LP25 project

Generated by Doxygen 1.8.18

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Data Structure Documentation

3.1 _directory Struct Reference

```
#include <types.h>
```

Data Fields

- char ∗ name
- time_t mod_time
- struct _directory ∗ subdirs
- s_file ∗ files
- struct _directory ∗ next_dir

3.1.1 Detailed Description

Definition at line 48 of file types.h.

3.1.2 Field Documentation

3.1.2.1 files

```
s_file* files
```

list of the file in the directory

Definition at line 52 of file types.h.

3.1.2.2 mod_time

```
time_t mod_time
```

modification time of the directory

Definition at line 50 of file types.h.

3.1.2.3 name

```
char* name
```

name of the directory

Definition at line 49 of file types.h.

3.1.2.4 next_dir

```
struct _directory* next_dir
```

next directory in the list of directory

Definition at line 53 of file types.h.

3.1.2.5 subdirs

```
struct _directory* subdirs
```

list of the subdirs in the directory

Definition at line 51 of file types.h.

The documentation for this struct was generated from the following file:

- src/types.h

3.2 _file Struct Reference

```
#include <types.h>
```

Data Fields

- e_type file_type
- char ∗ name
- time_t mod_time
- uint64_t file_size
- unsigned char ∗ md5sum
- struct _file ∗ pointed_file
- struct _file ∗ next_file

3.2.1 Detailed Description

Definition at line 31 of file types.h.

3.2.2 Field Documentation

3.2.2.1 file_size

```
uint64_t file_size
```

size of the file

Definition at line 35 of file types.h.

3.2.2.2 file_type

```
e_type file_type
```

type of the file

Definition at line 32 of file types.h.

3.2.2.3 md5sum

```
unsigned char* md5sum
```

value of the md5sum of the file

Definition at line 36 of file types.h.

3.2.2.4 mod_time

```
time_t mod_time
```

modification time of the file

Definition at line 34 of file types.h.

3.2.2.5 name

```
char* name
```

name of the file

Definition at line 33 of file types.h.

3.2.2.6 next_file

```
struct _file* next_file
```

next file in the link list of file

Definition at line 38 of file types.h.

3.2.2.7 pointed_file

```
struct _file* pointed_file
```

file pointed by the file if it is a soft link

Definition at line 37 of file types.h.

The documentation for this struct was generated from the following file:

- src/types.h

3.3 s_directory Struct Reference

strucrure for a dir and all the data about it and all it's component

```
#include <types.h>
```

3.3.1 Detailed Description

strucrure for a dir and all the data about it and all it's component

Author

Florian CLOAREC

The documentation for this struct was generated from the following file:

- src/types.h

3.4 s_file Struct Reference

structure for the file and all the data about it

```
#include <types.h>
```

3.4.1 Detailed Description

structure for the file and all the data about it

Author

Florian CLOAREC

The documentation for this struct was generated from the following file:

- src/types.h

Chapter 4

File Documentation

4.1 src/include.h File Reference

header file that contain all the header file to include in the other part

```
#include <linux/limits.h>
#include <time.h>
#include <openssl/md5.h>
#include <stdlib.h>
#include <sys/types.h>
#include <dirent.h>
#include <stdio.h>
#include <stdint.h>
#include <string.h>
#include <fcntl.h>
#include <sys/stat.h>
#include <unistd.h>
#include <stdbool.h>
#include "types.h"
#include "scan.h"
#include "md5sum.h"
#include "utils.h"
#include "tree.h"
#include "save.h"
```

4.1.1 Detailed Description

header file that contain all the header file to include in the other part

Author

Florian Cloarec

Version

0.1

Date

07 June 2021

Copyright

GNU GENERAL PUBLIC LICENSE

4.2 src/main.c File Reference

main file that contain the main fuction of the project

```
#include "include.h"
```

Functions

- int main (int argc, char ∗argv[ ])

4.2.1 Detailed Description

main file that contain the main fuction of the project

Author

Florian Cloarec

Version

0.1

Date

15 June 2021

Copyright

GNU GENERAL PUBLIC LICENSE

4.2.2 Function Documentation

4.2.2.1 main()

```
int main (
            int argc,
            char * argv[ ] )
```

Definition at line 14 of file main.c.

4.3 src/md5sum.c File Reference

source file that contain the implementation of the function in the md5sum part of the project

```
#include "include.h"
```

Functions

- int compute_md5 (char *path, unsigned char buffer[ ])

  compute the md5sum of a file

4.3.1 Detailed Description

source file that contain the implementation of the function in the md5sum part of the project

Author

Florian Cloarec

Version

0.1

Date

15 June 2021

Copyright

GNU GENERAL PUBLIC LICENSE

4.3.2 Function Documentation

4.3.2.1 compute_md5()

```
int compute_md5 (
            char * path,
            unsigned char buffer[ ] )
```

compute the md5sum of a file

Parameters

path	: the path of the file to compute
buffer	: the buffer where the result is written

Returns

int : control value 1 if succes

Author

Florian CLOAREC

Definition at line 14 of file md5sum.c.

4.4 src/md5sum.h File Reference

header file of the function in the md5sum part of the project

Functions

- int compute_md5 (char ∗path, unsigned char buffer[ ])
  compute the md5sum of a file

4.4.1 Detailed Description

header file of the function in the md5sum part of the project

Author

Florian Cloarec

Version

0.1

Date

11 June 2021

Copyright

GNU GENERAL PUBLIC LICENSE

4.4.2 Function Documentation

4.4.2.1 compute_md5()

```
int compute_md5 (
              char * path,
              unsigned char buffer[ ] )
```

compute the md5sum of a file

Parameters

path	: the path of the file to compute
buffer	: the buffer where the result is written

Returns

int : control value 1 if succes

Author

Florian CLOAREC

Definition at line 14 of file md5sum.c.

4.5 src/save.c File Reference

source file that contain the implementation of the function in the save part of the project

```
#include "include.h"
```

Functions

- int save_to_file (s_directory ∗root, char ∗path_to_target, const char ∗current_path)
- int save_to_file_recursive (FILE ∗output, s_directory ∗current_dir, int depth, const char ∗current_path)
- int write_file (FILE ∗output, s_file file, const char ∗path_to_parent_dir)
- int write_directory (FILE ∗output, s_directory dir, const char ∗path_to_parent_dir)
- int write_other (FILE ∗output, s_file file, const char ∗path_to_parent_dir)
- char ∗ generateFileName ()

4.5.1 Detailed Description

source file that contain the implementation of the function in the save part of the project

Version

0.1

Date

15 June 2021

Copyright

GNU GENERAL PUBLIC LICENSE

4.5.2 Function Documentation

4.5.2.1 generateFileName()

```
char* generateFileName ( )
```

Generate file name for save_to_file

Returns

The filepath

Definition at line 124 of file save.c.

4.5.2.2 save_to_file()

```
int save_to_file (
            s_directory * root,
            char * path_to_target,
            const char * current_path )
```

Save the directory tree in the file

Parameters

root	root to save
path_to_target	file target
current_path	directory's path

Returns

int

Definition at line 13 of file save.c.

4.5.2.3 save_to_file_recursive()

```
int save_to_file_recursive (
            FILE * output,
            s_directory * current_dir,
            int indentation_level,
            const char * current_path )
```

Internal : function that is called recursively to save the tree

Parameters

output	output file
current_dir	directory to process
indentation_level	depth level (tabulations)
current_path	directory's path

Returns

int

Definition at line 28 of file save.c.

4.5.2.4 write_directory()

```
int write_directory (
            FILE * output,
            s_directory dir,
            const char * path_to_parent_dir )
```

Write a directory line

Parameters

output	the output file
dir	directory to write
path_to_parent_dir	the path to paste before the directory name

Returns

int

Definition at line 98 of file save.c.

4.5.2.5 write_file()

```
int write_file (
            FILE * output,
            s_file file,
            const char * path_to_parent_dir )
```

Write a file line

Parameters

output	the output file
file	file to write
path_to_parent_dir	the path to paste before the file name

Returns

int

Definition at line 73 of file save.c.

4.5.2.6 write_other()

```
int write_other (
            FILE * output,
            s_file file,
            const char * path_to_parent_dir )
```

Definition at line 111 of file save.c.

4.6 src/save.h File Reference

header file of the function in the save part of the project

Functions

- int save_to_file (s_directory ∗root, char ∗path_to_target, const char ∗current_path)
- int save_to_file_recursive (FILE ∗output, s_directory ∗current_dir, int indentation_level, const char ∗current↩ _path)
- int write_file (FILE ∗output, s_file file, const char ∗path_to_parent_dir)
- int write_directory (FILE ∗output, s_directory dir, const char ∗path_to_parent_dir)
- char ∗ generateFileName ()

4.6.1 Detailed Description

header file of the function in the save part of the project

Author

Florian Cloarec

Version

0.1

Date

15 June 2021

Copyright

GNU GENERAL PUBLIC LICENSE

4.6.2 Function Documentation

4.6.2.1 generateFileName()

```
char* generateFileName ( )
```

Generate file name for save_to_file

Returns

The filepath

Definition at line 124 of file save.c.

4.6.2.2 save_to_file()

```
int save_to_file (
             s_directory * root,
             char * path_to_target,
             const char * current_path )
```

Save the directory tree in the file

Parameters

root	root to save
path_to_target	file target
current_path	directory's path

Returns

int

Definition at line 13 of file save.c.

4.6.2.3 save_to_file_recursive()

```
int save_to_file_recursive (
             FILE * output,
             s_directory * current_dir,
             int indentation_level,
             const char * current_path )
```

Internal : function that is called recursively to save the tree

Parameters

output	output file
current_dir	directory to process
indentation_level	depth level (tabulations)
current_path	directory's path

Returns

int

Definition at line 28 of file save.c.

4.6.2.4 write_directory()

```
int write_directory (
            FILE * output,
            s_directory dir,
            const char * path_to_parent_dir )
```

Write a directory line

Parameters

output	the output file
dir	directory to write
path_to_parent_dir	the path to paste before the directory name

Returns

int

Definition at line 98 of file save.c.

4.6.2.5 write_file()

```
int write_file (
            FILE * output,
            s_file file,
            const char * path_to_parent_dir )
```

Write a file line

Parameters

output	the output file
file	file to write
path_to_parent_dir	the path to paste before the file name

Returns

int

Definition at line 73 of file save.c.

4.7 src/scan.c File Reference

source file that contain the implementation of the function in the scan part of the project

```
#include "include.h"
```

Functions

- s_directory ∗ process_dir (char ∗path, bool computeMd5)

  recursive function that scan a directory and create the struct in memory with all the sub directory and file
- s_file ∗ process_file (char ∗path, bool computeMd5)

  create and compute the struct s_file whit the data of a file

4.7.1 Detailed Description

source file that contain the implementation of the function in the scan part of the project

Author

Florian Cloarec

Version

0.1

Date

07 June 2021

Copyright

GNU GENERAL PUBLIC LICENSE

4.7.2 Function Documentation

4.7.2.1 process_dir()

```
s_directory* process_dir (
             char * path,
             bool computeMd5 )
```

recursive function that scan a directory and create the struct in memory with all the sub directory and file

Parameters

path	: the path of the directory to scan

Returns

s_directory∗ : the struct with all the data about the directory

Author

Florian CLOAREC

Definition at line 14 of file scan.c.

4.7.2.2 process_file()

```
s_file* process_file (
            char * path,
            bool computeMd5 )
```

create and compute the struct s_file whit the data of a file

Parameters

path	: the path of the file to compute

Returns

s_file∗ : the struct with all the data avout the file

Author

Florian CLOAREC

Definition at line 67 of file scan.c.

4.8 src/scan.h File Reference

header file of the function and struct in the scan part of the project

Functions

- s_directory ∗ process_dir (char ∗path, bool computeMd5)

  recursive function that scan a directory and create the struct in memory with all the sub directory and file
- s_file ∗ process_file (char ∗path, bool computeMd5)

  create and compute the struct s_file whit the data of a file

4.8.1 Detailed Description

header file of the function and struct in the scan part of the project

Author

Florian Cloarec

Version

0.1

Date

07 June 2021

Copyright

GNU GENERAL PUBLIC LICENSE

4.8.2 Function Documentation

4.8.2.1 process_dir()

```
s_directory* process_dir (
            char * path,
            bool computeMd5 )
```

recursive function that scan a directory and create the struct in memory with all the sub directory and file

Parameters

path	: the path of the directory to scan

Returns

s_directory∗ : the struct with all the data about the directory

Author

Florian CLOAREC

Definition at line 14 of file scan.c.

4.8.2.2 process_file()

```
s_file* process_file (
            char * path,
            bool computeMd5 )
```

create and compute the struct s_file whit the data of a file

Parameters

path	: the path of the file to compute

Returns

s_file∗ : the struct with all the data avout the file

Author

Florian CLOAREC

Definition at line 67 of file scan.c.

4.9 src/tree.c File Reference

source file that contain the implementation of the function in the tree part of the project

```
#include "include.h"
```

Functions

- int append_subdir (s_directory ∗child, s_directory ∗parent)

  add a element at the end of the link list of directory
- int append_file (s_file ∗child, s_directory ∗parent)

  add an element at the end of the link list of file
- void clear_files (s_directory ∗parent)

  clear propely a whole link list of file
- void clear_subdirs (s_directory ∗parent)

  clear propely a vhole link list of directory
- void free_s_file (s_file ∗file)

  free a s_file variable
- void free_s_directory (s_directory ∗dir)

  free a s_directory variable

4.9.1 Detailed Description

source file that contain the implementation of the function in the tree part of the project

Version

0.1

Date

15 June 2021

Copyright

GNU GENERAL PUBLIC LICENSE

4.9.2 Function Documentation

4.9.2.1 append_file()

```
int append_file (
             s_file * child,
             s_directory * parent )
```

add an element at the end of the link list of file

Parameters

child	: the element to add
parent	: the list to ad it

Returns

int : control value

Author

Florian CLOAREC

Definition at line 31 of file tree.c.

4.9.2.2 append_subdir()

```
int append_subdir (
             s_directory * child,
             s_directory * parent )
```

add a element at the end of the link list of directory

Parameters

child	: the element to add
parent	: the list to add it

Returns

int : control value

Author

Florian CLOAREC

Definition at line 13 of file tree.c.

4.9.2.3 clear_files()

```
void clear_files (
            s_directory * parent )
```

clear propely a whole link list of file

Parameters

parent	: the list to clear

Author

Florian CLOAREC

Definition at line 47 of file tree.c.

4.9.2.4 clear_subdirs()

```
void clear_subdirs (
            s_directory * parent )
```

clear propely a vhole link list of directory

Parameters

parent	: the list to clear

Author

Florian CLOAREC

Definition at line 58 of file tree.c.

4.9.2.5 free_s_directory()

```
void free_s_directory (
            s_directory * dir )
```

free a s_directory variable

Parameters

dir	: the dir to clear

Author

Florian CLOAREC

Definition at line 78 of file tree.c.

4.9.2.6 free_s_file()

```
void free_s_file (
            s_file * file )
```

free a s_file variable

Parameters

file	: the file to clear

Author

Florian CLOAREC

Definition at line 72 of file tree.c.

4.10 src/tree.h File Reference

header file of the function and struct in the tree part of the project

Functions

- int append_subdir (s_directory ∗child, s_directory ∗parent)
  add a element at the end of the link list of directory
- int append_file (s_file ∗child, s_directory ∗parent)
  add an element at the end of the link list of file
- void clear_files (s_directory ∗parent)
  clear propely a whole link list of file
- void clear_subdirs (s_directory ∗parent)
  clear propely a vhole link list of directory
- void free_s_file (s_file ∗file)
  free a s_file variable
- void free_s_directory (s_directory ∗dir)
  free a s_directory variable

4.10.1 Detailed Description

header file of the function and struct in the tree part of the project

Author

Florian Cloarec

Version

0.1

Date

15 June 2021

Copyright

GNU GENERAL PUBLIC LICENSE

4.10.2 Function Documentation

4.10.2.1 append_file()

```
int append_file (
            s_file ∗ child,
            s_directory ∗ parent )
```

add an element at the end of the link list of file

Parameters

child	: the element to add
parent	: the list to ad it

Returns

int : control value

Author

Florian CLOAREC

Definition at line 31 of file tree.c.

4.10.2.2 append_subdir()

```
int append_subdir (
            s_directory * child,
            s_directory * parent )
```

add a element at the end of the link list of directory

Parameters

child	: the element to add
parent	: the list to add it

Returns

int : control value

Author

Florian CLOAREC

Definition at line 13 of file tree.c.

4.10.2.3 clear_files()

```
void clear_files (
            s_directory * parent )
```

clear propely a whole link list of file

Parameters

parent	: the list to clear

Author

Florian CLOAREC

Definition at line 47 of file tree.c.

4.10.2.4 clear_subdirs()

```
void clear_subdirs (
            s_directory * parent )
```

clear propely a vhole link list of directory

Parameters

parent	: the list to clear

Author

Florian CLOAREC

Definition at line 58 of file tree.c.

4.10.2.5 free_s_directory()

```
void free_s_directory (
            s_directory * dir )
```

free a s_directory variable

Parameters

dir	: the dir to clear

Author

Florian CLOAREC

Definition at line 78 of file tree.c.

4.10.2.6 free_s_file()

```
void free_s_file (
            s_file * file )
```

free a s_file variable

Parameters

file	: the file to clear

Author

Florian CLOAREC

Definition at line 72 of file tree.c.

4.11 src/types.h File Reference

file that contain all the struct used in this project

Data Structures

- struct _file
- struct _directory

Typedefs

- typedef struct _file s_file
- typedef struct _directory s_directory

Enumerations

- enum e_type { DIRECTORY, REGULAR_FILE, SYMBOLIK_LINK, OTHER_TYPE }
  possible type for a file

4.11.1 Detailed Description

file that contain all the struct used in this project

Author

Florian Cloarec

Version

0.1

Date

15 June 2021

Copyright

GNU GENERAL PUBLIC LICENSE

4.11.2 Typedef Documentation

4.11.2.1 s_directory

```
typedef struct _directory s_directory
```

4.11.2.2 s_file

```
typedef struct _file s_file
```

4.11.3 Enumeration Type Documentation

4.11.3.1 e_type

```
enum e_type
```

possible type for a file

Author

Florian CLOAREC

Enumerator

DIRECTORY	
REGULAR_FILE	
SYMBOLIK_LINK	
OTHER_TYPE	

Definition at line 23 of file types.h.

4.12 src/utils.c File Reference

implementation file of the function who can be util for all part of the project

```
#include "include.h"
```

Functions

- char ∗ getFilePath (char ∗fileName, char ∗dirPath)

  Get the File Path object.
- char ∗ getFileName (char ∗path)

  Get the File Name object.

4.12.1 Detailed Description

implementation file of the function who can be util for all part of the project

Author

Florian Cloarec

Version

0.1

Date

07 June 2021

Copyright

GNU GENERAL PUBLIC LICENSE

4.12.2 Function Documentation

4.12.2.1 getFileName()

```
char* getFileName (
            char * path )
```

Get the File Name object.

Parameters

path	: the name of the file

Returns

char∗ : the pointer on the created string

Author

Florian CLOAREC

Definition at line 24 of file utils.c.

4.12.2.2 getFilePath()

```
char* getFilePath (
            char * fileName,
            char * dirPath )
```

Get the File Path object.

Parameters

fileName	: the name of the file
dirPath	: the path of the file

Returns

char∗ : the pointer on the created string

Author

Florian CLOAREC

Definition at line 14 of file utils.c.

4.13 src/utils.h File Reference

header file of the function and struct who can be util for all part of the project

Functions

- char ∗ getFilePath (char ∗fileName, char ∗dirPath)

  Get the File Path object.
- char ∗ getFileName (char ∗path)

  Get the File Name object.

4.13.1 Detailed Description

header file of the function and struct who can be util for all part of the project

Author

Florian Cloarec

Version

0.1

Date

07 June 2021

Copyright

GNU GENERAL PUBLIC LICENSE

4.13.2 Function Documentation

4.13.2.1 getFileName()

```
char* getFileName (
            char * path )
```

Get the File Name object.

Parameters

path	: the name of the file

Returns

char∗ : the pointer on the created string

Author

Florian CLOAREC

Definition at line 24 of file utils.c.

4.13.2.2 getFilePath()

```
char* getFilePath (
            char * fileName,
            char * dirPath )
```

Get the File Path object.

Parameters

fileName	: the name of the file
dirPath	: the path of the file

Returns

char∗ : the pointer on the created string

Author

Florian CLOAREC

Definition at line 14 of file utils.c.

Index