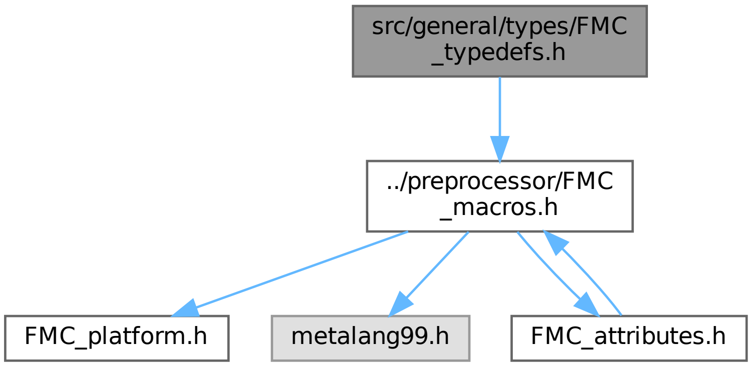
3.51 FMC_structs.h

Go to the documentation of this file.

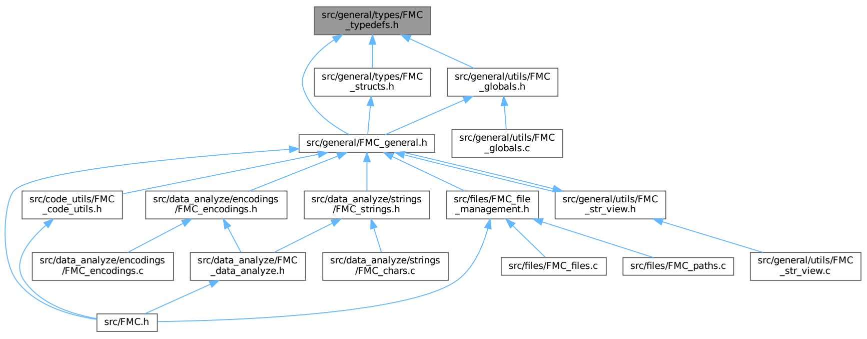
```c
/*

MIT License

Copyright (c) 2022-2023 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/

#pragma once

#ifndef FMC_STRUCTS_H
#define FMC_STRUCTS_H


#include <stdio.h>
#include "../preprocessor/FMC_consts.h"
#include "../preprocessor/FMC_macros.h"
#include "FMC_enums.h"
#include "FMC_typedefs.h"

FMC_SHARED struct FManC_File
{
    FILE *file;
    FMC_FileState isOpened;
    char path[MAX_FPATH_SIZE];
    char name[MAX_FNAME_SIZE];
    char extension[MAX_FEXT_SIZE];
    FMC_Encodings encoding;
};

typedef struct FManC_File FMC_File;

FMC_SHARED struct FManC_StrOcc
{
    size_t charCount;
    long long int *pos;
};

typedef struct FManC_StrOcc FMC_StrOcc;

FMC_SHARED struct FManC_CharComp
{
    unsigned int mostLeft : 8;
    unsigned int middleLeft : 8;
    unsigned int middleRight : 8;
    unsigned int mostRight : 8;
};

typedef struct FManC_CharComp FMC_CharComp;

FMC_SHARED struct FManC_Char
{
    FMC_Encodings encoding;
    FMC_CharComp comp;
    FMC_CharControl isNull;
};

typedef struct FManC_Char FMC_Char;


FMC_SHARED struct FManC_String
{
    FMC_Char *chars;
    size_t size;
};

typedef struct FManC_String FMC_String;

FMC_SHARED struct FManC_CStrView
{
    size_t size;
    char *str;
};

typedef struct FManC_CStrView FMC_CStrView;

/*#include <threads.h>


FMC_SHARED struct FManC_ArenaElement
{
    void* current;
    size_t alignement;
};

FMC_SHARED struct FManC_Arena
{
    void* start;
    void* end;

};*/

#endif // FMC_STRUCTS_H
```

3.52 src/general/types/FMC_typedefs.h File Reference

Include dependency graph for FMC_typedefs.h:


src/general/types/FMC_typedefs.h
../preprocessor/FMC_macros.h
FMC_platform.h
metalang99.h
FMC_attributes.h

This graph shows which files directly or indirectly include this file:


src/general/types/FMC_typedefs.h
src/general/types/FMC_structs.h
src/general/utils/FMC_globals.h
src/general/FMC_general.h
src/general/utils/FMC_globals.c
src/code_utils/FMC_code_utils.h
src/data_analyze/encodings/FMC_encodings.h
src/data_analyze/strings/FMC_strings.h
src/files/FMC_file_management.h
src/general/utils/FMC_str_view.h
src/data_analyze/encodings/FMC_encodings.c
src/data_analyze/FMC_data_analyze.h
src/data_analyze/strings/FMC_chars.c
src/files/FMC_files.c
src/files/FMC_paths.c
src/general/utils/FMC_str_view.c
src/FMC.h

Macros

- #define FMC_TYPEDEFS_H

Typedefs

- typedef _Bool FMC_Bool
- typedef _Bool FMC_CharControl
- typedef _Bool FMC_FileState
- typedef int found_bs_n
- typedef int found_bs_r_bs_n
- typedef int found_bs_t

3.52.1 Macro Definition Documentation

3.52.1.1 FMC_TYPEDEFS_H

```
#define FMC_TYPEDEFS_H
```

Definition at line 30 of file FMC_typedefs.h.

3.52.2 Typedef Documentation

3.52.2.1 FMC_Bool

```
typedef _Bool FMC_Bool
```

Definition at line 39 of file FMC_typedefs.h.

3.52.2.2 FMC_CharControl

```
typedef _Bool FMC_CharControl
```

Definition at line 37 of file FMC_typedefs.h.

3.52.2.3 FMC_FileState

```
typedef _Bool FMC_FileState
```

Definition at line 38 of file FMC_typedefs.h.

3.52.2.4 found_bs_n

```
typedef int found_bs_n
```

Definition at line 34 of file FMC_typedefs.h.

3.52.2.5 found_bs_r_bs_n

```
typedef int found_bs_r_bs_n
```

Definition at line 36 of file FMC_typedefs.h.

3.52.2.6 found_bs_t

```
typedef int found_bs_t
```

Definition at line 35 of file FMC_typedefs.h.

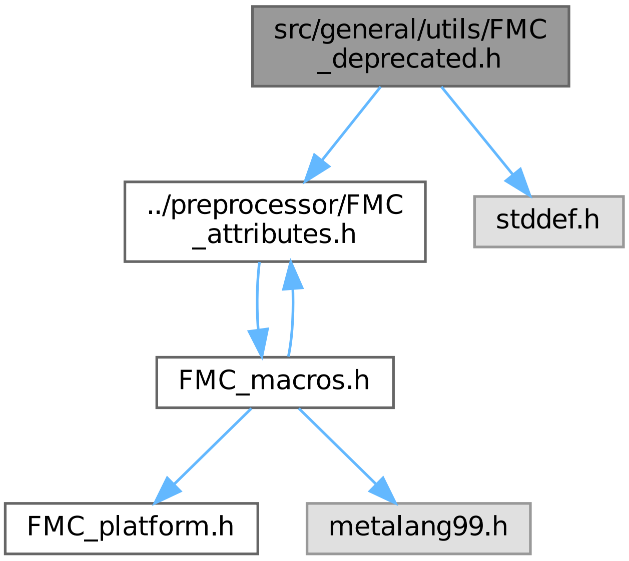
3.53 FMC_typedefs.h

Go to the documentation of this file.

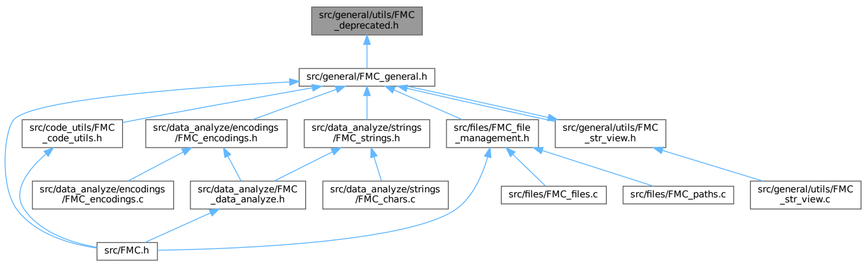
```
/*

MIT License

Copyright (c) 2022-2023 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/

#pragma once

#ifndef FMC_TYPEDEFS_H
#define FMC_TYPEDEFS_H

#include "../preprocessor/FMC_macros.h"

typedef int found_bs_n;
typedef int found_bs_t;
typedef int found_bs_r_bs_n;
typedef _Bool FMC_CharControl;
typedef _Bool FMC_FileState;
typedef _Bool FMC_Bool;


#endif // FMC_TYPEDEFS_H
```

3.54 src/general/utils/FMC_deprecated.h File Reference

Include dependency graph for FMC_deprecated.h:



This graph shows which files directly or indirectly include this file:



Functions

- FMC_FUNC_UNAVAILABLE (This function is not anymore available in the library since the version 1.0.0. Use FMC_cutFilename instead) void fgetFilePath(char ∗sourceFilePath
- FMC_FUNC_UNAVAILABLE (This function is not anymore available in the library since the version 1.0.0. Use FMC_extractFilename instead) void fgetFileName(char ∗sourceFilePath
- FMC_FUNC_UNAVAILABLE (This function is not anymore available in the library since the version 1.0.0. Use FMC_getExtension instead) void fgetFileExtension(char ∗sourceFilePath
- FMC_FUNC_UNAVAILABLE (This function is not anymore available in the library since the version 1.0.0.) char ∗copyFileWithoutTabAndLineBreak(char ∗sourceFilePath
- FMC_TYPE_UNAVAILABLE (This type is not anymore available in the library since the version 1.0.0.) struct FMANC_SO

Variables

- char ∗ extension
- char ∗ fileName
- char ∗ filePath
- char ∗∗ pathToCopy
- char ∗ toSearch

3.54.1 Function Documentation

3.54.1.1 FMC_FUNC_UNAVAILABLE() [1/4]

```
FMC_FUNC_UNAVAILABLE (
             This function is not anymore available in the library since the version 1.0.0.
Use FMC_cutFilename instead )
```

3.54.1.2 FMC_FUNC_UNAVAILABLE() [2/4]

```
FMC_FUNC_UNAVAILABLE (
             This function is not anymore available in the library since the version 1.0.0.
Use FMC_extractFilename instead )
```

3.54.1.3 FMC_FUNC_UNAVAILABLE() [3/4]

```
FMC_FUNC_UNAVAILABLE (
             This function is not anymore available in the library since the version 1.0.0.
Use FMC_getExtension instead )
```

3.54.1.4 FMC_FUNC_UNAVAILABLE() [4/4]

```
FMC_FUNC_UNAVAILABLE (
             This function is not anymore available in the library since the version 1.0.  0.
)
```

3.54.1.5 FMC_TYPE_UNAVAILABLE()

```
FMC_TYPE_UNAVAILABLE (
             This type is not anymore available in the library since the version 1.0.  0.  )
```

Definition at line 8 of file FMC_deprecated.h.

3.54.2 Variable Documentation

3.54.2.1 extension

```
char* extension
```

Definition at line 28 of file FMC_deprecated.h.

3.54.2.2 fileName

```
char* fileName
```

Definition at line 22 of file FMC_deprecated.h.

3.54.2.3 filePath

```
char* filePath
```

Definition at line 25 of file FMC_deprecated.h.

3.54.2.4 pathToCopy

```
char** pathToCopy
```

Definition at line 19 of file FMC_deprecated.h.

3.54.2.5 toSearch

```
char* toSearch
```

Definition at line 40 of file FMC_deprecated.h.

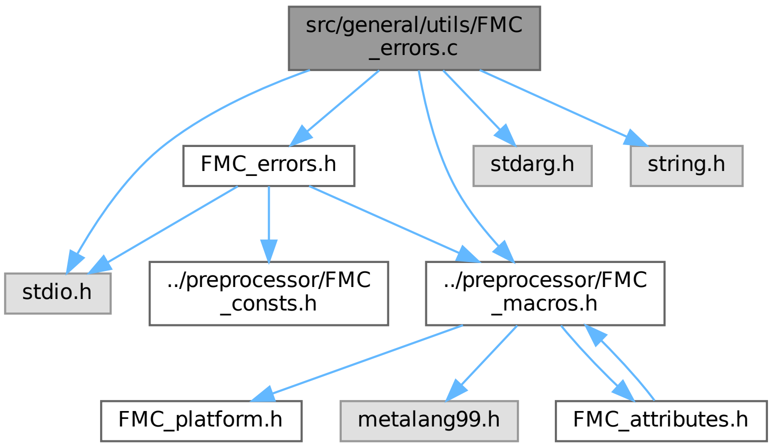
3.55 FMC_deprecated.h

Go to the documentation of this file.

```
#ifndef FMC_DEPRECATED_H
#define FMC_DEPRECATED_H

#include "../preprocessor/FMC_attributes.h"
#include <stddef.h>

#if !defined(BUILDING_FMANC)
FMC_TYPE_UNAVAILABLE(This type is not anymore available in the library since the version 1.0.0.)
struct FMANC_SO
{
    size_t charCount;
    long long int *pos;
};

FMC_TYPE_UNAVAILABLE(This type is not anymore available in the library since the version 1.0.0.)
typedef struct FMANC_SO stringOccurrences;

FMC_FUNC_UNAVAILABLE(This function is not anymore available in the library since the version 1.0.0.)
char *copyFileWithoutTabAndLineBreak(char *sourceFilePath, char **pathToCopy);

FMC_FUNC_UNAVAILABLE(This function is not anymore available in the library since the version 1.0.0.
   Use FMC_extractFilename instead)
void fgetFileName(char *sourceFilePath, char *fileName);

FMC_FUNC_UNAVAILABLE(This function is not anymore available in the library since the version 1.0.0.
   Use FMC_cutFilename instead)
void fgetFilePath(char *sourceFilePath, char *filePath);

FMC_FUNC_UNAVAILABLE(This function is not anymore available in the library since the version 1.0.0.
   Use FMC_getExtension instead)
void fgetFileExtension(char *sourceFilePath, char *extension);

FMC_FUNC_UNAVAILABLE(This function is not anymore available in the library since the version 1.0.0.)
size_t countCharInFile(char *filePath);

FMC_FUNC_UNAVAILABLE(This function is not anymore available in the library since the version 1.0.0.)
stringOccurrences *init_StringOccurences(size_t sizeOfString);

FMC_FUNC_UNAVAILABLE(This function is not anymore available in the library since the version 1.0.0.)
void free_stringOccurrences();

FMC_FUNC_UNAVAILABLE(This function is not anymore available in the library since the version 1.0.0.)
stringOccurrences *searchStringInFile(char *filePath, char *toSearch);

FMC_FUNC_UNAVAILABLE(This function is not anymore available in the library since the version 1.0.0.)
int deleteCStyleComments(char *filePath);

#endif // BUILDING_FMANC
#endif // FMC_DEPRECATED_H
```

3.56 src/general/utils/FMC_errors.c File Reference

Include dependency graph for FMC_errors.c:


src/general/utils/FMC_errors.c
FMC_errors.h
stdarg.h
string.h
stdio.h
../preprocessor/FMC_consts.h
../preprocessor/FMC_macros.h
FMC_platform.h
metalang99.h
FMC_attributes.h

Functions

- void FMC_changeStreamTextColorToBlue (FILE ∗stream)
- void FMC_changeStreamTextColorToBrightBlue (FILE ∗stream)
- void FMC_changeStreamTextColorToBrightCyan (FILE ∗stream)
- void FMC_changeStreamTextColorToBrightGreen (FILE ∗stream)
- void FMC_changeStreamTextColorToBrightMagenta (FILE ∗stream)
- void FMC_changeStreamTextColorToBrightRed (FILE ∗stream)
- void FMC_changeStreamTextColorToBrightWhite (FILE ∗stream)
- void FMC_changeStreamTextColorToBrightYellow (FILE ∗stream)
- void FMC_changeStreamTextColorToCyan (FILE ∗stream)
- void FMC_changeStreamTextColorToGreen (FILE ∗stream)
- void FMC_changeStreamTextColorToMagenta (FILE ∗stream)
- void FMC_changeStreamTextColorToRed (FILE ∗stream)
- void FMC_changeStreamTextColorToWhite (FILE ∗stream)
- void FMC_changeStreamTextColorToYellow (FILE ∗stream)
- void FMC_makeMsg_f (char ∗buff, unsigned int argc,...)
- void FMC_printBlueError (FILE ∗stream, const char ∗text)
- void FMC_printBlueText (FILE ∗stream, const char ∗text)
- void FMC_printBrightBlueError (FILE ∗stream, const char ∗text)
- void FMC_printBrightBlueText (FILE ∗stream, const char ∗text)
- void FMC_printBrightCyanError (FILE ∗stream, const char ∗text)
- void FMC_printBrightCyanText (FILE ∗stream, const char ∗text)
- void FMC_printBrightGreenError (FILE ∗stream, const char ∗text)
- void FMC_printBrightGreenText (FILE ∗stream, const char ∗text)
- void FMC_printBrightMagentaError (FILE ∗stream, const char ∗text)
- void FMC_printBrightMagentaText (FILE ∗stream, const char ∗text)
- void FMC_printBrightRedError (FILE ∗stream, const char ∗text)
- void FMC_printBrightRedText (FILE ∗stream, const char ∗text)

- void FMC_printBrightWhiteError (FILE ∗stream, const char ∗text)
- void FMC_printBrightWhiteText (FILE ∗stream, const char ∗text)
- void FMC_printBrightYellowError (FILE ∗stream, const char ∗text)
- void FMC_printBrightYellowText (FILE ∗stream, const char ∗text)
- void FMC_printCyanError (FILE ∗stream, const char ∗text)
- void FMC_printCyanText (FILE ∗stream, const char ∗text)
- void FMC_printGreenError (FILE ∗stream, const char ∗text)
- void FMC_printGreenText (FILE ∗stream, const char ∗text)
- void FMC_printMagentaError (FILE ∗stream, const char ∗text)
- void FMC_printMagentaText (FILE ∗stream, const char ∗text)
- void FMC_printRedError (FILE ∗stream, const char ∗text)
- void FMC_printRedText (FILE ∗stream, const char ∗text)
- void FMC_printWhiteError (FILE ∗stream, const char ∗text)
- void FMC_printWhiteText (FILE ∗stream, const char ∗text)
- void FMC_printYellowError (FILE ∗stream, const char ∗text)
- void FMC_printYellowText (FILE ∗stream, const char ∗text)
- void FMC_resetStreamOutputStyle (FILE ∗stream)

3.56.1 Function Documentation

3.56.1.1 FMC_changeStreamTextColorToBlue()

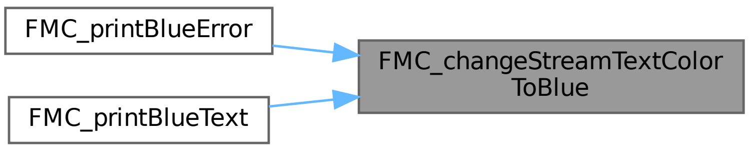
```
void FMC_changeStreamTextColorToBlue (
            FILE ∗ stream )
```

Definition at line 63 of file FMC_errors.h.

References FG_BLUE.

Referenced by FMC_printBlueError(), and FMC_printBlueText().

Here is the caller graph for this function:



3.56.1.2 FMC_changeStreamTextColorToBrightBlue()

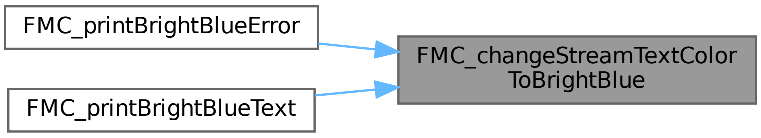
```
void FMC_changeStreamTextColorToBrightBlue (
            FILE * stream )
```

Definition at line 98 of file FMC_errors.h.

References FG_BRIGHT_BLUE.

Referenced by FMC_printBrightBlueError(), and FMC_printBrightBlueText().

Here is the caller graph for this function:



3.56.1.3 FMC_changeStreamTextColorToBrightCyan()

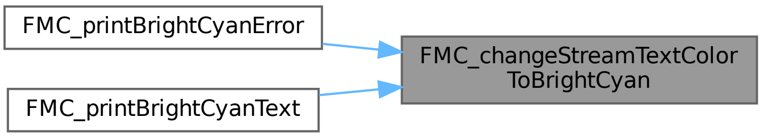
```
void FMC_changeStreamTextColorToBrightCyan (
            FILE * stream )
```

Definition at line 108 of file FMC_errors.h.

References FG_BRIGHT_CYAN.

Referenced by FMC_printBrightCyanError(), and FMC_printBrightCyanText().

Here is the caller graph for this function:



3.56.1.4 FMC_changeStreamTextColorToBrightGreen()

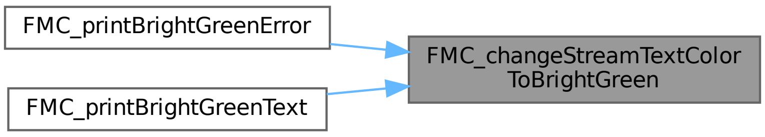
```
void FMC_changeStreamTextColorToBrightGreen (
            FILE * stream )
```

Definition at line 88 of file FMC_errors.h.

References FG_BRIGHT_GREEN.

Referenced by FMC_printBrightGreenError(), and FMC_printBrightGreenText().

Here is the caller graph for this function:


FMC_printBrightGreenError
FMC_printBrightGreenText
FMC_changeStreamTextColor ToBrightGreen

3.56.1.5 FMC_changeStreamTextColorToBrightMagenta()

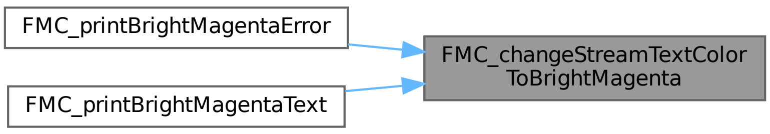
```
void FMC_changeStreamTextColorToBrightMagenta (
            FILE * stream )
```

Definition at line 103 of file FMC_errors.h.

References FG_BRIGHT_MAGENTA.

Referenced by FMC_printBrightMagentaError(), and FMC_printBrightMagentaText().

Here is the caller graph for this function:


FMC_printBrightMagentaError
FMC_printBrightMagentaText
FMC_changeStreamTextColor ToBrightMagenta

3.56.1.6 FMC_changeStreamTextColorToBrightRed()

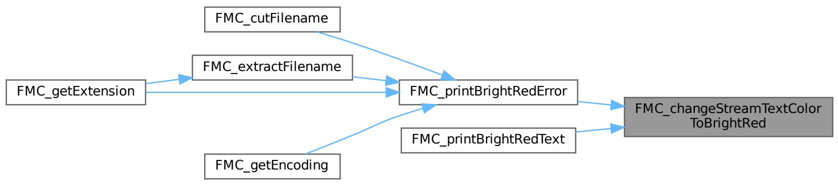
```
void FMC_changeStreamTextColorToBrightRed (
            FILE * stream )
```

Definition at line 83 of file FMC_errors.h.

References FG_BRIGHT_RED.

Referenced by FMC_printBrightRedError(), and FMC_printBrightRedText().

Here is the caller graph for this function:



3.56.1.7 FMC_changeStreamTextColorToBrightWhite()

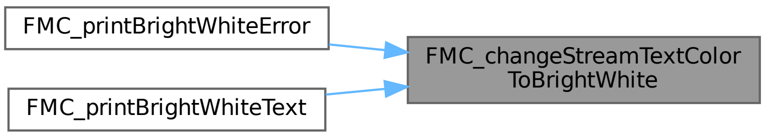
```
void FMC_changeStreamTextColorToBrightWhite (
            FILE * stream )
```

Definition at line 113 of file FMC_errors.h.

References FG_BRIGHT_WHITE.

Referenced by FMC_printBrightWhiteError(), and FMC_printBrightWhiteText().

Here is the caller graph for this function:


FMC_printBrightWhiteError
FMC_changeStreamTextColor ToBrightWhite
FMC_printBrightWhiteText

3.56.1.8 FMC_changeStreamTextColorToBrightYellow()

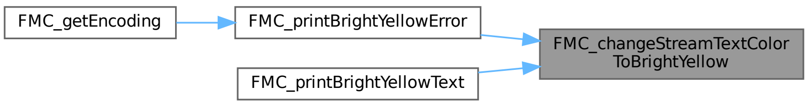
```
void FMC_changeStreamTextColorToBrightYellow (
             FILE * stream )
```

Definition at line 93 of file FMC_errors.h.

References FG_BRIGHT_YELLOW.

Referenced by FMC_printBrightYellowError(), and FMC_printBrightYellowText().

Here is the caller graph for this function:


FMC_getEncoding
FMC_printBrightYellowError
FMC_printBrightYellowText
FMC_changeStreamTextColor ToBrightYellow

3.56.1.9 FMC_changeStreamTextColorToCyan()

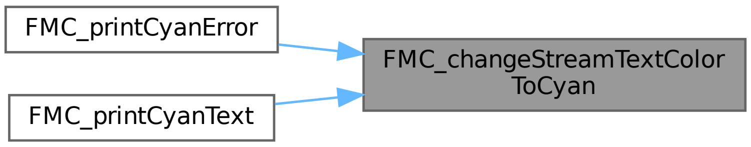
```
void FMC_changeStreamTextColorToCyan (
             FILE * stream )
```

Definition at line 73 of file FMC_errors.h.

References FG_CYAN.

Referenced by FMC_printCyanError(), and FMC_printCyanText().

Here is the caller graph for this function:



3.56.1.10 FMC_changeStreamTextColorToGreen()

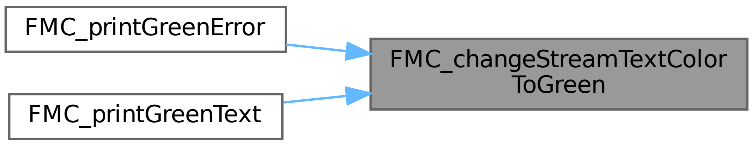
```
void FMC_changeStreamTextColorToGreen (
            FILE * stream )
```

Definition at line 53 of file FMC_errors.h.

References FG_GREEN.

Referenced by FMC_printGreenError(), and FMC_printGreenText().

Here is the caller graph for this function:



3.56.1.11 FMC_changeStreamTextColorToMagenta()

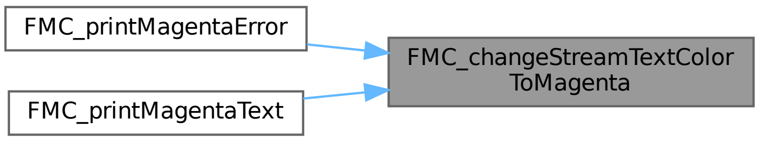
```
void FMC_changeStreamTextColorToMagenta (
            FILE * stream )
```

Definition at line 68 of file FMC_errors.h.

References FG_MAGENTA.

Referenced by FMC_printMagentaError(), and FMC_printMagentaText().

Here is the caller graph for this function:



3.56.1.12 FMC_changeStreamTextColorToRed()

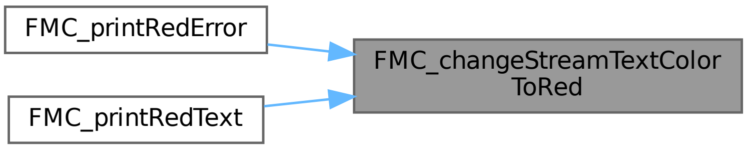
```
void FMC_changeStreamTextColorToRed (
            FILE * stream )
```

Definition at line 48 of file FMC_errors.h.

References FG_RED.

Referenced by FMC_printRedError(), and FMC_printRedText().

Here is the caller graph for this function:



3.56.1.13 FMC_changeStreamTextColorToWhite()

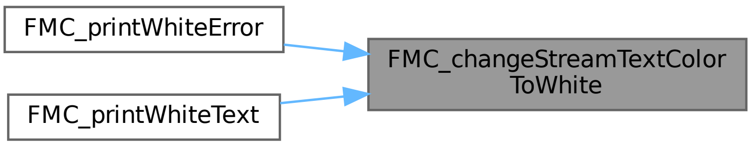
```
void FMC_changeStreamTextColorToWhite (
            FILE * stream )
```

Definition at line 78 of file FMC_errors.h.

References FG_WHITE.

Referenced by FMC_printWhiteError(), and FMC_printWhiteText().

Here is the caller graph for this function:



3.56.1.14 FMC_changeStreamTextColorToYellow()

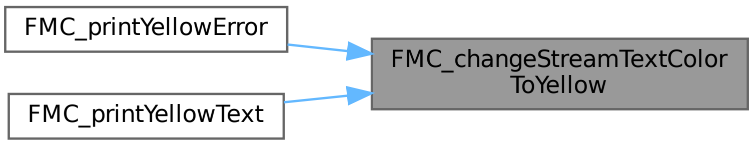
```
void FMC_changeStreamTextColorToYellow (
             FILE * stream )
```

Definition at line 58 of file FMC_errors.h.

References FG_YELLOW.

Referenced by FMC_printYellowError(), and FMC_printYellowText().

Here is the caller graph for this function:



3.56.1.15 FMC_makeMsg_f()

```
void FMC_makeMsg_f (
             char * buff,
             unsigned int argc,
              ... )
```

Definition at line 33 of file FMC_errors.c.

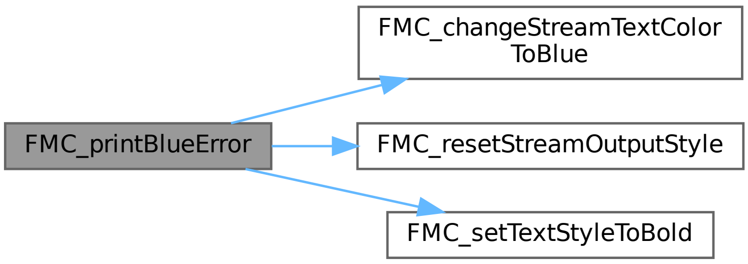
3.56.1.16 FMC_printBlueError()

```
void FMC_printBlueError (
             FILE * stream,
             const char * text )
```

Definition at line 341 of file FMC_errors.h.

References FMC_changeStreamTextColorToBlue(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:



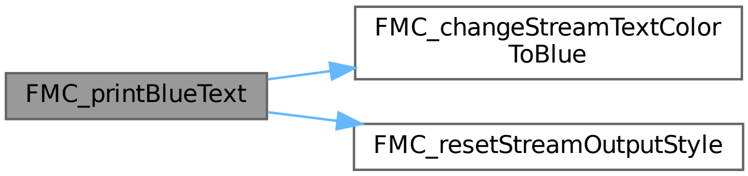
3.56.1.17 FMC_printBlueText()

```
void FMC_printBlueText (
            FILE * stream,
            const char * text )
```

Definition at line 240 of file FMC_errors.h.

References FMC_changeStreamTextColorToBlue(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:



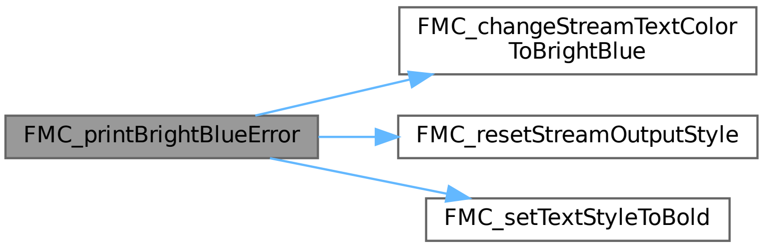
3.56.1.18 FMC_printBrightBlueError()

```
void FMC_printBrightBlueError (
             FILE * stream,
             const char * text )
```

Definition at line 397 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightBlue(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:


FMC_changeStreamTextColor ToBrightBlue
FMC_printBrightBlueError
FMC_resetStreamOutputStyle
FMC_setTextStyleToBold

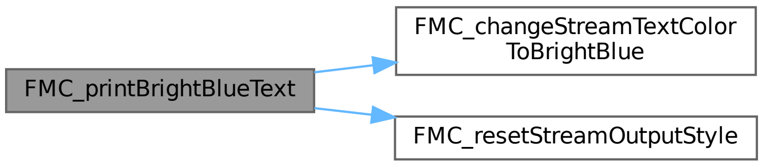
3.56.1.19 FMC_printBrightBlueText()

```
void FMC_printBrightBlueText (
             FILE * stream,
             const char * text )
```

Definition at line 289 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightBlue(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:


FMC_changeStreamTextColor ToBrightBlue
FMC_printBrightBlueText
FMC_resetStreamOutputStyle

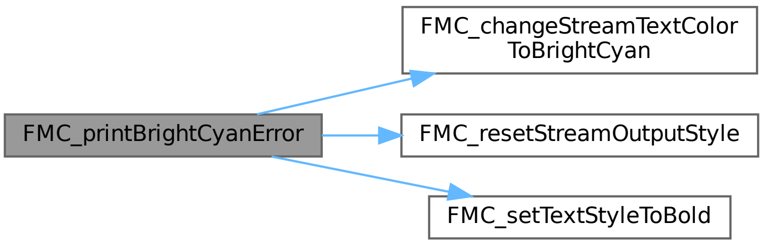
3.56.1.20 FMC_printBrightCyanError()

```
void FMC_printBrightCyanError (
            FILE * stream,
            const char * text )
```

Definition at line 413 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightCyan(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:


FMC_changeStreamTextColor ToBrightCyan
FMC_printBrightCyanError
FMC_resetStreamOutputStyle
FMC_setTextStyleToBold

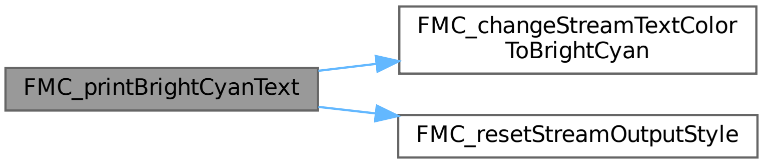
3.56.1.21 FMC_printBrightCyanText()

```
void FMC_printBrightCyanText (
            FILE * stream,
            const char * text )
```

Definition at line 303 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightCyan(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:


FMC_changeStreamTextColor ToBrightCyan
FMC_printBrightCyanText
FMC_resetStreamOutputStyle

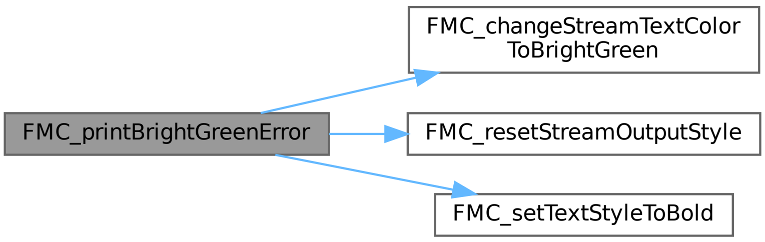
3.56.1.22 FMC_printBrightGreenError()

```
void FMC_printBrightGreenError (
             FILE * stream,
             const char * text )
```

Definition at line 381 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightGreen(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:


FMC_printBrightGreenError
FMC_changeStreamTextColor ToBrightGreen
FMC_resetStreamOutputStyle
FMC_setTextStyleToBold

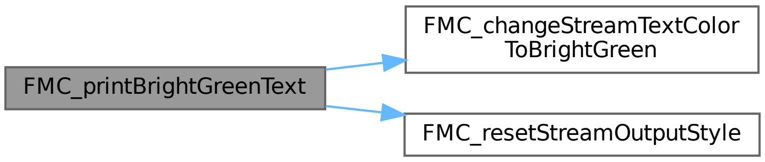
3.56.1.23 FMC_printBrightGreenText()

```
void FMC_printBrightGreenText (
             FILE * stream,
             const char * text )
```

Definition at line 275 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightGreen(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:


FMC_printBrightGreenText
FMC_changeStreamTextColor ToBrightGreen
FMC_resetStreamOutputStyle

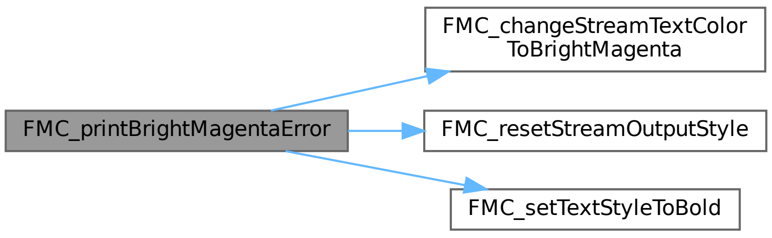
3.56.1.24 FMC_printBrightMagentaError()

```
void FMC_printBrightMagentaError (
            FILE * stream,
            const char * text )
```

Definition at line 405 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightMagenta(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:


FMC_changeStreamTextColor ToBrightMagenta
FMC_printBrightMagentaError
FMC_resetStreamOutputStyle
FMC_setTextStyleToBold

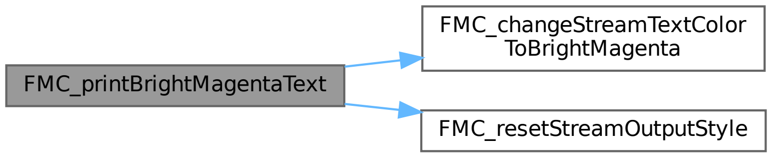
3.56.1.25 FMC_printBrightMagentaText()

```
void FMC_printBrightMagentaText (
            FILE * stream,
            const char * text )
```

Definition at line 296 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightMagenta(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:


FMC_changeStreamTextColor ToBrightMagenta
FMC_printBrightMagentaText
FMC_resetStreamOutputStyle

3.56.1.26 FMC_printBrightRedError()

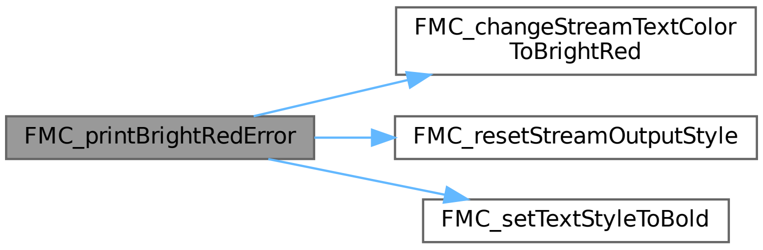
```
void FMC_printBrightRedError (
            FILE * stream,
            const char * text )
```

Definition at line 373 of file FMC_errors.h.

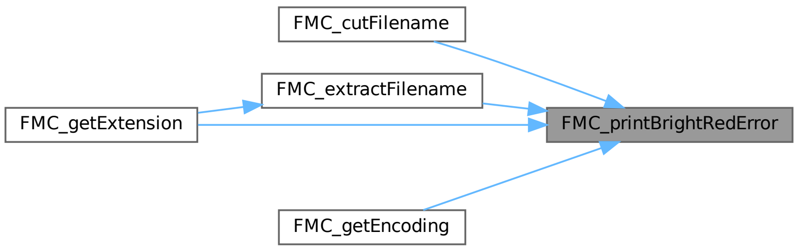
References FMC_changeStreamTextColorToBrightRed(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Referenced by FMC_cutFilename(), FMC_extractFilename(), FMC_getEncoding(), and FMC_getExtension().

Here is the call graph for this function:


FMC_changeStreamTextColor ToBrightRed
FMC_printBrightRedError
FMC_resetStreamOutputStyle
FMC_setTextStyleToBold

Here is the caller graph for this function:



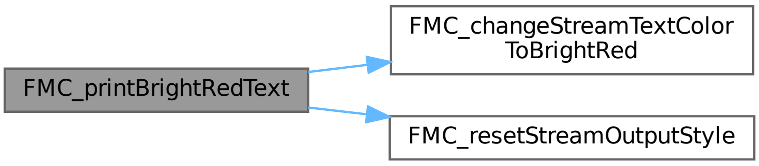
3.56.1.27 FMC_printBrightRedText()

```
void FMC_printBrightRedText (
            FILE * stream,
            const char * text )
```

Definition at line 268 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightRed(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:


FMC_printBrightRedText
FMC_changeStreamTextColor ToBrightRed
FMC_resetStreamOutputStyle

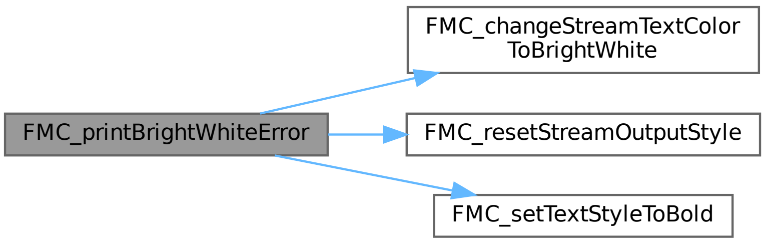
3.56.1.28 FMC_printBrightWhiteError()

```
void FMC_printBrightWhiteError (
            FILE * stream,
            const char * text )
```

Definition at line 421 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightWhite(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:


FMC_printBrightWhiteError
FMC_changeStreamTextColor ToBrightWhite
FMC_resetStreamOutputStyle
FMC_setTextStyleToBold

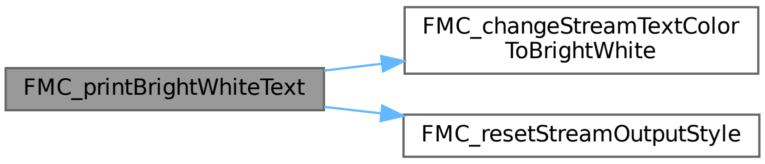
3.56.1.29 FMC_printBrightWhiteText()

```
void FMC_printBrightWhiteText (
            FILE * stream,
            const char * text )
```

Definition at line 310 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightWhite(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:


FMC_printBrightWhiteText
FMC_changeStreamTextColor ToBrightWhite
FMC_resetStreamOutputStyle

3.56.1.30 FMC_printBrightYellowError()

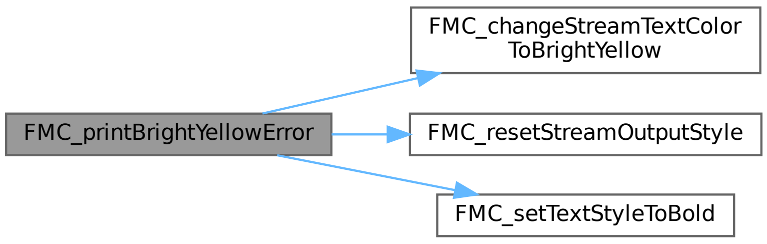
```
void FMC_printBrightYellowError (
            FILE * stream,
            const char * text )
```

Definition at line 389 of file FMC_errors.h.

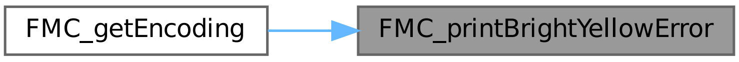
References FMC_changeStreamTextColorToBrightYellow(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Referenced by FMC_getEncoding().

Here is the call graph for this function:


FMC_printBrightYellowError
FMC_changeStreamTextColor ToBrightYellow
FMC_resetStreamOutputStyle
FMC_setTextStyleToBold

Here is the caller graph for this function:



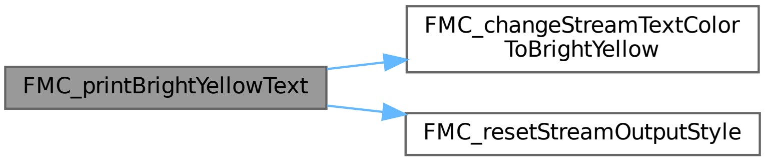
3.56.1.31 FMC_printBrightYellowText()

```
void FMC_printBrightYellowText (
            FILE * stream,
            const char * text )
```

Definition at line 282 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightYellow(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:


FMC_printBrightYellowText
FMC_changeStreamTextColor ToBrightYellow
FMC_resetStreamOutputStyle

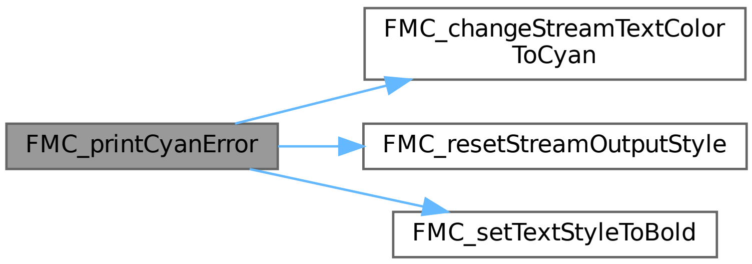
3.56.1.32 FMC_printCyanError()

```
void FMC_printCyanError (
            FILE * stream,
            const char * text )
```

Definition at line 357 of file FMC_errors.h.

References FMC_changeStreamTextColorToCyan(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:



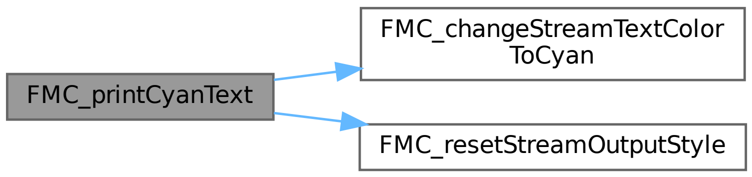
3.56.1.33 FMC_printCyanText()

```
void FMC_printCyanText (
             FILE * stream,
             const char * text )
```

Definition at line 254 of file FMC_errors.h.

References FMC_changeStreamTextColorToCyan(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:



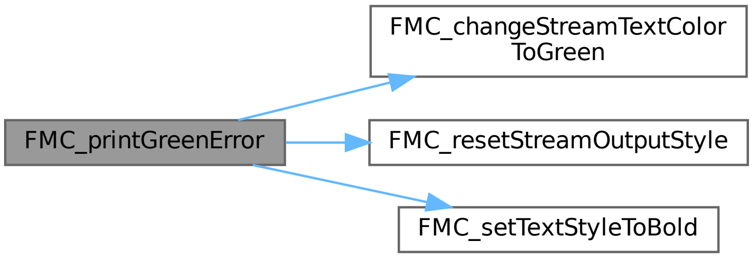
3.56.1.34 FMC_printGreenError()

```
void FMC_printGreenError (
             FILE * stream,
             const char * text )
```

Definition at line 325 of file FMC_errors.h.

References FMC_changeStreamTextColorToGreen(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:



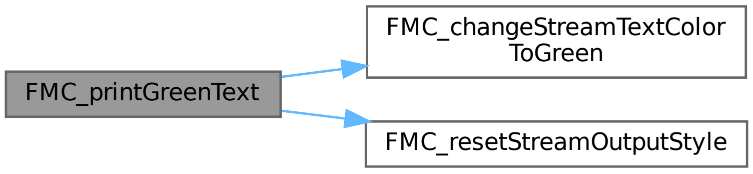
3.56.1.35 FMC_printGreenText()

```
void FMC_printGreenText (
             FILE * stream,
             const char * text )
```

Definition at line 226 of file FMC_errors.h.

References FMC_changeStreamTextColorToGreen(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:



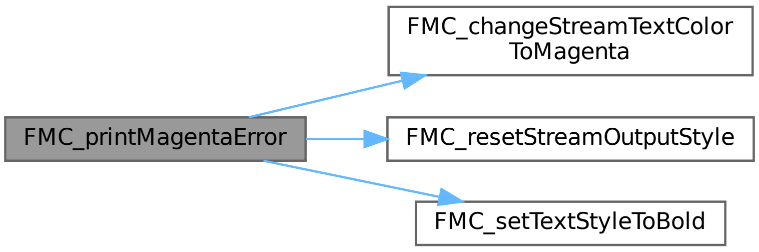
3.56.1.36 FMC_printMagentaError()

```
void FMC_printMagentaError (
            FILE * stream,
            const char * text )
```

Definition at line 349 of file FMC_errors.h.

References FMC_changeStreamTextColorToMagenta(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:


FMC_printMagentaError
FMC_changeStreamTextColor ToMagenta
FMC_resetStreamOutputStyle
FMC_setTextStyleToBold

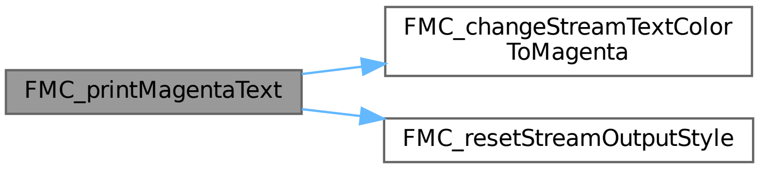
3.56.1.37 FMC_printMagentaText()

```
void FMC_printMagentaText (
            FILE * stream,
            const char * text )
```

Definition at line 247 of file FMC_errors.h.

References FMC_changeStreamTextColorToMagenta(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:


FMC_printMagentaText
FMC_changeStreamTextColor ToMagenta
FMC_resetStreamOutputStyle

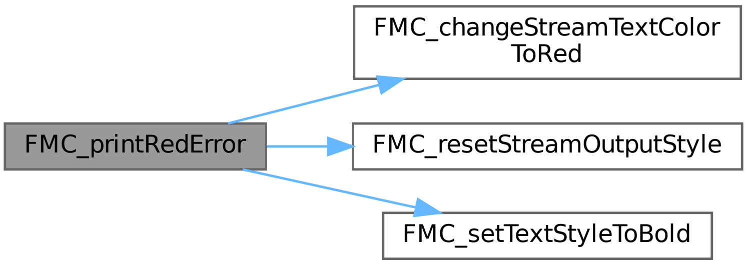
3.56.1.38 FMC_printRedError()

```
void FMC_printRedError (
            FILE * stream,
            const char * text )
```

Definition at line 317 of file FMC_errors.h.

References FMC_changeStreamTextColorToRed(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:



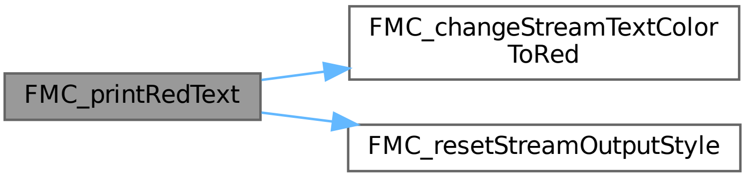
3.56.1.39 FMC_printRedText()

```
void FMC_printRedText (
            FILE * stream,
            const char * text )
```

Definition at line 219 of file FMC_errors.h.

References FMC_changeStreamTextColorToRed(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:



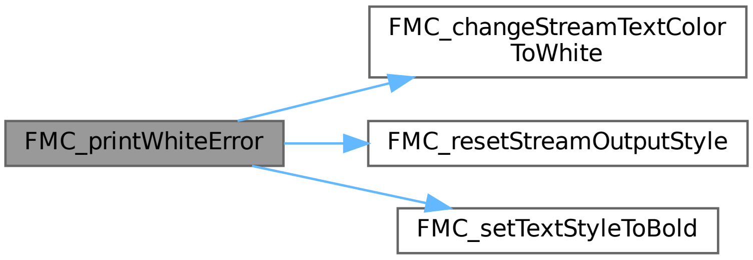
3.56.1.40 FMC_printWhiteError()

```
void FMC_printWhiteError (
             FILE * stream,
             const char * text )
```

Definition at line 365 of file FMC_errors.h.

References FMC_changeStreamTextColorToWhite(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:



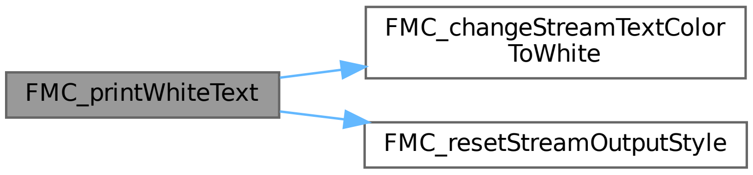
3.56.1.41 FMC_printWhiteText()

```
void FMC_printWhiteText (
             FILE * stream,
             const char * text )
```

Definition at line 261 of file FMC_errors.h.

References FMC_changeStreamTextColorToWhite(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:



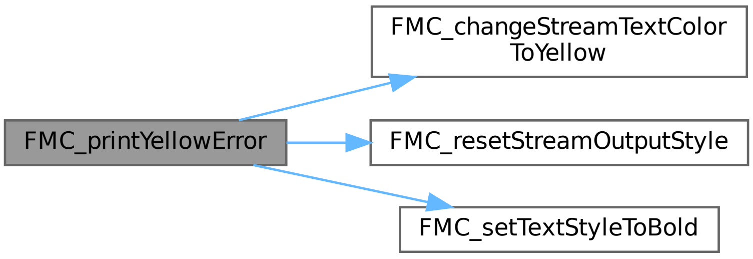
3.56.1.42 FMC_printYellowError()

```
void FMC_printYellowError (
            FILE * stream,
            const char * text )
```

Definition at line 333 of file FMC_errors.h.

References FMC_changeStreamTextColorToYellow(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:


FMC_printYellowError
FMC_changeStreamTextColor ToYellow
FMC_resetStreamOutputStyle
FMC_setTextStyleToBold

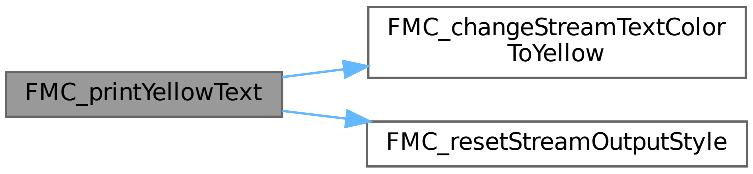
3.56.1.43 FMC_printYellowText()

```
void FMC_printYellowText (
            FILE * stream,
            const char * text )
```

Definition at line 233 of file FMC_errors.h.

References FMC_changeStreamTextColorToYellow(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:


FMC_printYellowText
FMC_changeStreamTextColor ToYellow
FMC_resetStreamOutputStyle

3.56.1.44 FMC_resetStreamOutputStyle()

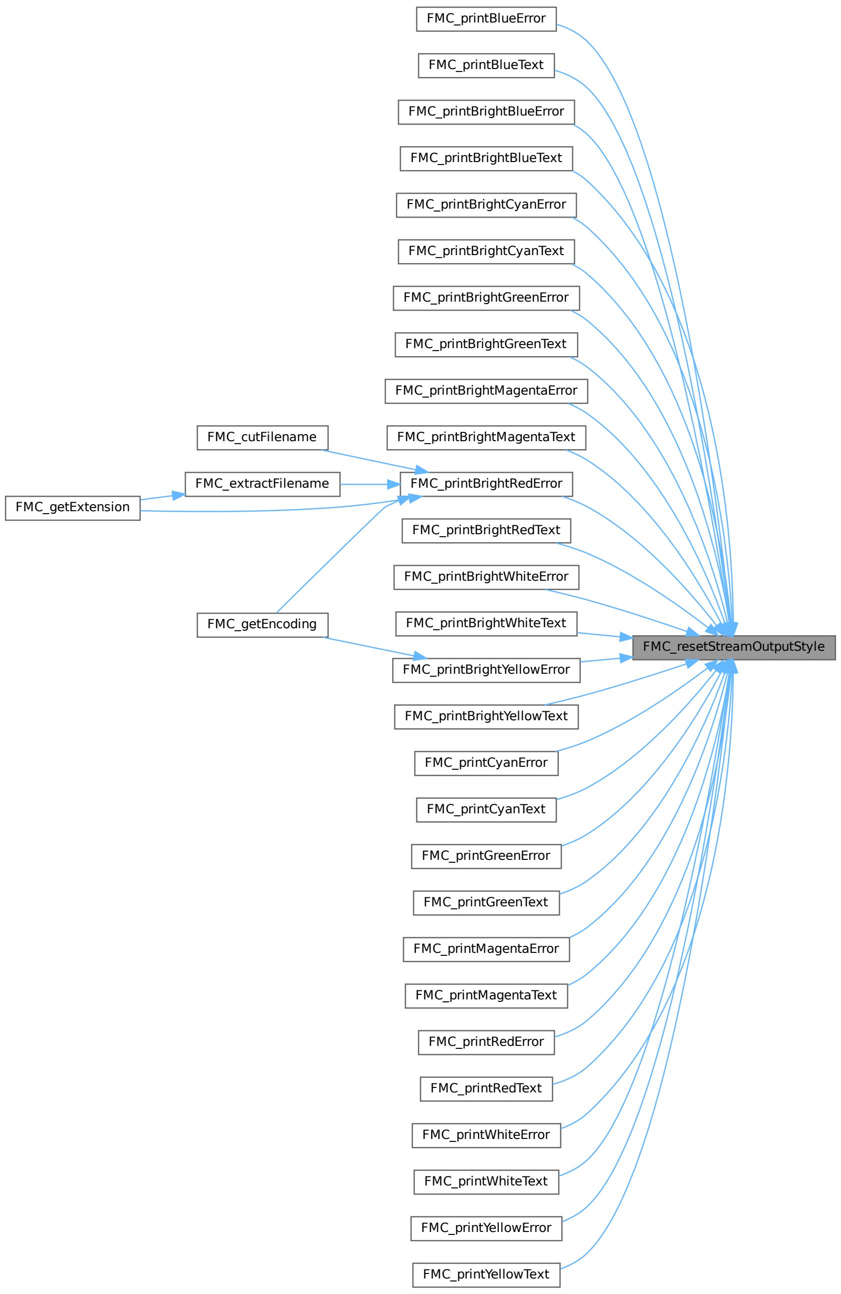
```
void FMC_resetStreamOutputStyle (
            FILE * stream )
```

Definition at line 42 of file FMC_errors.h.

References RESET.

Referenced by FMC_printBlueError(), FMC_printBlueText(), FMC_printBrightBlueError(), FMC_printBrightBlueText(), FMC_printBrightCyanError(), FMC_printBrightCyanText(), FMC_printBrightGreenError(), FMC_printBrightGreenText(), FMC_printBrightMagentaError(), FMC_printBrightMagentaText(), FMC_printBrightRedError(), FMC_printBrightRedText(), FMC_printBrightWhiteError(), FMC_printBrightWhiteText(), FMC_printBrightYellowError(), FMC_printBrightYellowText(), FMC_printCyanError(), FMC_printCyanText(), FMC_printGreenError(), FMC_printGreenText(), FMC_printMagentaError(), FMC_printMagentaText(), FMC_printRedError(), FMC_printRedText(), FMC_printWhiteError(), FMC_printWhiteText(), FMC_printYellowError(), and FMC_printYellowText().

Here is the caller graph for this function:



3.57 FMC_errors.c

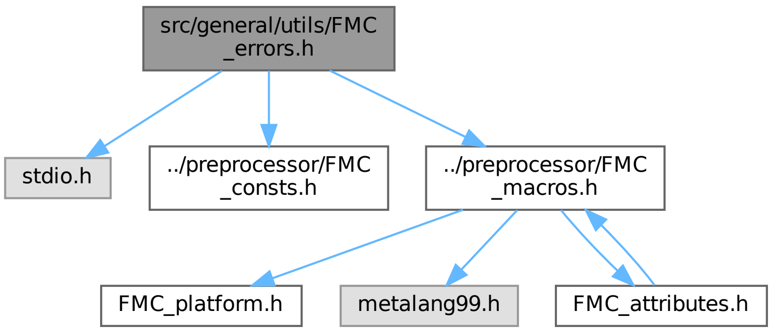
Go to the documentation of this file.

```
/*

MIT License

Copyright (c) 2022 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
```

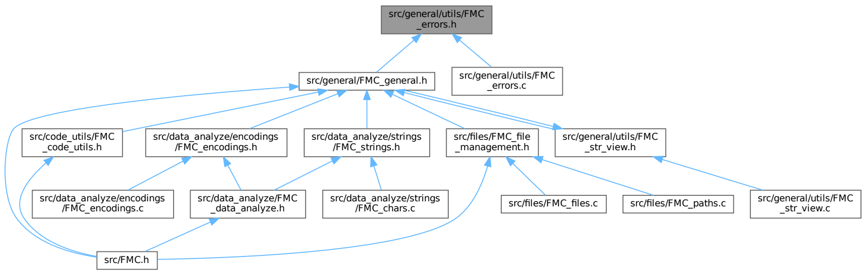
```
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/

#include "FMC_errors.h"
#include "../preprocessor/FMC_macros.h"
#include <stdio.h>
#include <stdarg.h>
#include <string.h>

FMC_SHARED FMC_FUNC_NONNULL(1) void FMC_makeMsg_f(char *buff, unsigned int argc, ...)
{
    va_list args;
    va_start(args, argc);
    for (unsigned int i = 0; i < argc; i++)
    {
        char *arg = va_arg(args, char *);
        buff = strcat(buff, arg);
    }
    va_end(args);
}

extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_resetStreamOutputStyle(FILE *stream);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToRed(FILE *stream);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToGreen(FILE *stream);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToYellow(FILE *stream);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToBlue(FILE *stream);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToMagenta(FILE *stream);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToCyan(FILE *stream);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToWhite(FILE *stream);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToBrightRed(FILE *stream);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToBrightGreen(FILE *stream);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToBrightYellow(FILE *stream);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToBrightBlue(FILE *stream);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToBrightMagenta(FILE *stream);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToBrightCyan(FILE *stream);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToBrightWhite(FILE *stream);

extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printRedText(FILE *stream, const char *text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printGreenText(FILE *stream, const char *text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printYellowText(FILE *stream, const char *text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBlueText(FILE *stream, const char *text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printMagentaText(FILE *stream, const char *text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printCyanText(FILE *stream, const char *text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printWhiteText(FILE *stream, const char *text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightRedText(FILE *stream, const char *text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightGreenText(FILE *stream, const char *text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightYellowText(FILE *stream, const char
*text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightBlueText(FILE *stream, const char *text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightMagentaText(FILE *stream, const char
*text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightCyanText(FILE *stream, const char *text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightWhiteText(FILE *stream, const char *text);

extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printRedError(FILE *stream, const char *text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printGreenError(FILE *stream, const char *text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printYellowError(FILE *stream, const char *text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBlueError(FILE *stream, const char *text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printMagentaError(FILE *stream, const char *text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printCyanError(FILE *stream, const char *text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printWhiteError(FILE *stream, const char *text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightRedError(FILE *stream, const char *text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightGreenError(FILE *stream, const char
*text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightYellowError(FILE *stream, const char
*text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightBlueError(FILE *stream, const char *text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightMagentaError(FILE *stream, const char
*text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightCyanError(FILE *stream, const char *text);
extern FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightWhiteError(FILE *stream, const char
*text);
```

3.58 src/general/utils/FMC_errors.h File Reference

Include dependency graph for FMC_errors.h:


src/general/utils/FMC_errors.h
stdio.h
../preprocessor/FMC_consts.h
../preprocessor/FMC_macros.h
FMC_platform.h
metalang99.h
FMC_attributes.h

This graph shows which files directly or indirectly include this file:


src/general/utils/FMC_errors.h
src/general/FMC_general.h
src/general/utils/FMC_errors.c
src/code_utils/FMC_code_utils.h
src/data_analyze/encodings/FMC_encodings.h
src/data_analyze/strings/FMC_strings.h
src/files/FMC_file_management.h
src/general/utils/FMC_str_view.h
src/data_analyze/encodings/FMC_encodings.c
src/data_analyze/FMC_data_analyze.h
src/data_analyze/strings/FMC_chars.c
src/files/FMC_files.c
src/files/FMC_paths.c
src/general/utils/FMC_str_view.c
src/FMC.h

Macros

- #define FMC_ERRORS
- #define FMC_makeMsg(err_var_name, argc, ...)

Functions

- void FMC_changeStreamTextColorToBlue (FILE ∗stream)
- void FMC_changeStreamTextColorToBrightBlue (FILE ∗stream)
- void FMC_changeStreamTextColorToBrightCyan (FILE ∗stream)
- void FMC_changeStreamTextColorToBrightGreen (FILE ∗stream)
- void FMC_changeStreamTextColorToBrightMagenta (FILE ∗stream)
- void FMC_changeStreamTextColorToBrightRed (FILE ∗stream)
- void FMC_changeStreamTextColorToBrightWhite (FILE ∗stream)
- void FMC_changeStreamTextColorToBrightYellow (FILE ∗stream)
- void FMC_changeStreamTextColorToCyan (FILE ∗stream)
- void FMC_changeStreamTextColorToGreen (FILE ∗stream)

- void FMC_changeStreamTextColorToMagenta (FILE ∗stream)
- void FMC_changeStreamTextColorToRed (FILE ∗stream)
- void FMC_changeStreamTextColorToWhite (FILE ∗stream)
- void FMC_changeStreamTextColorToYellow (FILE ∗stream)
- void FMC_makeMsg_f (char ∗buff, unsigned int argc,...)
- void FMC_printBlueError (FILE ∗stream, const char ∗text)
- void FMC_printBlueText (FILE ∗stream, const char ∗text)
- void FMC_printBrightBlueError (FILE ∗stream, const char ∗text)
- void FMC_printBrightBlueText (FILE ∗stream, const char ∗text)
- void FMC_printBrightCyanError (FILE ∗stream, const char ∗text)
- void FMC_printBrightCyanText (FILE ∗stream, const char ∗text)
- void FMC_printBrightGreenError (FILE ∗stream, const char ∗text)
- void FMC_printBrightGreenText (FILE ∗stream, const char ∗text)
- void FMC_printBrightMagentaError (FILE ∗stream, const char ∗text)
- void FMC_printBrightMagentaText (FILE ∗stream, const char ∗text)
- void FMC_printBrightRedError (FILE ∗stream, const char ∗text)
- void FMC_printBrightRedText (FILE ∗stream, const char ∗text)
- void FMC_printBrightWhiteError (FILE ∗stream, const char ∗text)
- void FMC_printBrightWhiteText (FILE ∗stream, const char ∗text)
- void FMC_printBrightYellowError (FILE ∗stream, const char ∗text)
- void FMC_printBrightYellowText (FILE ∗stream, const char ∗text)
- void FMC_printCyanError (FILE ∗stream, const char ∗text)
- void FMC_printCyanText (FILE ∗stream, const char ∗text)
- void FMC_printGreenError (FILE ∗stream, const char ∗text)
- void FMC_printGreenText (FILE ∗stream, const char ∗text)
- void FMC_printMagentaError (FILE ∗stream, const char ∗text)
- void FMC_printMagentaText (FILE ∗stream, const char ∗text)
- void FMC_printRedError (FILE ∗stream, const char ∗text)
- void FMC_printRedText (FILE ∗stream, const char ∗text)
- void FMC_printWhiteError (FILE ∗stream, const char ∗text)
- void FMC_printWhiteText (FILE ∗stream, const char ∗text)
- void FMC_printYellowError (FILE ∗stream, const char ∗text)
- void FMC_printYellowText (FILE ∗stream, const char ∗text)
- void FMC_resetStreamOutputStyle (FILE ∗stream)
- void FMC_setBGStreamColorToBlue (FILE ∗stream)
- void FMC_setBGStreamColorToBrightBlue (FILE ∗stream)
- void FMC_setBGStreamColorToBrightCyan (FILE ∗stream)
- void FMC_setBGStreamColorToBrightGreen (FILE ∗stream)
- void FMC_setBGStreamColorToBrightMagenta (FILE ∗stream)
- void FMC_setBGStreamColorToBrightRed (FILE ∗stream)
- void FMC_setBGStreamColorToBrightWhite (FILE ∗stream)
- void FMC_setBGStreamColorToBrightYellow (FILE ∗stream)
- void FMC_setBGStreamColorToCyan (FILE ∗stream)
- void FMC_setBGStreamColorToGreen (FILE ∗stream)
- void FMC_setBGStreamColorToMagenta (FILE ∗stream)
- void FMC_setBGStreamColorToRed (FILE ∗stream)
- void FMC_setBGStreamColorToWhite (FILE ∗stream)
- void FMC_setBGStreamColorToYellow (FILE ∗stream)
- void FMC_setTextStyleToBlink (FILE ∗stream)
- void FMC_setTextStyleToBold (FILE ∗stream)
- void FMC_setTextStyleToDim (FILE ∗stream)
- void FMC_setTextStyleToHidden (FILE ∗stream)
- void FMC_setTextStyleToReverse (FILE ∗stream)
- void FMC_setTextStyleToUnderlined (FILE ∗stream)

3.58.1 Macro Definition Documentation

3.58.1.1 FMC_ERRORS

```
#define FMC_ERRORS
```

Definition at line 30 of file FMC_errors.h.

3.58.1.2 FMC_makeMsg

```
#define FMC_makeMsg(
              err_var_name,
              argc,
              ... )
```

Value:

```
    char err_var_name[256] = {"\0"};                \
    FMC_makeMsg_f(err_var_name, argc, __VA_ARGS__)
```

Definition at line 38 of file FMC_errors.h.

3.58.2 Function Documentation

3.58.2.1 FMC_changeStreamTextColorToBlue()

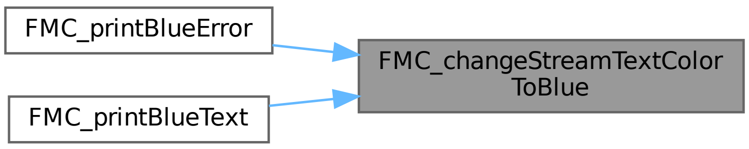
```
void FMC_changeStreamTextColorToBlue (
             FILE * stream )
```

Definition at line 63 of file FMC_errors.h.

References FG_BLUE.

Referenced by FMC_printBlueError(), and FMC_printBlueText().

Here is the caller graph for this function:



3.58.2.2 FMC_changeStreamTextColorToBrightBlue()

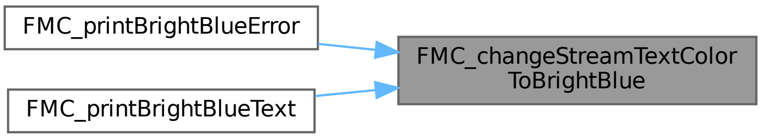
```
void FMC_changeStreamTextColorToBrightBlue (
            FILE * stream )
```

Definition at line 98 of file FMC_errors.h.

References FG_BRIGHT_BLUE.

Referenced by FMC_printBrightBlueError(), and FMC_printBrightBlueText().

Here is the caller graph for this function:



3.58.2.3 FMC_changeStreamTextColorToBrightCyan()

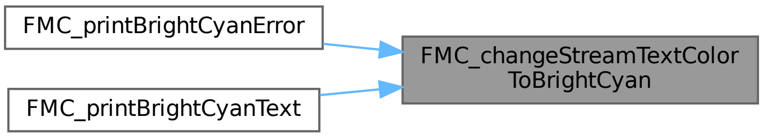
```
void FMC_changeStreamTextColorToBrightCyan (
            FILE * stream )
```

Definition at line 108 of file FMC_errors.h.

References FG_BRIGHT_CYAN.

Referenced by FMC_printBrightCyanError(), and FMC_printBrightCyanText().

Here is the caller graph for this function:



3.58.2.4 FMC_changeStreamTextColorToBrightGreen()

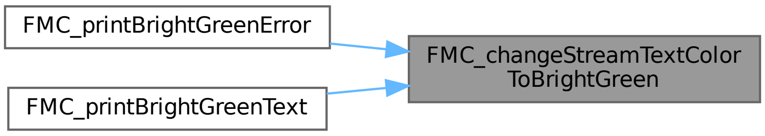
```
void FMC_changeStreamTextColorToBrightGreen (
            FILE * stream )
```

Definition at line 88 of file FMC_errors.h.

References FG_BRIGHT_GREEN.

Referenced by FMC_printBrightGreenError(), and FMC_printBrightGreenText().

Here is the caller graph for this function:


FMC_printBrightGreenError
FMC_printBrightGreenText
FMC_changeStreamTextColor ToBrightGreen

3.58.2.5 FMC_changeStreamTextColorToBrightMagenta()

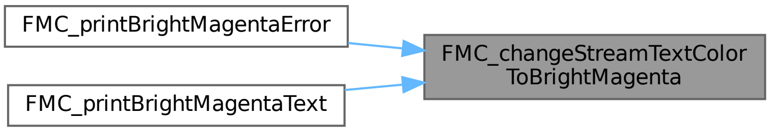
```
void FMC_changeStreamTextColorToBrightMagenta (
            FILE * stream )
```

Definition at line 103 of file FMC_errors.h.

References FG_BRIGHT_MAGENTA.

Referenced by FMC_printBrightMagentaError(), and FMC_printBrightMagentaText().

Here is the caller graph for this function:


FMC_printBrightMagentaError
FMC_printBrightMagentaText
FMC_changeStreamTextColor ToBrightMagenta

3.58.2.6 FMC_changeStreamTextColorToBrightRed()

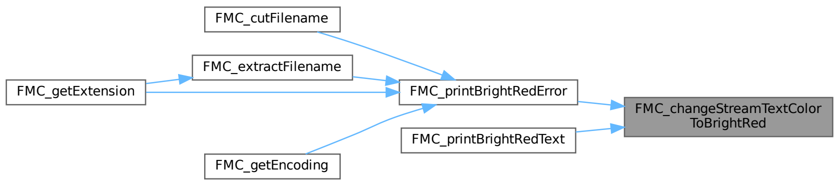
```
void FMC_changeStreamTextColorToBrightRed (
            FILE * stream )
```

Definition at line 83 of file FMC_errors.h.

References FG_BRIGHT_RED.

Referenced by FMC_printBrightRedError(), and FMC_printBrightRedText().

Here is the caller graph for this function:



3.58.2.7 FMC_changeStreamTextColorToBrightWhite()

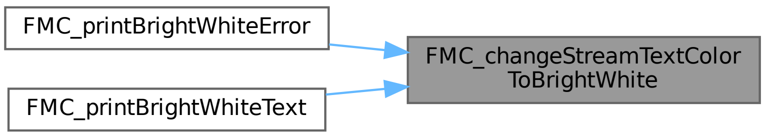
```
void FMC_changeStreamTextColorToBrightWhite (
            FILE * stream )
```

Definition at line 113 of file FMC_errors.h.

References FG_BRIGHT_WHITE.

Referenced by FMC_printBrightWhiteError(), and FMC_printBrightWhiteText().

Here is the caller graph for this function:


FMC_printBrightWhiteError
FMC_printBrightWhiteText
FMC_changeStreamTextColor ToBrightWhite

3.58.2.8 FMC_changeStreamTextColorToBrightYellow()

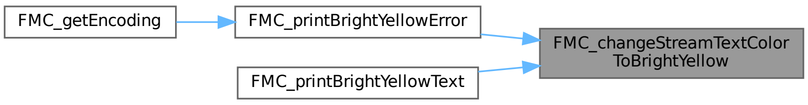
```
void FMC_changeStreamTextColorToBrightYellow (
            FILE * stream )
```

Definition at line 93 of file FMC_errors.h.

References FG_BRIGHT_YELLOW.

Referenced by FMC_printBrightYellowError(), and FMC_printBrightYellowText().

Here is the caller graph for this function:


FMC_getEncoding
FMC_printBrightYellowError
FMC_printBrightYellowText
FMC_changeStreamTextColor ToBrightYellow

3.58.2.9 FMC_changeStreamTextColorToCyan()

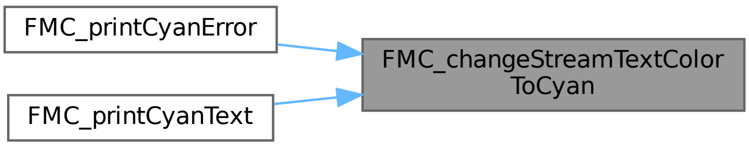
```
void FMC_changeStreamTextColorToCyan (
            FILE * stream )
```

Definition at line 73 of file FMC_errors.h.

References FG_CYAN.

Referenced by FMC_printCyanError(), and FMC_printCyanText().

Here is the caller graph for this function:



3.58.2.10 FMC_changeStreamTextColorToGreen()

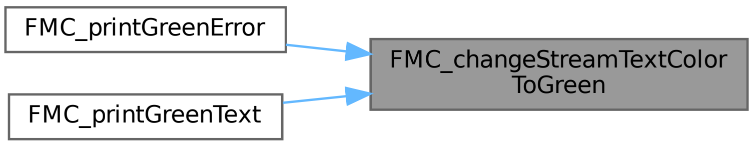
```
void FMC_changeStreamTextColorToGreen (
            FILE * stream )
```

Definition at line 53 of file FMC_errors.h.

References FG_GREEN.

Referenced by FMC_printGreenError(), and FMC_printGreenText().

Here is the caller graph for this function:



3.58.2.11 FMC_changeStreamTextColorToMagenta()

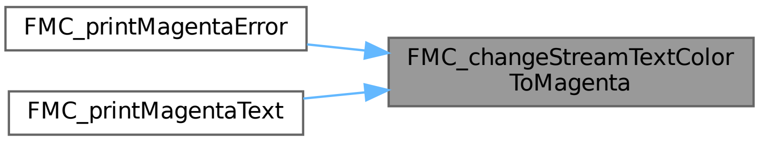
```
void FMC_changeStreamTextColorToMagenta (
            FILE * stream )
```

Definition at line 68 of file FMC_errors.h.

References FG_MAGENTA.

Referenced by FMC_printMagentaError(), and FMC_printMagentaText().

Here is the caller graph for this function:



3.58.2.12 FMC_changeStreamTextColorToRed()

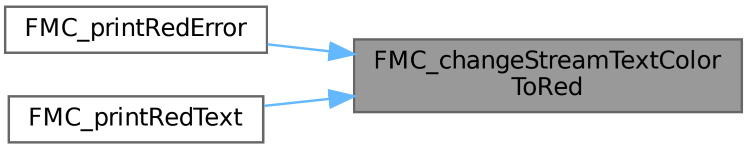
```
void FMC_changeStreamTextColorToRed (
             FILE * stream )
```

Definition at line 48 of file FMC_errors.h.

References FG_RED.

Referenced by FMC_printRedError(), and FMC_printRedText().

Here is the caller graph for this function:



3.58.2.13 FMC_changeStreamTextColorToWhite()

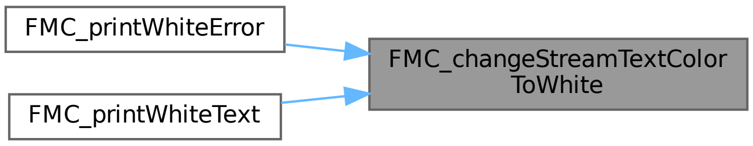
```
void FMC_changeStreamTextColorToWhite (
             FILE * stream )
```

Definition at line 78 of file FMC_errors.h.

References FG_WHITE.

Referenced by FMC_printWhiteError(), and FMC_printWhiteText().

Here is the caller graph for this function:



3.58.2.14 FMC_changeStreamTextColorToYellow()

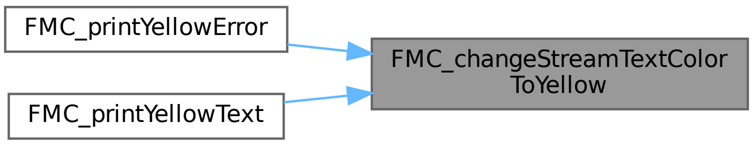
```
void FMC_changeStreamTextColorToYellow (
            FILE * stream )
```

Definition at line 58 of file FMC_errors.h.

References FG_YELLOW.

Referenced by FMC_printYellowError(), and FMC_printYellowText().

Here is the caller graph for this function:



3.58.2.15 FMC_makeMsg_f()

```
void FMC_makeMsg_f (
            char * buff,
            unsigned int argc,
             ... )
```

Definition at line 33 of file FMC_errors.c.

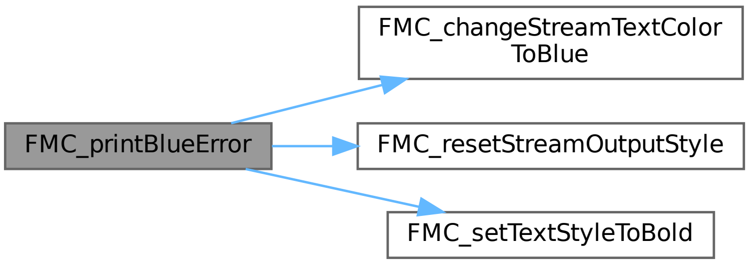
3.58.2.16 FMC_printBlueError()

```
void FMC_printBlueError (
            FILE * stream,
            const char * text )
```

Definition at line 341 of file FMC_errors.h.

References FMC_changeStreamTextColorToBlue(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:



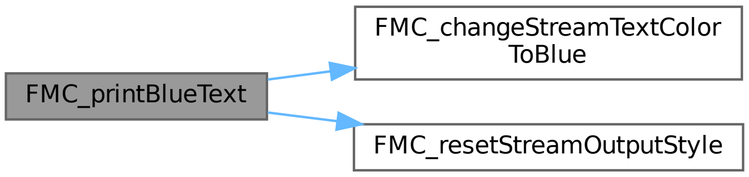
3.58.2.17 FMC_printBlueText()

```
void FMC_printBlueText (
            FILE * stream,
            const char * text )
```

Definition at line 240 of file FMC_errors.h.

References FMC_changeStreamTextColorToBlue(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:



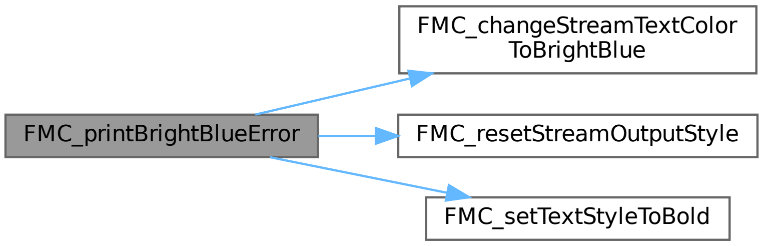
3.58.2.18 FMC_printBrightBlueError()

```
void FMC_printBrightBlueError (
             FILE * stream,
             const char * text )
```

Definition at line 397 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightBlue(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:


FMC_printBrightBlueError
FMC_changeStreamTextColor ToBrightBlue
FMC_resetStreamOutputStyle
FMC_setTextStyleToBold

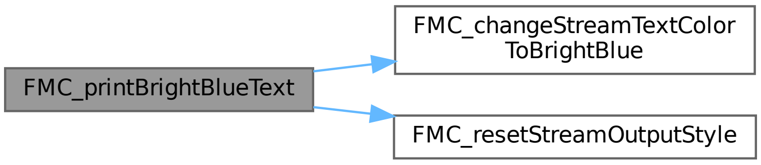
3.58.2.19 FMC_printBrightBlueText()

```
void FMC_printBrightBlueText (
             FILE * stream,
             const char * text )
```

Definition at line 289 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightBlue(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:


FMC_printBrightBlueText
FMC_changeStreamTextColor ToBrightBlue
FMC_resetStreamOutputStyle

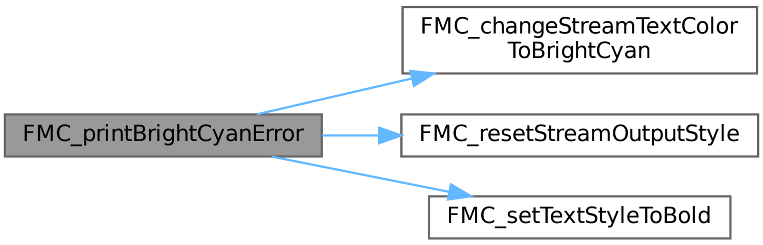
3.58.2.20 FMC_printBrightCyanError()

```
void FMC_printBrightCyanError (
             FILE * stream,
             const char * text )
```

Definition at line 413 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightCyan(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:


FMC_printBrightCyanError
FMC_changeStreamTextColor ToBrightCyan
FMC_resetStreamOutputStyle
FMC_setTextStyleToBold

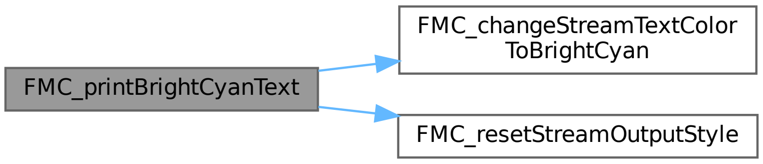
3.58.2.21 FMC_printBrightCyanText()

```
void FMC_printBrightCyanText (
             FILE * stream,
             const char * text )
```

Definition at line 303 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightCyan(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:


FMC_printBrightCyanText
FMC_changeStreamTextColor ToBrightCyan
FMC_resetStreamOutputStyle

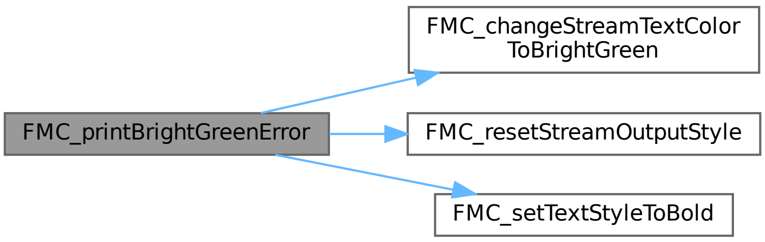
3.58.2.22 FMC_printBrightGreenError()

```
void FMC_printBrightGreenError (
             FILE * stream,
             const char * text )
```

Definition at line 381 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightGreen(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:


FMC_printBrightGreenError
FMC_changeStreamTextColor ToBrightGreen
FMC_resetStreamOutputStyle
FMC_setTextStyleToBold

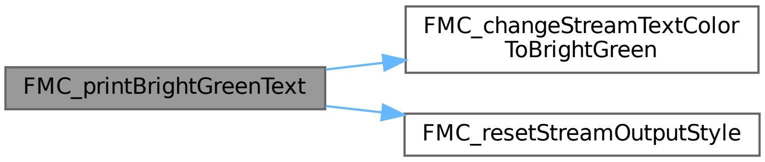
3.58.2.23 FMC_printBrightGreenText()

```
void FMC_printBrightGreenText (
             FILE * stream,
             const char * text )
```

Definition at line 275 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightGreen(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:


FMC_printBrightGreenText
FMC_changeStreamTextColor ToBrightGreen
FMC_resetStreamOutputStyle

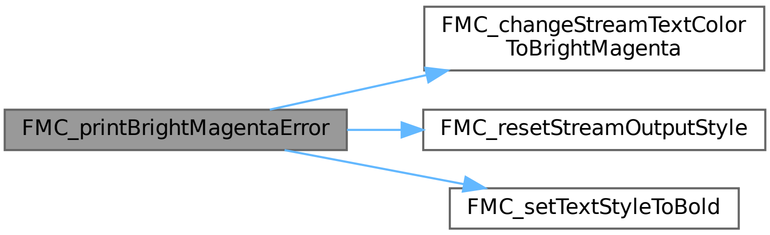
3.58.2.24 FMC_printBrightMagentaError()

```
void FMC_printBrightMagentaError (
             FILE * stream,
             const char * text )
```

Definition at line 405 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightMagenta(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:


FMC_changeStreamTextColor ToBrightMagenta
FMC_printBrightMagentaError
FMC_resetStreamOutputStyle
FMC_setTextStyleToBold

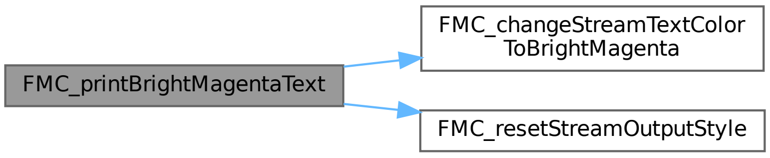
3.58.2.25 FMC_printBrightMagentaText()

```
void FMC_printBrightMagentaText (
             FILE * stream,
             const char * text )
```

Definition at line 296 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightMagenta(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:


FMC_changeStreamTextColor ToBrightMagenta
FMC_printBrightMagentaText
FMC_resetStreamOutputStyle

3.58.2.26 FMC_printBrightRedError()

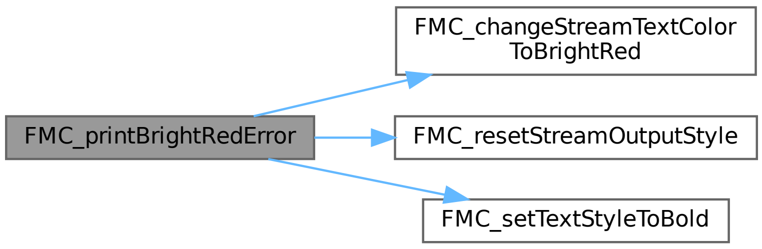
```
void FMC_printBrightRedError (
            FILE * stream,
            const char * text )
```

Definition at line 373 of file FMC_errors.h.

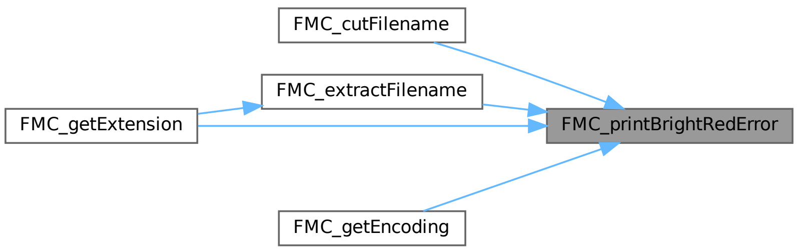
References FMC_changeStreamTextColorToBrightRed(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Referenced by FMC_cutFilename(), FMC_extractFilename(), FMC_getEncoding(), and FMC_getExtension().

Here is the call graph for this function:


FMC_changeStreamTextColor ToBrightRed
FMC_printBrightRedError
FMC_resetStreamOutputStyle
FMC_setTextStyleToBold

Here is the caller graph for this function:



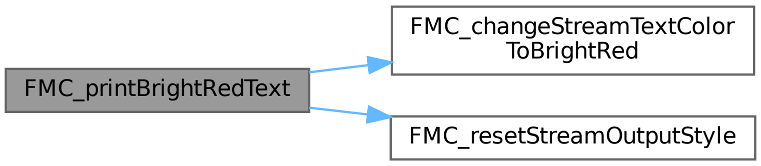
3.58.2.27 FMC_printBrightRedText()

```
void FMC_printBrightRedText (
            FILE * stream,
            const char * text )
```

Definition at line 268 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightRed(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:


FMC_printBrightRedText
FMC_changeStreamTextColor ToBrightRed
FMC_resetStreamOutputStyle

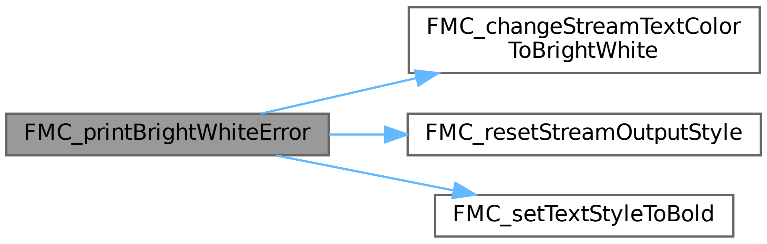
3.58.2.28 FMC_printBrightWhiteError()

```
void FMC_printBrightWhiteError (
             FILE * stream,
             const char * text )
```

Definition at line 421 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightWhite(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:


FMC_printBrightWhiteError
FMC_changeStreamTextColor ToBrightWhite
FMC_resetStreamOutputStyle
FMC_setTextStyleToBold

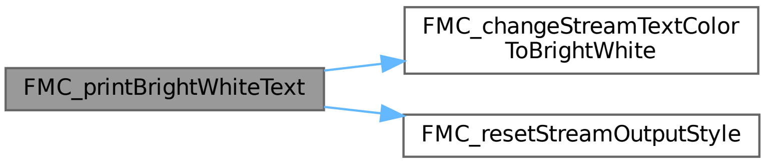
3.58.2.29 FMC_printBrightWhiteText()

```
void FMC_printBrightWhiteText (
            FILE * stream,
            const char * text )
```

Definition at line 310 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightWhite(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:


FMC_printBrightWhiteText
FMC_changeStreamTextColor ToBrightWhite
FMC_resetStreamOutputStyle

3.58.2.30 FMC_printBrightYellowError()

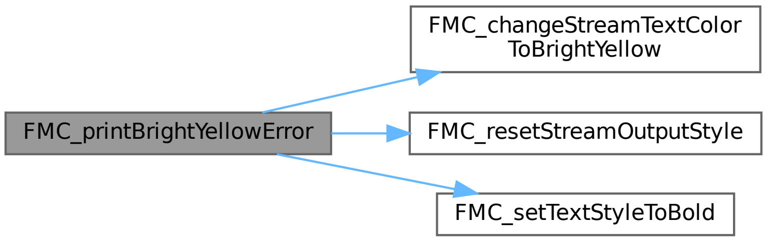
```
void FMC_printBrightYellowError (
            FILE * stream,
            const char * text )
```

Definition at line 389 of file FMC_errors.h.

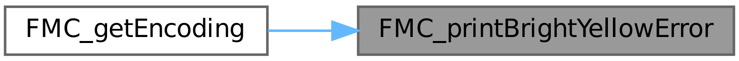
References FMC_changeStreamTextColorToBrightYellow(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Referenced by FMC_getEncoding().

Here is the call graph for this function:


FMC_printBrightYellowError
FMC_changeStreamTextColor ToBrightYellow
FMC_resetStreamOutputStyle
FMC_setTextStyleToBold

Here is the caller graph for this function:



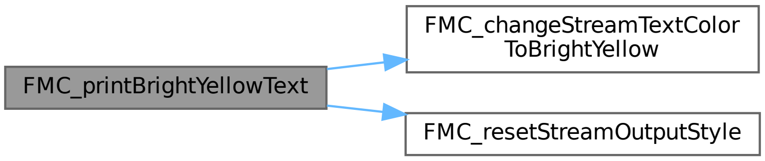
3.58.2.31 FMC_printBrightYellowText()

```
void FMC_printBrightYellowText (
            FILE * stream,
            const char * text )
```

Definition at line 282 of file FMC_errors.h.

References FMC_changeStreamTextColorToBrightYellow(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:


FMC_printBrightYellowText
FMC_changeStreamTextColor ToBrightYellow
FMC_resetStreamOutputStyle

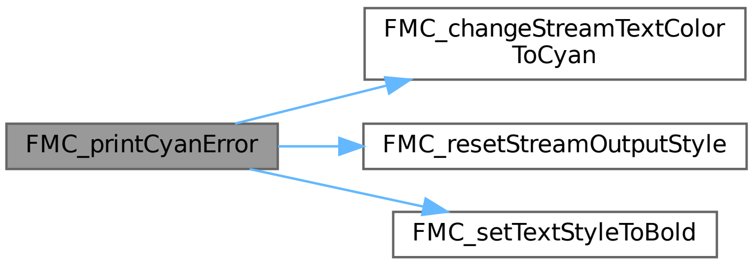
3.58.2.32 FMC_printCyanError()

```
void FMC_printCyanError (
            FILE * stream,
            const char * text )
```

Definition at line 357 of file FMC_errors.h.

References FMC_changeStreamTextColorToCyan(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:



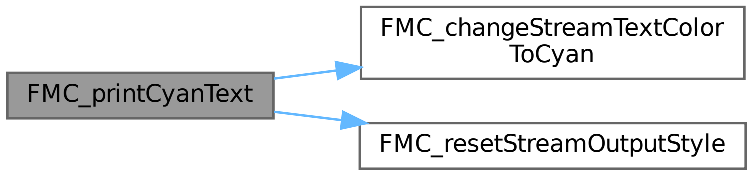
3.58.2.33 FMC_printCyanText()

```
void FMC_printCyanText (
            FILE * stream,
            const char * text )
```

Definition at line 254 of file FMC_errors.h.

References FMC_changeStreamTextColorToCyan(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:



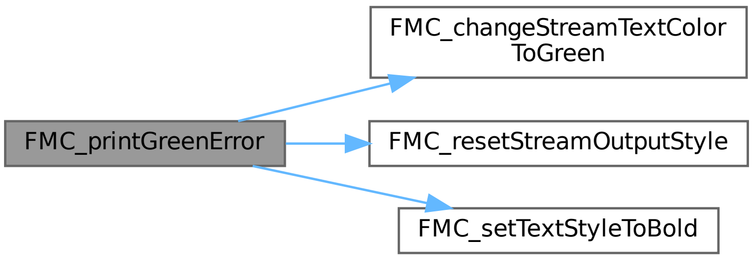
3.58.2.34 FMC_printGreenError()

```
void FMC_printGreenError (
             FILE * stream,
             const char * text )
```

Definition at line 325 of file FMC_errors.h.

References FMC_changeStreamTextColorToGreen(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:



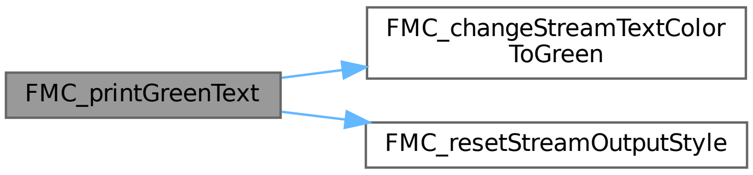
3.58.2.35 FMC_printGreenText()

```
void FMC_printGreenText (
             FILE * stream,
             const char * text )
```

Definition at line 226 of file FMC_errors.h.

References FMC_changeStreamTextColorToGreen(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:



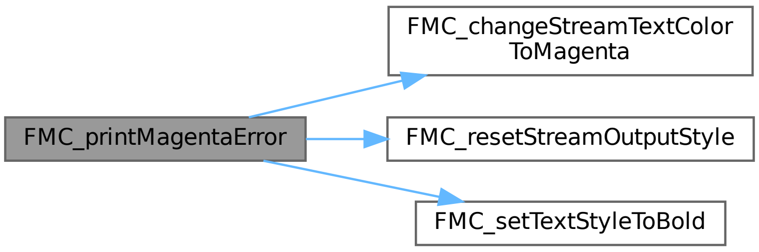
3.58.2.36 FMC_printMagentaError()

```
void FMC_printMagentaError (
            FILE * stream,
            const char * text )
```

Definition at line 349 of file FMC_errors.h.

References FMC_changeStreamTextColorToMagenta(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:


FMC_printMagentaError
FMC_changeStreamTextColor ToMagenta
FMC_resetStreamOutputStyle
FMC_setTextStyleToBold

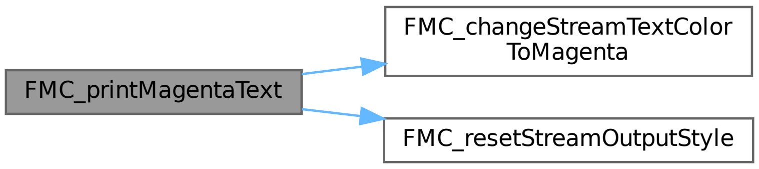
3.58.2.37 FMC_printMagentaText()

```
void FMC_printMagentaText (
            FILE * stream,
            const char * text )
```

Definition at line 247 of file FMC_errors.h.

References FMC_changeStreamTextColorToMagenta(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:


FMC_printMagentaText
FMC_changeStreamTextColor ToMagenta
FMC_resetStreamOutputStyle

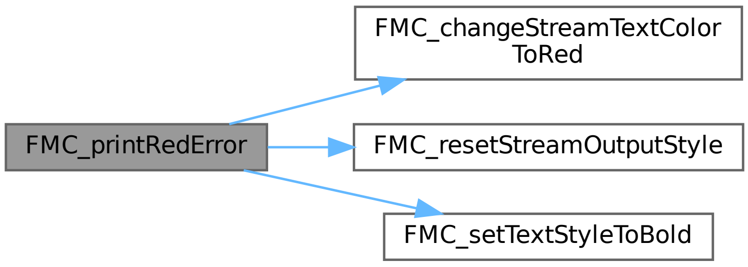
3.58.2.38 FMC_printRedError()

```
void FMC_printRedError (
            FILE * stream,
            const char * text )
```

Definition at line 317 of file FMC_errors.h.

References FMC_changeStreamTextColorToRed(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:



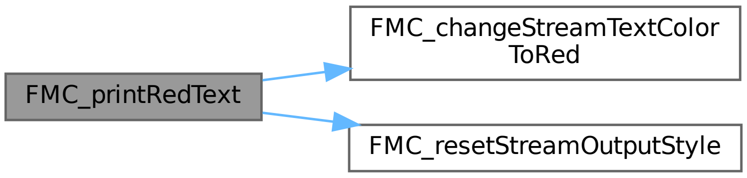
3.58.2.39 FMC_printRedText()

```
void FMC_printRedText (
            FILE * stream,
            const char * text )
```

Definition at line 219 of file FMC_errors.h.

References FMC_changeStreamTextColorToRed(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:



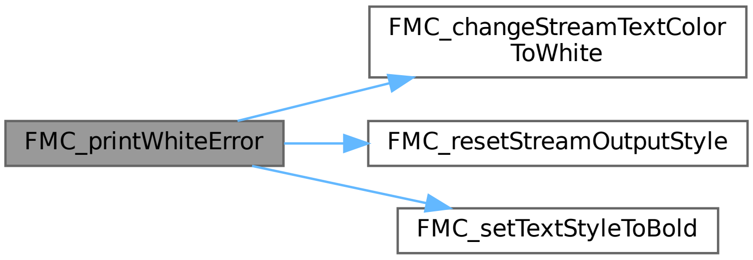
3.58.2.40 FMC_printWhiteError()

```
void FMC_printWhiteError (
            FILE * stream,
            const char * text )
```

Definition at line 365 of file FMC_errors.h.

References FMC_changeStreamTextColorToWhite(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:



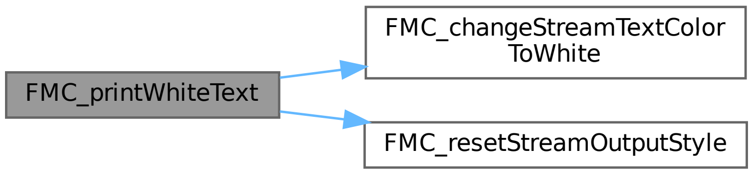
3.58.2.41 FMC_printWhiteText()

```
void FMC_printWhiteText (
            FILE * stream,
            const char * text )
```

Definition at line 261 of file FMC_errors.h.

References FMC_changeStreamTextColorToWhite(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:



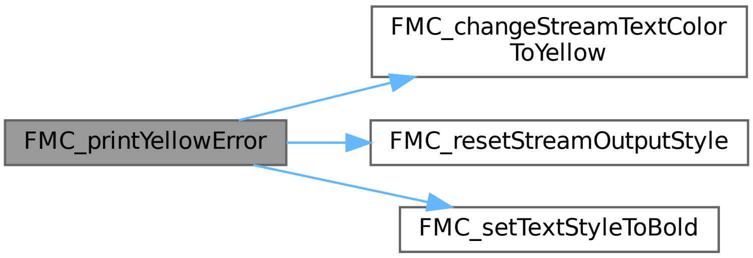
3.58.2.42 FMC_printYellowError()

```
void FMC_printYellowError (
             FILE * stream,
             const char * text )
```

Definition at line 333 of file FMC_errors.h.

References FMC_changeStreamTextColorToYellow(), FMC_resetStreamOutputStyle(), and FMC_setTextStyleToBold().

Here is the call graph for this function:


FMC_printYellowError
FMC_changeStreamTextColor ToYellow
FMC_resetStreamOutputStyle
FMC_setTextStyleToBold

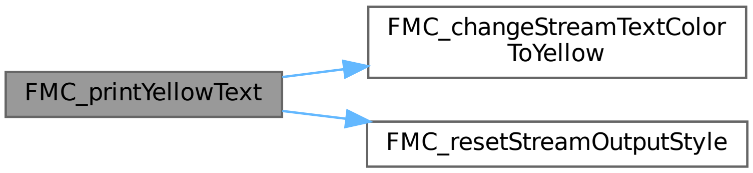
3.58.2.43 FMC_printYellowText()

```
void FMC_printYellowText (
             FILE * stream,
             const char * text )
```

Definition at line 233 of file FMC_errors.h.

References FMC_changeStreamTextColorToYellow(), and FMC_resetStreamOutputStyle().

Here is the call graph for this function:


FMC_printYellowText
FMC_changeStreamTextColor ToYellow
FMC_resetStreamOutputStyle

3.58.2.44 FMC_resetStreamOutputStyle()

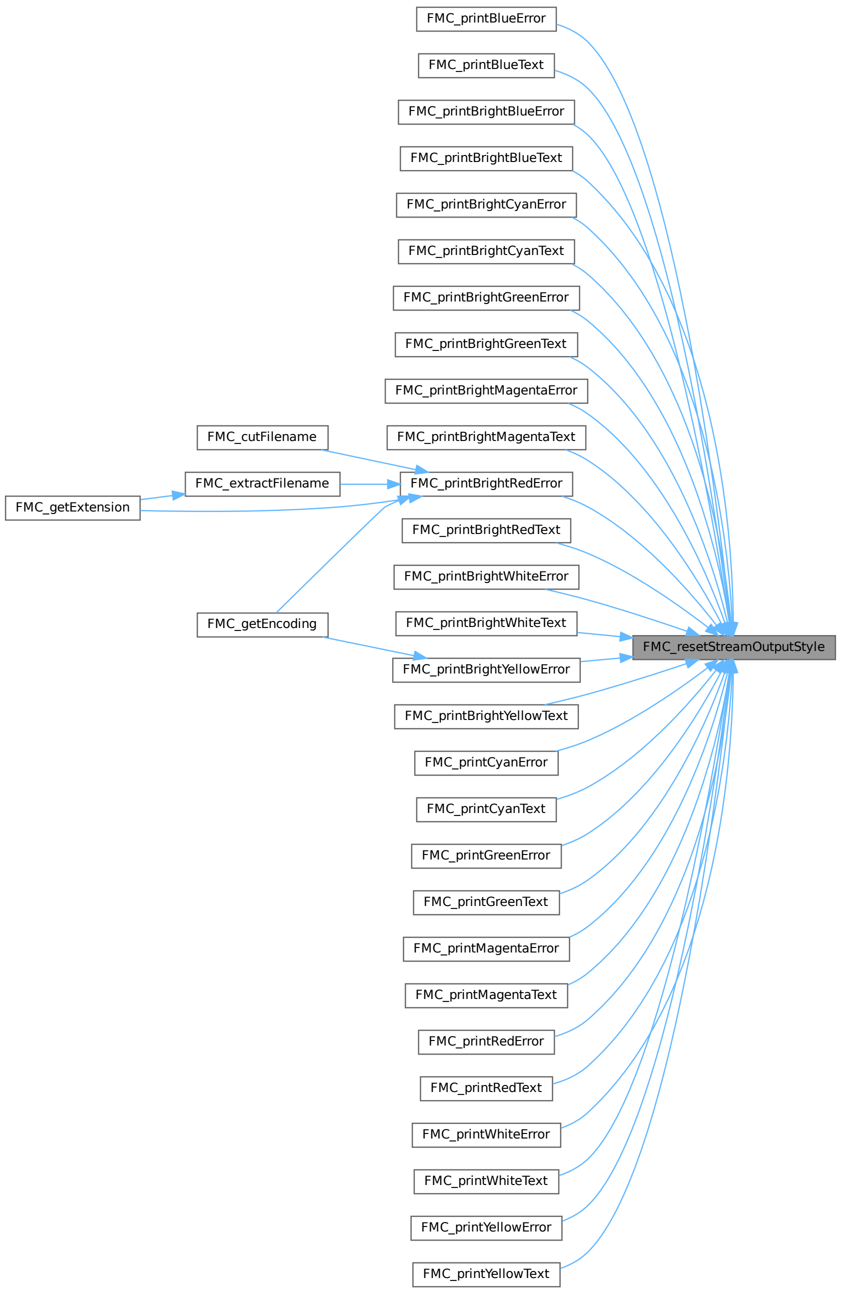
```
void FMC_resetStreamOutputStyle (
            FILE * stream )
```

Definition at line 42 of file FMC_errors.h.

References RESET.

Referenced by FMC_printBlueError(), FMC_printBlueText(), FMC_printBrightBlueError(), FMC_printBrightBlueText(), FMC_printBrightCyanError(), FMC_printBrightCyanText(), FMC_printBrightGreenError(), FMC_printBrightGreenText(), FMC_printBrightMagentaError(), FMC_printBrightMagentaText(), FMC_printBrightRedError(), FMC_printBrightRedText(), FMC_printBrightWhiteError(), FMC_printBrightWhiteText(), FMC_printBrightYellowError(), FMC_printBrightYellowText(), FMC_printCyanError(), FMC_printCyanText(), FMC_printGreenError(), FMC_printGreenText(), FMC_printMagentaError(), FMC_printMagentaText(), FMC_printRedError(), FMC_printRedText(), FMC_printWhiteError(), FMC_printWhiteText(), FMC_printYellowError(), and FMC_printYellowText().

Here is the caller graph for this function:



3.58.2.45 FMC_setBGStreamColorToBlue()

```
void FMC_setBGStreamColorToBlue (
            FILE * stream )
```

Definition at line 133 of file FMC_errors.h.

References BG_BLUE.

3.58.2.46 FMC_setBGStreamColorToBrightBlue()

```
void FMC_setBGStreamColorToBrightBlue (
            FILE * stream )
```

Definition at line 168 of file FMC_errors.h.

References BG_BRIGHT_BLUE.

3.58.2.47 FMC_setBGStreamColorToBrightCyan()

```
void FMC_setBGStreamColorToBrightCyan (
            FILE * stream )
```

Definition at line 178 of file FMC_errors.h.

References BG_BRIGHT_CYAN.

3.58.2.48 FMC_setBGStreamColorToBrightGreen()

```
void FMC_setBGStreamColorToBrightGreen (
            FILE * stream )
```

Definition at line 158 of file FMC_errors.h.

References BG_BRIGHT_GREEN.

3.58.2.49 FMC_setBGStreamColorToBrightMagenta()

```
void FMC_setBGStreamColorToBrightMagenta (
            FILE * stream )
```

Definition at line 173 of file FMC_errors.h.

References BG_BRIGHT_MAGENTA.

3.58.2.50 FMC_setBGStreamColorToBrightRed()

```
void FMC_setBGStreamColorToBrightRed (
            FILE * stream )
```

Definition at line 153 of file FMC_errors.h.

References BG_BRIGHT_RED.

3.58.2.51 FMC_setBGStreamColorToBrightWhite()

```
void FMC_setBGStreamColorToBrightWhite (
            FILE * stream )
```

Definition at line 183 of file FMC_errors.h.

References BG_BRIGHT_WHITE.

3.58.2.52 FMC_setBGStreamColorToBrightYellow()

```
void FMC_setBGStreamColorToBrightYellow (
            FILE * stream )
```

Definition at line 163 of file FMC_errors.h.

References BG_BRIGHT_YELLOW.

3.58.2.53 FMC_setBGStreamColorToCyan()

```
void FMC_setBGStreamColorToCyan (
            FILE * stream )
```

Definition at line 143 of file FMC_errors.h.

References BG_CYAN.

3.58.2.54 FMC_setBGStreamColorToGreen()

```
void FMC_setBGStreamColorToGreen (
            FILE * stream )
```

Definition at line 123 of file FMC_errors.h.

References BG_GREEN.

3.58.2.55 FMC_setBGStreamColorToMagenta()

```
void FMC_setBGStreamColorToMagenta (
            FILE * stream )
```

Definition at line 138 of file FMC_errors.h.

References BG_MAGENTA.

3.58.2.56 FMC_setBGStreamColorToRed()

```
void FMC_setBGStreamColorToRed (
             FILE * stream )
```

Definition at line 118 of file FMC_errors.h.

References BG_RED.

3.58.2.57 FMC_setBGStreamColorToWhite()

```
void FMC_setBGStreamColorToWhite (
             FILE * stream )
```

Definition at line 148 of file FMC_errors.h.

References BG_WHITE.

3.58.2.58 FMC_setBGStreamColorToYellow()

```
void FMC_setBGStreamColorToYellow (
             FILE * stream )
```

Definition at line 128 of file FMC_errors.h.

References BG_YELLOW.

3.58.2.59 FMC_setTextStyleToBlink()

```
void FMC_setTextStyleToBlink (
             FILE * stream )
```

Definition at line 203 of file FMC_errors.h.

References TXT_BLINK.

3.58.2.60 FMC_setTextStyleToBold()

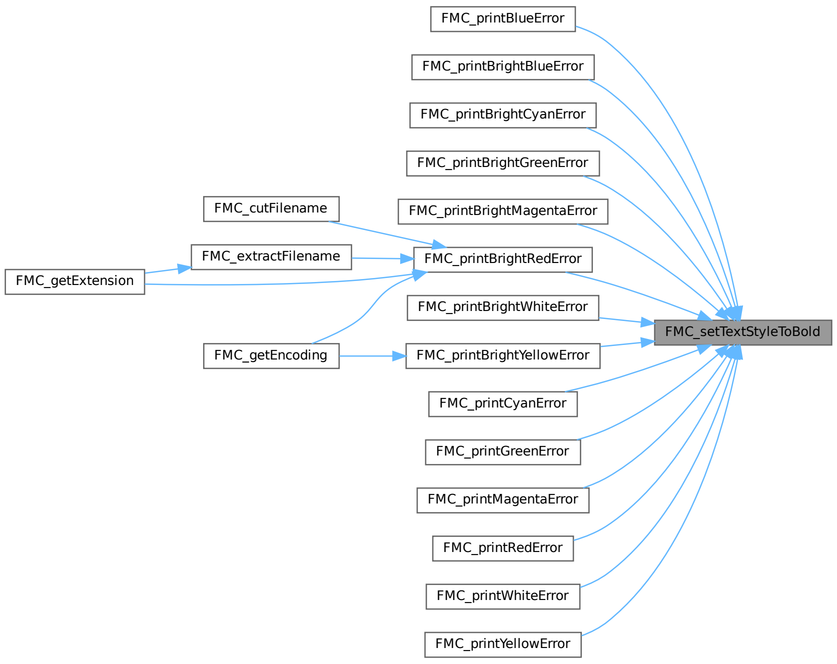
```
void FMC_setTextStyleToBold (
             FILE * stream )
```

Definition at line 188 of file FMC_errors.h.

References TXT_BOLD.

Referenced by FMC_printBlueError(), FMC_printBrightBlueError(), FMC_printBrightCyanError(), FMC_printBrightGreenError(), FMC_printBrightMagentaError(), FMC_printBrightRedError(), FMC_printBrightWhiteError(), FMC_printBrightYellowError(), FMC_printCyanError(), FMC_printGreenError(), FMC_printMagentaError(), FMC_printRedError(), FMC_printWhiteError(), and FMC_printYellowError().

Here is the caller graph for this function:



3.58.2.61 FMC_setTextStyleToDim()

```
void FMC_setTextStyleToDim (
             FILE * stream )
```

Definition at line 193 of file FMC_errors.h.

References TXT_DIM.

3.58.2.62 FMC_setTextStyleToHidden()

```
void FMC_setTextStyleToHidden (
            FILE * stream )
```

Definition at line 213 of file FMC_errors.h.

References TXT_HIDDEN.

3.58.2.63 FMC_setTextStyleToReverse()

```
void FMC_setTextStyleToReverse (
            FILE * stream )
```

Definition at line 208 of file FMC_errors.h.

References TXT_REVERSE.

3.58.2.64 FMC_setTextStyleToUnderlined()

```
void FMC_setTextStyleToUnderlined (
            FILE * stream )
```

Definition at line 198 of file FMC_errors.h.

References TXT_UNDERLINED.

3.59 FMC_errors.h

Go to the documentation of this file.

```
/*

MIT License

Copyright (c) 2022 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/

#pragma once
```

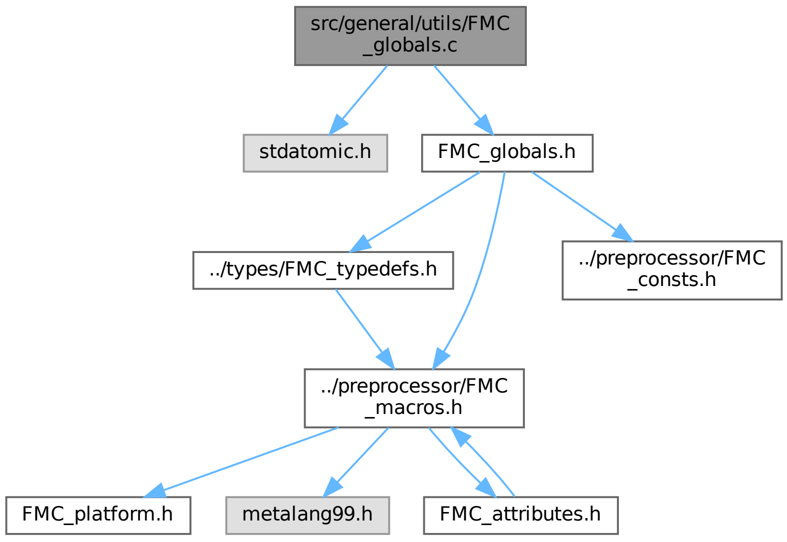
```c

#ifndef FMC_ERRORS
#define FMC_ERRORS

#include <stdio.h>
#include "../preprocessor/FMC_consts.h"
#include "../preprocessor/FMC_macros.h"

FMC_SHARED FMC_FUNC_NONNULL(1) void FMC_makeMsg_f(char *buff, unsigned int argc, ...);

#define FMC_makeMsg(err_var_name, argc, ...)            \
    char err_var_name[256] = {"\0"};                    \
    FMC_makeMsg_f(err_var_name, argc, __VA_ARGS__)

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_resetStreamOutputStyle(FILE *stream)
{
    fprintf(stream, RESET);
}


FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToRed(FILE *stream)
{
    fprintf(stream, FG_RED);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToGreen(FILE *stream)
{
    fprintf(stream, FG_GREEN);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToYellow(FILE *stream)
{
    fprintf(stream, FG_YELLOW);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToBlue(FILE *stream)
{
    fprintf(stream, FG_BLUE);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToMagenta(FILE *stream)
{
    fprintf(stream, FG_MAGENTA);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToCyan(FILE *stream)
{
    fprintf(stream, FG_CYAN);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToWhite(FILE *stream)
{
    fprintf(stream, FG_WHITE);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToBrightRed(FILE *stream)
{
    fprintf(stream, FG_BRIGHT_RED);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToBrightGreen(FILE *stream)
{
    fprintf(stream, FG_BRIGHT_GREEN);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToBrightYellow(FILE *stream)
{
    fprintf(stream, FG_BRIGHT_YELLOW);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToBrightBlue(FILE *stream)
{
    fprintf(stream, FG_BRIGHT_BLUE);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToBrightMagenta(FILE *stream)
{
    fprintf(stream, FG_BRIGHT_MAGENTA);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToBrightCyan(FILE *stream)
{
    fprintf(stream, FG_BRIGHT_CYAN);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_changeStreamTextColorToBrightWhite(FILE *stream)
{
    fprintf(stream, FG_BRIGHT_WHITE);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_setBGStreamColorToRed(FILE *stream)
{
    fprintf(stream, BG_RED);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_setBGStreamColorToGreen(FILE *stream)
{
    fprintf(stream, BG_GREEN);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_setBGStreamColorToYellow(FILE *stream)
{
    fprintf(stream, BG_YELLOW);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_setBGStreamColorToBlue(FILE *stream)
{
    fprintf(stream, BG_BLUE);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_setBGStreamColorToMagenta(FILE *stream)
{
    fprintf(stream, BG_MAGENTA);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_setBGStreamColorToCyan(FILE *stream)
{
    fprintf(stream, BG_CYAN);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_setBGStreamColorToWhite(FILE *stream)
{
    fprintf(stream, BG_WHITE);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_setBGStreamColorToBrightRed(FILE *stream)
{
    fprintf(stream, BG_BRIGHT_RED);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_setBGStreamColorToBrightGreen(FILE *stream)
{
    fprintf(stream, BG_BRIGHT_GREEN);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_setBGStreamColorToBrightYellow(FILE *stream)
{
    fprintf(stream, BG_BRIGHT_YELLOW);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_setBGStreamColorToBrightBlue(FILE *stream)
{
    fprintf(stream, BG_BRIGHT_BLUE);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_setBGStreamColorToBrightMagenta(FILE *stream)
{
    fprintf(stream, BG_BRIGHT_MAGENTA);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_setBGStreamColorToBrightCyan(FILE *stream)
{
    fprintf(stream, BG_BRIGHT_CYAN);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_setBGStreamColorToBrightWhite(FILE *stream)
{
    fprintf(stream, BG_BRIGHT_WHITE);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_setTextStyleToBold(FILE *stream)
{
    fprintf(stream, TXT_BOLD);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_setTextStyleToDim(FILE *stream)
{
    fprintf(stream, TXT_DIM);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_setTextStyleToUnderlined(FILE *stream)
{
    fprintf(stream, TXT_UNDERLINED);
}
```

```
FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_setTextStyleToBlink(FILE *stream)
{
    fprintf(stream, TXT_BLINK);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_setTextStyleToReverse(FILE *stream)
{
    fprintf(stream, TXT_REVERSE);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_setTextStyleToHidden(FILE *stream)
{
    fprintf(stream, TXT_HIDDEN);
}


FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printRedText(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToRed(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printGreenText(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToGreen(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printYellowText(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToYellow(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBlueText(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToBlue(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printMagentaText(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToMagenta(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printCyanText(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToCyan(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printWhiteText(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToWhite(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightRedText(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToBrightRed(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightGreenText(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToBrightGreen(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightYellowText(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToBrightYellow(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

```

```
FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightBlueText(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToBrightBlue(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightMagentaText(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToBrightMagenta(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightCyanText(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToBrightCyan(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightWhiteText(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToBrightWhite(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printRedError(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToRed(stream);
    FMC_setTextStyleToBold(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printGreenError(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToGreen(stream);
    FMC_setTextStyleToBold(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printYellowError(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToYellow(stream);
    FMC_setTextStyleToBold(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBlueError(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToBlue(stream);
    FMC_setTextStyleToBold(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printMagentaError(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToMagenta(stream);
    FMC_setTextStyleToBold(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printCyanError(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToCyan(stream);
    FMC_setTextStyleToBold(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printWhiteError(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToWhite(stream);
    FMC_setTextStyleToBold(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightRedError(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToBrightRed(stream);
    FMC_setTextStyleToBold(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightGreenError(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToBrightGreen(stream);
    FMC_setTextStyleToBold(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightYellowError(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToBrightYellow(stream);
    FMC_setTextStyleToBold(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightBlueError(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToBrightBlue(stream);
    FMC_setTextStyleToBold(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightMagentaError(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToBrightMagenta(stream);
    FMC_setTextStyleToBold(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightCyanError(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToBrightCyan(stream);
    FMC_setTextStyleToBold(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

FMC_FUNC_FLATTEN FMC_FUNC_INLINE void FMC_printBrightWhiteError(FILE *stream, const char *text)
{
    FMC_changeStreamTextColorToBrightWhite(stream);
    FMC_setTextStyleToBold(stream);
    fprintf(stream, "%s\n", text);
    FMC_resetStreamOutputStyle(stream);
}

#endif // FMC_ERRORS
```

3.60 src/general/utils/FMC_globals.c File Reference

Include dependency graph for FMC_globals.c:



Functions

- static volatile _Atomic (FMC_Bool)
- FMC_Bool FMC_getDebugState (void)

3.60.1 Function Documentation

3.60.1.1 _Atomic()

```
static volatile _Atomic (
                FMC_Bool  )   [static]
```

Definition at line 5 of file FMC_globals.c.

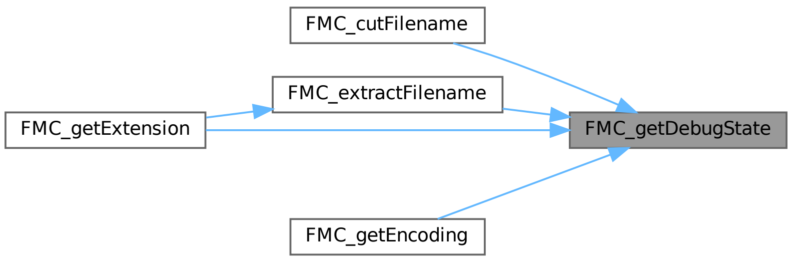
3.60.1.2 FMC_getDebugState()

```
FMC_Bool FMC_getDebugState (
              void  )
```

Definition at line 20 of file FMC_globals.c.

Referenced by FMC_cutFilename(), FMC_extractFilename(), FMC_getEncoding(), and FMC_getExtension().

Here is the caller graph for this function:



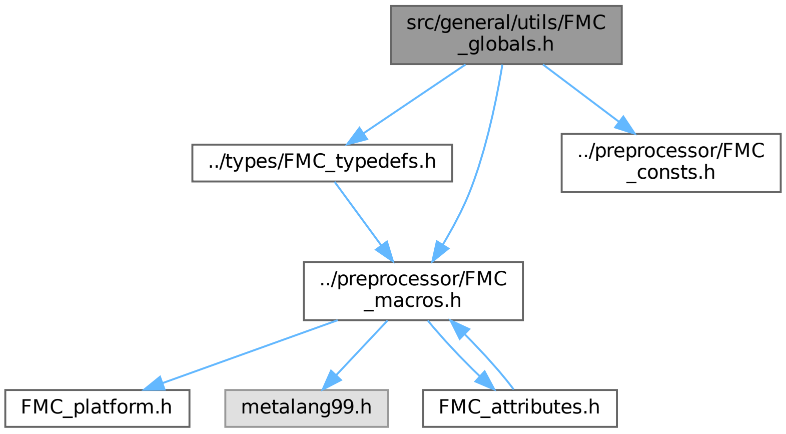
3.61 FMC_globals.c

Go to the documentation of this file.

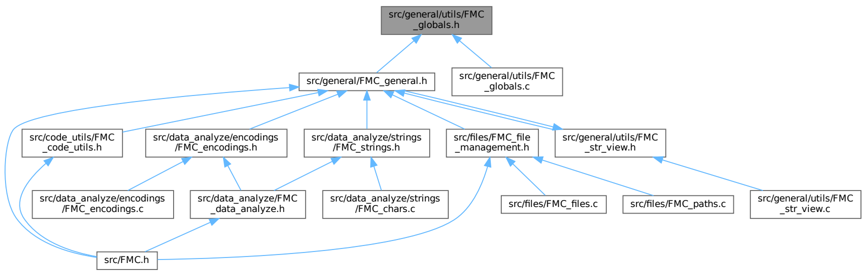
```
#include <stdatomic.h>
#include "FMC_globals.h"

#ifndef __STDC_NO_ATOMICS__
FMC_SHARED static volatile _Atomic(FMC_Bool) FMC_ENABLE_DEBUG FMC_VAR_COMMON;
#else
FMC_SHARED static volatile FMC_Bool FMC_ENABLE_DEBUG FMC_VAR_COMMON;
#endif

FMC_SHARED FMC_FUNC_COLD FMC_Bool FMC_setDebugState(FMC_Bool state)
{
    #ifndef __STDC_NO_ATOMICS__
    atomic_store(&FMC_ENABLE_DEBUG, state);
    #else
    FMC_ENABLE_DEBUG = state;
    #endif
    return FMC_ENABLE_DEBUG == state;
}

FMC_SHARED FMC_FUNC_HOT FMC_Bool FMC_getDebugState(void)
{
    #ifndef __STDC_NO_ATOMICS__
    return atomic_load(&FMC_ENABLE_DEBUG);
    #else
    return FMC_ENABLE_DEBUG;
    #endif
}
```

3.62 src/general/utils/FMC_globals.h File Reference

Include dependency graph for FMC_globals.h:



This graph shows which files directly or indirectly include this file:



Functions

- FMC_Bool FMC_getDebugState (void)
- FMC_FUNC_COLD FMC_Bool FMC_setDebugState (FMC_Bool state)

3.62.1 Function Documentation

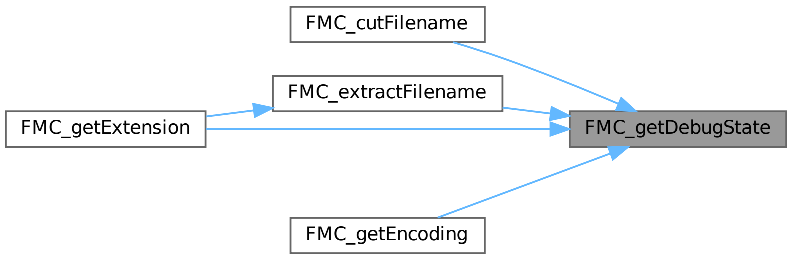
3.62.1.1 FMC_getDebugState()

```
FMC_Bool FMC_getDebugState (
            void  )
```

Definition at line 20 of file FMC_globals.c.

Referenced by FMC_cutFilename(), FMC_extractFilename(), FMC_getEncoding(), and FMC_getExtension().

Here is the caller graph for this function:



3.62.1.2 FMC_setDebugState()

```
FMC_FUNC_COLD FMC_Bool FMC_setDebugState (
            FMC_Bool state )
```

3.63 FMC_globals.h

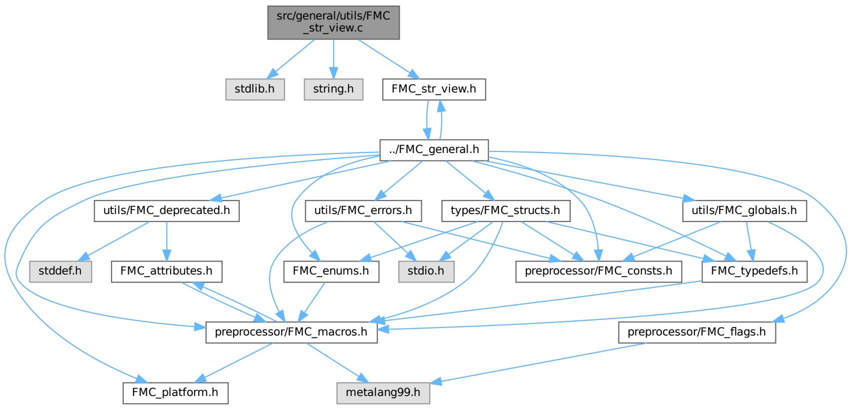
Go to the documentation of this file.

```
/*

MIT License

Copyright (c) 2022-2023 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/

```

```
#ifndef FMC_GLOBALS_H
#define FMC_GLOBALS_H

#include "../types/FMC_typedefs.h"
#include "../preprocessor/FMC_macros.h"
#include "../preprocessor/FMC_consts.h"

FMC_SHARED FMC_FUNC_COLD FMC_Bool FMC_setDebugState(FMC_Bool state);
FMC_SHARED FMC_FUNC_HOT FMC_Bool FMC_getDebugState(void);

#endif // FMC_GLOBALS_H
```

3.64 src/general/utils/FMC_str_view.c File Reference

Include dependency graph for FMC_str_view.c:



Functions

- void FMC_freeStrView (FMC_CStrView ∗view)
- FMC_FUNC_MALLOC (FMC_freeStrView, 1)

3.64.1 Function Documentation

3.64.1.1 FMC_freeStrView()

```
void FMC_freeStrView (
            FMC_CStrView ∗ view )
```

Definition at line 49 of file FMC_str_view.c.

References FManC_CStrView::str.

3.64.1.2 FMC_FUNC_MALLOC()

```
FMC_FUNC_MALLOC (
            FMC_freeStrView ,
            1  )
```

Definition at line 31 of file FMC_str_view.c.

References len, FManC_CStrView::size, and FManC_CStrView::str.

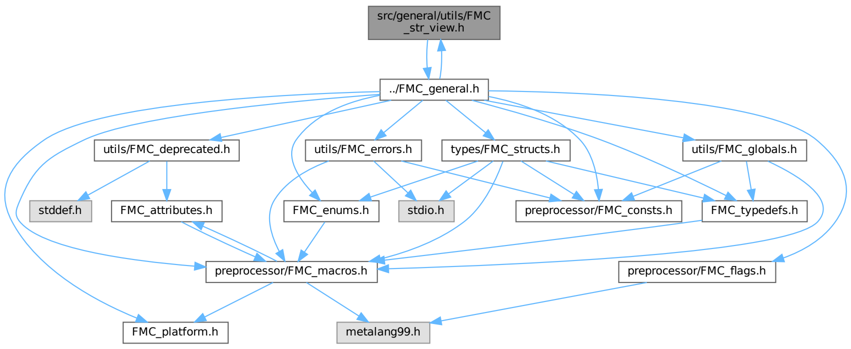
3.65 FMC_str_view.c

Go to the documentation of this file.

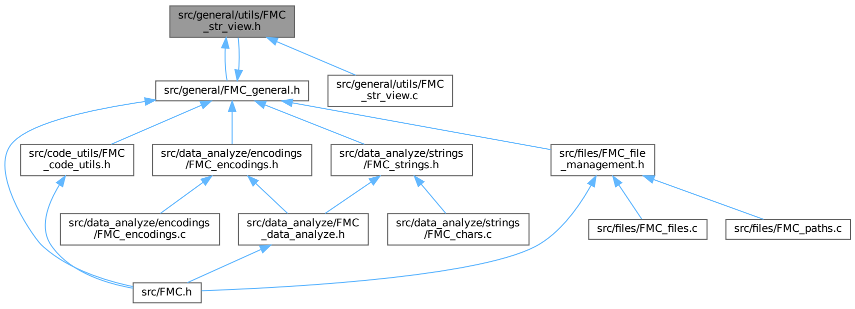
```
00001 /*
00002 
00003 MIT License
00004 
00005 Copyright (c) 2022-2023 Axel PASCON
00006 
00007 Permission is hereby granted, free of charge, to any person obtaining a copy
00008 of this software and associated documentation files (the "Software"), to deal
00009 in the Software without restriction, including without limitation the rights
00010 to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
00011 copies of the Software, and to permit persons to whom the Software is
00012 furnished to do so, subject to the following conditions:
00013 
00014 The above copyright notice and this permission notice shall be included in all
00015 copies or substantial portions of the Software.
00016 
00017 THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
00018 IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
00019 FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
00020 AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
00021 LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
00022 OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
00023 SOFTWARE.
00024 
00025 */
00026 
00027 #include <stdlib.h>
00028 #include <string.h>
00029 #include "FMC_str_view.h"
00030 
00031 FMC_SHARED FMC_FUNC_MALLOC(FMC_freeStrView, 1) FMC_CStrView* FMC_allocStrView(const char* const str,
      size_t len)
00032 {
00033     FMC_CStrView* view = malloc(sizeof(FMC_CStrView));
00034     if (view == NULL)
00035     {
00036         return NULL;
00037     }
00038     view->size = len + 1;
00039     view->str = malloc(sizeof(char) * view->size);
00040     if (view->str == NULL)
00041     {
00042         free(view);
00043         return NULL;
00044     }
00045     strncpy(view->str, str, view->size);
00046     return view;
00047 }
00048 
00049 FMC_SHARED void FMC_freeStrView(FMC_CStrView* view)
00050 {
00051     free(view->str);
00052     free(view);
00053 }
```

3.66 src/general/utils/FMC_str_view.h File Reference

Include dependency graph for FMC_str_view.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define FMC_STR_VIEW_H

Functions

- void FMC_freeStrView (FMC_CStrView ∗view)
- FMC_FUNC_MALLOC (FMC_freeStrView, 1) FMC_CStrView ∗FMC_allocStrView(const char ∗const str

Variables

- size_t len

3.66.1 Macro Definition Documentation

3.66.1.1 FMC_STR_VIEW_H

```
#define FMC_STR_VIEW_H
```

Definition at line 30 of file FMC_str_view.h.

3.66.2 Function Documentation

3.66.2.1 FMC_freeStrView()

```
void FMC_freeStrView (
            FMC_CStrView * view )
```

Definition at line 49 of file FMC_str_view.c.

References FManC_CStrView::str.

3.66.2.2 FMC_FUNC_MALLOC()

```
FMC_FUNC_MALLOC (
            FMC_freeStrView ,
            1 ) const
```

3.66.3 Variable Documentation

3.66.3.1 len

```
size_t len
```

Definition at line 39 of file FMC_str_view.h.

Referenced by FMC_FUNC_MALLOC().

3.67 FMC_str_view.h

Go to the documentation of this file.

```
/*

MIT License

Copyright (c) 2022-2023 Axel PASCON

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

*/

#pragma once

#ifndef FMC_STR_VIEW_H
#define FMC_STR_VIEW_H

#include "../FMC_general.h"

#ifndef FMC_makeStrView
    #define FMC_makeStrView(_str, _len) ((FMC_CStrView){ .str = _str, .size = (size_t)_len })
#endif

FMC_SHARED void FMC_freeStrView(FMC_CStrView* view);
FMC_SHARED FMC_FUNC_MALLOC(FMC_freeStrView, 1) FMC_CStrView* FMC_allocStrView(const char* const str,
size_t len);

#endif // FMC_STR_VIEW_H
```

Index