

Csuper - Compteur de Score Universel Permettant l'Exemption de Reflexion

2.4.0

Generated by Doxygen 1.8.6

Sun May 18 2014 12:57:01

Contents

1	Data Structure Index	1
1.1	Data Structures	1
2	File Index	3
2.1	File List	3
3	Data Structure Documentation	5
3.1	csuStruct Struct Reference	5
3.1.1	Detailed Description	5
3.1.2	Field Documentation	5
3.1.2.1	config	5
3.1.2.2	day	5
3.1.2.3	distributor	6
3.1.2.4	month	6
3.1.2.5	nb_player	6
3.1.2.6	nb_turn	6
3.1.2.7	player_names	6
3.1.2.8	point	6
3.1.2.9	rank	6
3.1.2.10	size_max_name	6
3.1.2.11	total_points	6
3.1.2.12	version	6
3.1.2.13	year	6
3.2	game_config Struct Reference	7
3.2.1	Detailed Description	7
3.2.2	Field Documentation	7
3.2.2.1	begin_score	7
3.2.2.2	decimal_place	7
3.2.2.3	first_way	7
3.2.2.4	max	7
3.2.2.5	name	7
3.2.2.6	nb_max	7

3.2.2.7	turn_by_turn	7
3.2.2.8	use_distributor	8
3.3	list_game_config Struct Reference	8
3.3.1	Field Documentation	8
3.3.1.1	name_game_config	8
3.3.1.2	nb_config	8
4	File Documentation	9
4.1	csu_files.c File Reference	9
4.1.1	Detailed Description	9
4.1.2	Function Documentation	9
4.1.2.1	openFileCsuExtension	9
4.1.2.2	readCsuFile	10
4.1.2.3	writeCsuFile	10
4.1.2.4	writeFileNewTurn	11
4.2	csu_files.h File Reference	11
4.2.1	Detailed Description	12
4.2.2	Macro Definition Documentation	12
4.2.2.1	FILE_EXTENSION	12
4.2.2.2	SIZE_MAX_FILE_NAME	12
4.2.2.3	STRING_CHECK_CSU_FILE	12
4.2.3	Function Documentation	12
4.2.3.1	openFileCsuExtension	12
4.2.3.2	readCsuFile	13
4.2.3.3	writeCsuFile	13
4.2.3.4	writeFileNewTurn	14
4.3	csu_struct.c File Reference	14
4.3.1	Detailed Description	15
4.3.2	Function Documentation	15
4.3.2.1	addDistributorCsuStruct	15
4.3.2.2	closeCsuStruct	15
4.3.2.3	endNewTurn	16
4.3.2.4	exceedMaxNumber	16
4.3.2.5	maxNbTurn	16
4.3.2.6	newCsuStruct	16
4.3.2.7	rankCalculation	17
4.3.2.8	searchPlayerIndex	17
4.3.2.9	startNewTurn	18
4.4	csu_struct.h File Reference	18
4.4.1	Detailed Description	19

4.4.2	Macro Definition Documentation	19
4.4.2.1	SIZE_MAX_NAME	19
4.4.2.2	VERSION	19
4.4.3	Function Documentation	19
4.4.3.1	addDistributorCsuStruct	19
4.4.3.2	closeCsuStruct	20
4.4.3.3	endNewTurn	20
4.4.3.4	exceedMaxNumber	20
4.4.3.5	maxNbTurn	21
4.4.3.6	newCsuStruct	22
4.4.3.7	rankCalculation	22
4.4.3.8	searchPlayerIndex	22
4.4.3.9	startNewTurn	23
4.5	file_system_path.c File Reference	23
4.5.1	Detailed Description	24
4.5.2	Function Documentation	24
4.5.2.1	changeSystemPath	24
4.5.2.2	createFileSystemPath	24
4.5.2.3	readFileSystemPath	25
4.5.2.4	readHomePath	25
4.5.2.5	readHomePathSlash	26
4.5.2.6	readSystemPath	26
4.6	file_system_path.h File Reference	26
4.6.1	Detailed Description	27
4.6.2	Macro Definition Documentation	27
4.6.2.1	FILE_NAME_SYSTEM_PATH	27
4.6.2.2	MAIN_FOLDER_NAME	27
4.6.3	Function Documentation	27
4.6.3.1	changeSystemPath	27
4.6.3.2	createFileSystemPath	28
4.6.3.3	readFileSystemPath	28
4.6.3.4	readHomePath	29
4.6.3.5	readHomePathSlash	29
4.6.3.6	readSystemPath	29
4.7	game_config.c File Reference	30
4.7.1	Detailed Description	30
4.7.2	Function Documentation	31
4.7.2.1	addConfigListFile	31
4.7.2.2	closeListGameConfig	31
4.7.2.3	exportConfigFile	31

4.7.2.4	importConfigFile	32
4.7.2.5	makeConfigListFile	32
4.7.2.6	newConfigFile	33
4.7.2.7	newListGameConfig	33
4.7.2.8	readConfigFile	34
4.7.2.9	readConfigListFile	34
4.7.2.10	removeConfigFile	35
4.7.2.11	removeConfigListFile	36
4.8	game_config.h File Reference	36
4.8.1	Detailed Description	37
4.8.2	Macro Definition Documentation	37
4.8.2.1	CONFIGURATION_FILE_NAME	37
4.8.2.2	CONFIGURATION_FOLDER_NAME	37
4.8.2.3	STRING_CHECK_GAME_CONFIG	37
4.8.3	Function Documentation	37
4.8.3.1	addConfigListFile	37
4.8.3.2	closeListGameConfig	38
4.8.3.3	exportConfigFile	38
4.8.3.4	importConfigFile	39
4.8.3.5	makeConfigListFile	39
4.8.3.6	newConfigFile	40
4.8.3.7	newListGameConfig	40
4.8.3.8	readConfigFile	41
4.8.3.9	readConfigListFile	41
4.8.3.10	removeConfigFile	42
4.8.3.11	removeConfigListFile	43
4.9	libcsuper.h File Reference	43
4.9.1	Detailed Description	43
4.10	main_argument.c File Reference	43
4.10.1	Detailed Description	44
4.10.2	Function Documentation	44
4.10.2.1	displayHelp	44
4.10.2.2	searchArgument	44
4.11	main_argument.h File Reference	45
4.11.1	Detailed Description	45
4.11.2	Macro Definition Documentation	46
4.11.2.1	HELP	46
4.11.2.2	OPEN_FILE	46
4.11.2.3	READ_FILE	46
4.11.2.4	STRING_HELP	46

4.11.2.5	STRING_HELP_RED	46
4.11.2.6	STRING_OPEN_FILE	46
4.11.2.7	STRING_OPEN_FILE_RED	46
4.11.2.8	STRING_READ_FILE	46
4.11.2.9	STRING_READ_FILE_RED	46
4.11.3	Function Documentation	46
4.11.3.1	displayHelp	46
4.11.3.2	searchArgument	47
4.12	share.c File Reference	47
4.12.1	Detailed Description	48
4.12.2	Function Documentation	48
4.12.2.1	addFileCsuExtension	48
4.12.2.2	clearScreen	48
4.12.2.3	closeFile	48
4.12.2.4	compareFloatAscending	49
4.12.2.5	compareFloatDescending	49
4.12.2.6	deleteFile	49
4.12.2.7	libcsuper_initialize	50
4.12.2.8	myAlloc	50
4.12.2.9	myRealloc	51
4.12.2.10	openFile	51
4.12.2.11	readFileSize	51
4.12.2.12	renameFile	51
4.12.2.13	wrongChoice	52
4.13	share.h File Reference	52
4.13.1	Detailed Description	53
4.13.2	Macro Definition Documentation	53
4.13.2.1	_	53
4.13.2.2	MY_FALSE	53
4.13.2.3	MY_TRUE	53
4.13.3	Function Documentation	53
4.13.3.1	addFileCsuExtension	53
4.13.3.2	clearScreen	54
4.13.3.3	closeFile	54
4.13.3.4	compareFloatAscending	54
4.13.3.5	compareFloatDescending	55
4.13.3.6	deleteFile	55
4.13.3.7	libcsuper_initialize	55
4.13.3.8	myAlloc	55
4.13.3.9	myRealloc	56

4.13.3.10 openFile	56
4.13.3.11 readFileSize	57
4.13.3.12 renameFile	57
4.13.3.13 wrongChoice	58
Index	59

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

csuStruct	5
game_config	7
list_game_config	8

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

csu_files.c	Files management	9
csu_files.h	Files management	11
csu_struct.c	Management of the csu files	14
csu_struct.h	Management of the csu files header	18
file_system_path.c	Fonctions qui l'emrankment des fichiers sauvegardes	23
file_system_path.h	Prototypes des fonctions qui l'emrankment des fichiers sauvegardes	26
game_config.c	Game configuration	30
game_config.h	Game configurations	36
libcsuper.h	Inclusion of all header files of libcsuper	43
main_argument.c	Begin csuper	43
main_argument.h	Begin csuper	45
share.c	Essential function of libcsuper	47
share.h	Header for the essential function of libcsuper	52

Chapter 3

Data Structure Documentation

3.1 csuStruct Struct Reference

```
#include <csu_struct.h>
```

Data Fields

- float [version](#)
- float [size_max_name](#)
- float [day](#)
- float [month](#)
- float [year](#)
- float [nb_player](#)
- [game_config](#) config
- char ** [player_names](#)
- float * [total_points](#)
- float * [rank](#)
- float * [nb_turn](#)
- float [distributor](#)
- float ** [point](#)

3.1.1 Detailed Description

Represent a csu file

Represent a list of game configuration

3.1.2 Field Documentation

3.1.2.1 [game_config](#) config

The game configuration.

3.1.2.2 float [day](#)

Day of the structure creation.

3.1.2.3 float distributor

Index of the distributor.

3.1.2.4 float month

Month of the structure creation.

3.1.2.5 float nb_player

Number of player.

3.1.2.6 float* nb_turn

Array containing the number of turn of all players.

3.1.2.7 char** player_names

Array containing the name of all players.

3.1.2.8 float** point

Array containing the points of all players in each turn.

3.1.2.9 float* rank

Array containing the rank of all players.

3.1.2.10 float size_max_name

Maximum size that can reach a player name.

3.1.2.11 float* total_points

Array containing the total score of all players.

3.1.2.12 float version

Version of the structure.

3.1.2.13 float year

Year of the structure creation.

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

- [csu_struct.h](#)

3.2 game_config Struct Reference

```
#include <csu_struct.h>
```

Data Fields

- float [nb_max](#)
- char [first_way](#)
- char [turn_by_turn](#)
- char [use_distributor](#)
- char [decimal_place](#)
- char [max](#)
- char [name](#) [[SIZE_MAX_NAME](#)]
- float [begin_score](#)

3.2.1 Detailed Description

Represent a game configuration

3.2.2 Field Documentation

3.2.2.1 float begin_score

The score of all players in the beginning of the game

3.2.2.2 char decimal_place

The number of decimal place which are display

3.2.2.3 char first_way

Is 1 if the first those has the maximum of points, -1 otherwise

3.2.2.4 char max

Is 1 if the game use a maximum, 0 if it's a minimum

3.2.2.5 char name[SIZE_MAX_NAME]

The name of the game configuration

3.2.2.6 float nb_max

Number maximum or minimum that can reach a player.

3.2.2.7 char turn_by_turn

Is 1 if the game is on turn by turn, 0 otherwise

3.2.2.8 char use_distributor

Is 1 if the game use a distributor, 0 otherwise

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

- [csu_struct.h](#)

3.3 list_game_config Struct Reference

```
#include <game_config.h>
```

Data Fields

- int [nb_config](#)
- char ** [name_game_config](#)

3.3.1 Field Documentation

3.3.1.1 char** name_game_config

The list of the game configuration.

3.3.1.2 int nb_config

Number of game configuration.

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

- [game_config.h](#)

Chapter 4

File Documentation

4.1 csu_files.c File Reference

Files management.

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

Functions

- FILE * [openFileCsuExtension](#) (char file_name[], char mode[])
- csuStruct * [readCsuFile](#) (char *file_name)
- int [writeCsuFile](#) (char *file_name, csuStruct *ptr_csu_struct)
- int [writeFileNewTurn](#) (char *file_name, csuStruct *ptr_csu_struct)

4.1.1 Detailed Description

Files management.

Author

Remi BERTHO

Date

27/04/14

Version

2.2.0

4.1.2 Function Documentation

4.1.2.1 FILE * [openFileCsuExtension](#) (char *file_name*[], char *mode*[])

Open a file with his name and with a specific mode and add the file extension if necessary.

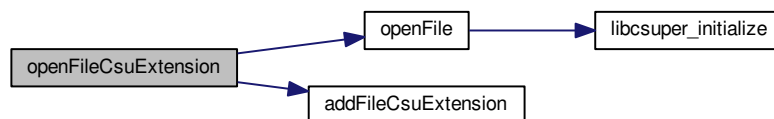
Parameters

in	<i>file_name[]</i>	the filename
in	<i>mode[]</i>	the mode

Returns

a pointer on the open file, NULL if there is a problem

Here is the call graph for this function:

**4.1.2.2 csuStruct * readCsuFile (char * file_name)**

Read the file with the name `file_name` and copy the result in a new csu structure.

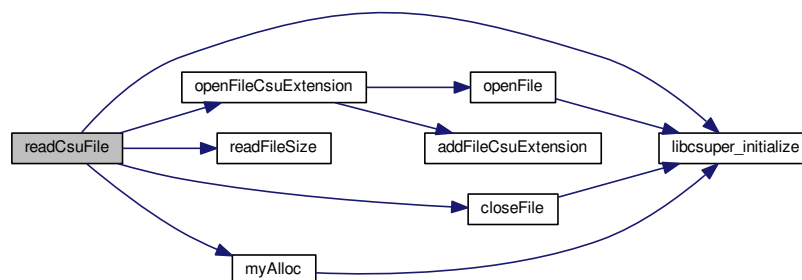
Parameters

in	<i>file_name[]</i>	the filename
----	--------------------	--------------

Returns

a pointer on the new csu structure, NULL if there is a problem

Here is the call graph for this function:

**4.1.2.3 int writeCsuFile (char * file_name, csuStruct * ptr_csu_struct)**

Write a csu file

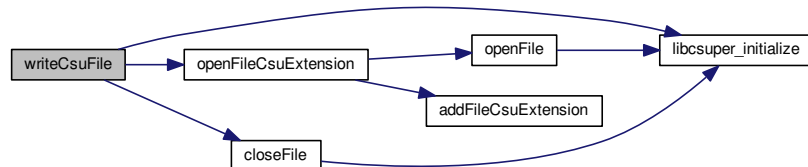
Parameters

in	<i>*file_name</i>	the filename
in	<i>*ptr_csu_struct</i>	a pointer on a csuStruct

Returns

MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:



4.1.2.4 void writeFileNewTurn (char * file_name, csuStruct * ptr_csu_struct)

Update the file with the new scores

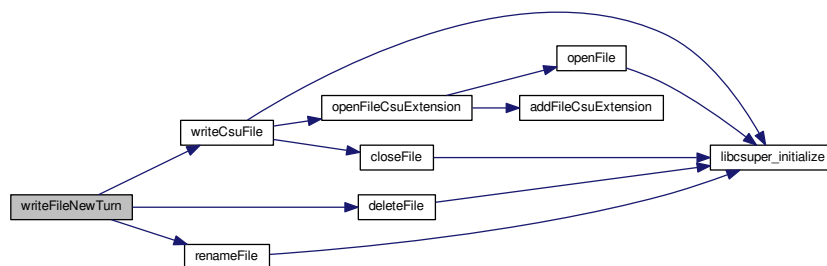
Parameters

in	<i>*file_name</i>	the filename
in	<i>*ptr_csu_struct</i>	a pointer on a csuStruct

Returns

MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:



4.2 csu_files.h File Reference

Files management.

```
#include "csu_struct.h"
#include <unistd.h>
```

Macros

- `#define SIZE_MAX_FILE_NAME 250`
- `#define FILE_EXTENSION "csu"`
- `#define STRING_CHECK_CSU_FILE "CompteurScoreUniversel"`

Functions

- `FILE * openFileCsuExtension (char file_name[], char mode[])`
- `csuStruct * readCsuFile (char *file_name)`
- `int writeCsuFile (char *file_name, csuStruct *ptr_csu_struct)`
- `int writeFileNewTurn (char *file_name, csuStruct *ptr_csu_struct)`

4.2.1 Detailed Description

Files management.

Author

Remi BERTHO

Date

16/04/14

Version

2.2.0

4.2.2 Macro Definition Documentation

4.2.2.1 `#define FILE_EXTENSION "csu"`

Define the file extension to "csu"

4.2.2.2 `#define SIZE_MAX_FILE_NAME 250`

Define the size maximum of a filename to 250

4.2.2.3 `#define STRING_CHECK_CSU_FILE "CompteurScoreUniversel"`

String for checking if the file is a csu file.

4.2.3 Function Documentation

4.2.3.1 `FILE* openFileCsuExtension (char file_name[], char mode[])`

Open a file with his name and with a specific mode and add the file extension if necessary.

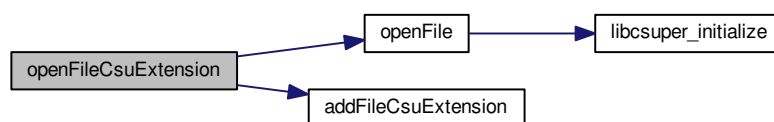
Parameters

in	<i>file_name[]</i>	the filename
in	<i>mode[]</i>	the mode

Returns

a pointer on the open file, NULL if there is a problem

Here is the call graph for this function:



4.2.3.2 csuStruct* readCsuFile (char * file_name)

Read the file with the name file_name and copy the result in a new csu structure.

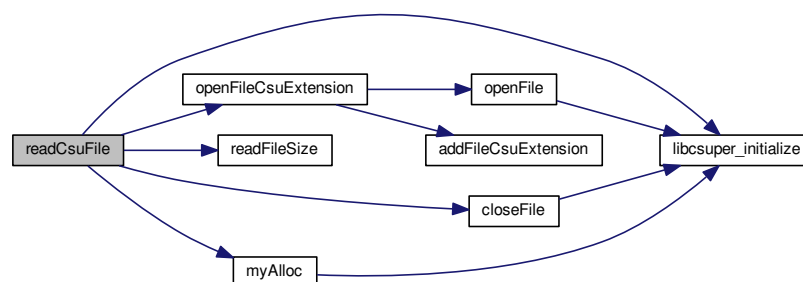
Parameters

in	<i>file_name[]</i>	the filename
----	--------------------	--------------

Returns

a pointer on the new csu structure, NULL if there is a problem

Here is the call graph for this function:



4.2.3.3 int writeCsuFile (char * file_name, csuStruct * ptr_csu_struct)

Write a csu file

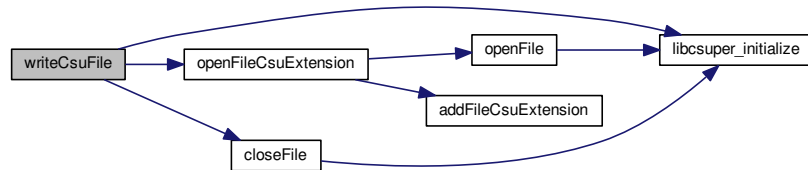
Parameters

in	<i>*file_name</i>	the filename
in	<i>*ptr_csu_struct</i>	a pointer on a csuStruct

Returns

MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:



4.2.3.4 int writeFileNewTurn (char * file_name, csuStruct * ptr_csu_struct)

Update the file with the new scores

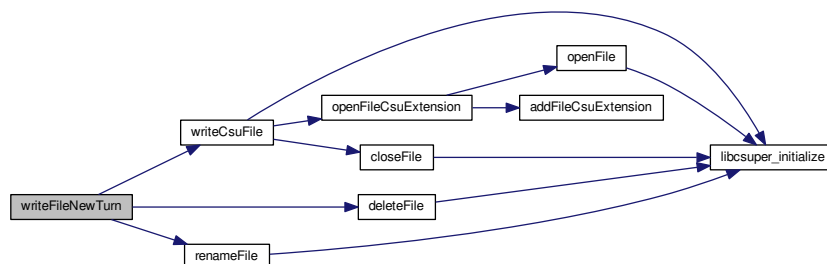
Parameters

in	<i>*file_name</i>	the filename
in	<i>*ptr_csu_struct</i>	a pointer on a csuStruct

Returns

MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:



4.3 csu_struct.c File Reference

Management of the csu files.

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

Functions

- [csuStruct](#) * [newCsuStruct](#) (float nb_player, [game_config](#) config)
- void [closeCsuStruct](#) ([csuStruct](#) *ptr_csu_struct)
- void [startNewTurn](#) ([csuStruct](#) *ptr_csu_struct, int index_player)
- void [endNewTurn](#) ([csuStruct](#) *ptr_csu_struct, int index_player)
- void [rankCalculation](#) ([csuStruct](#) *ptr_csu_struct)
- void [addDistributorCsuStruct](#) ([csuStruct](#) *ptr_csu_struct, char *distributor_name)
- int [exceedMaxNumber](#) ([csuStruct](#) *ptr_csu_struct)
- int [maxNbTurn](#) ([csuStruct](#) *ptr_csu_struct)
- int [searchPlayerIndex](#) ([csuStruct](#) *ptr_csu_struct, char *player_name)

4.3.1 Detailed Description

Management of the csu files.

Author

Remi BERTHO

Date

15/04/14

Version

2.2.0

4.3.2 Function Documentation

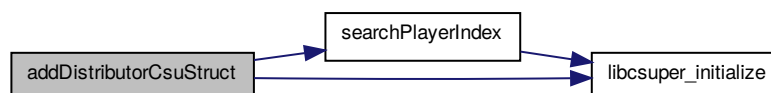
4.3.2.1 void [addDistributorCsuStruct](#) ([csuStruct](#) * *ptr_csu_struct*, char * *distributor_name*)

Add the distributor on the structure

Parameters

in	* <i>distributor_name</i>	the name of the distributor
in	* <i>ptr_csu_struct</i>	a pointer on a csuStruct

Here is the call graph for this function:



4.3.2.2 void [closeCsuStruct](#) ([csuStruct](#) * *ptr_csu_struct*)

Free a [csuStruct](#)

Parameters

in, out	<i>*ptr_csu_struct</i>	a pointer to the csuStruct
---------	------------------------	--

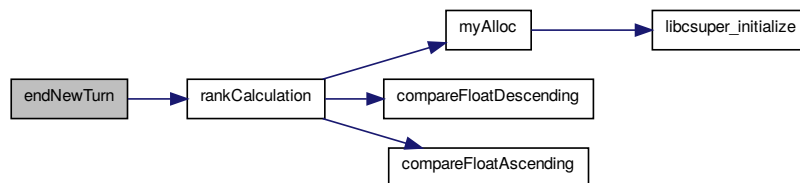
4.3.2.3 void endNewTurn (csuStruct * ptr_csu_struct, int index_player)

Update the total points, the number of turn, the distributor and the rank for a new turn

Parameters

in, out	<i>*ptr_csu_struct</i>	a pointer on a csuStruct
in, out	<i>index_player</i>	index_player the index of the player who begin a new turn, -1 if everybody begin a new turn

Here is the call graph for this function:



4.3.2.4 int exceedMaxNumber (csuStruct * ptr_csu_struct)

Check if someone exceed the maximum number

Parameters

in	<i>*ptr_csu_struct</i>	a pointer on a csuStruct
----	------------------------	--

Returns

MY_TRUE if someone exceed, MY_FALSE otherwise

4.3.2.5 int maxNbTurn (csuStruct * ptr_csu_struct)

Search the maximal number of turn

Parameters

in	<i>*ptr_csu_struct</i>	a pointer on a csuStruct
----	------------------------	--

Returns

the maximal number of turn

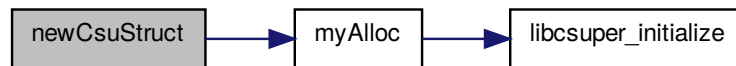
4.3.2.6 csuStruct * newCsuStruct (float nb_player, game_config config)

Create a new [csuStruct](#) from a game configuration and the number of player.

Parameters

in	<i>nb_player</i>	the number of player
in	<i>config</i>	the game configuration

Here is the call graph for this function:



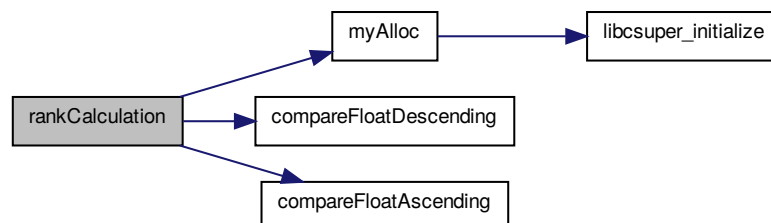
4.3.2.7 void rankCalculation (csuStruct * ptr_csu_struct)

Calculate the rank

Parameters

in, out	<i>*ptr_csu_struct</i>	a pointer on a csuStruct
---------	------------------------	--

Here is the call graph for this function:



4.3.2.8 int searchPlayerIndex (csuStruct * ptr_csu_struct, char * player_name)

Search the index of a person

Parameters

in	<i>*player_name</i>	the name of the player
in	<i>*ptr_csu_struct</i>	a pointer on a csuStruct

Returns

the index, -1 if there is not found

Here is the call graph for this function:



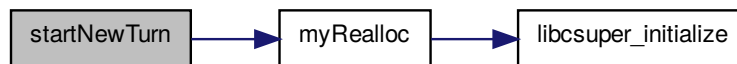
4.3.2.9 void startNewTurn (csuStruct * ptr_csu_struct, int index_player)

Reallocate the memory for the point to begin a new turn.

Parameters

in, out	<i>*ptr_csu_struct</i>	a pointer on a csuStruct
in, out	<i>index_player</i>	the index of the player who begin a new turn, -1 if everybody begin a new turn

Here is the call graph for this function:



4.4 csu_struct.h File Reference

Management of the csu files header.

```
#include <time.h>
#include <float.h>
#include "share.h"
```

Data Structures

- struct [game_config](#)
- struct [csuStruct](#)

Macros

- #define [SIZE_MAX_NAME](#) 30
- #define [VERSION](#) 1.4

Functions

- `csuStruct * newCsuStruct` (float nb_player, `game_config` config)
- void `closeCsuStruct` (`csuStruct` *ptr_csu_struct)
- void `startNewTurn` (`csuStruct` *ptr_csu_struct, int index_player)
- void `endNewTurn` (`csuStruct` *ptr_csu_struct, int index_player)
- void `rankCalculation` (`csuStruct` *ptr_csu_struct)
- void `addDistributorCsuStruct` (`csuStruct` *ptr_csu_struct, char *distributor_name)
- int `exceedMaxNumber` (`csuStruct` *ptr_csu_struct)
- int `maxNbTurn` (`csuStruct` *ptr_csu_struct)
- int `searchPlayerIndex` (`csuStruct` *ptr_csu_struct, char *player_name)

4.4.1 Detailed Description

Management of the csu files header.

Author

Remi BERTHO

Date

16/04/14

Version

2.2.0

4.4.2 Macro Definition Documentation

4.4.2.1 #define SIZE_MAX_NAME 30

Define size max of name to 30

4.4.2.2 #define VERSION 1.4

Define the version to 1.4

4.4.3 Function Documentation

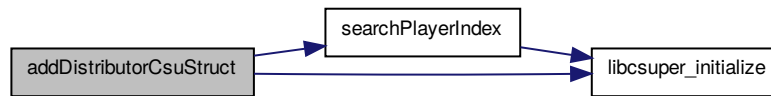
4.4.3.1 void addDistributorCsuStruct (csuStruct * ptr_csu_struct, char * distributor_name)

Add the distributor on the structure

Parameters

in	* <i>distributor_name</i>	the name of the distributor
in	* <i>ptr_csu_struct</i>	a pointer on a <code>csuStruct</code>

Here is the call graph for this function:



4.4.3.2 void closeCsuStruct (csuStruct * ptr_csu_struct)

Free a [csuStruct](#)

Parameters

in, out	*ptr_csu_struct	a pointer to the csuStruct
---------	-----------------	--

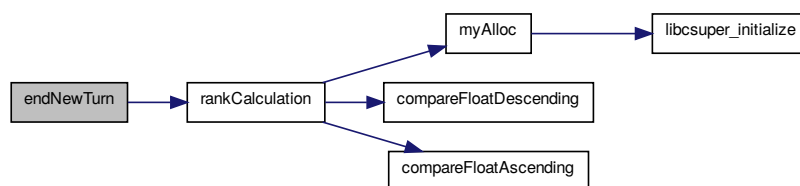
4.4.3.3 void endNewTurn (csuStruct * ptr_csu_struct, int index_player)

Update the total points, the number of turn, the distributor and the rank for a new turn

Parameters

in, out	*ptr_csu_struct	a pointer on a csuStruct
in, out	index_player	index_player the index of the player who begin a new turn, -1 if everybody begin a new turn

Here is the call graph for this function:



4.4.3.4 int exceedMaxNumber (csuStruct * ptr_csu_struct)

Check if someone exceed the maximum number

Parameters

in	*ptr_csu_struct	a pointer on a csuStruct
----	-----------------	--

Returns

MY_TRUE if someone exceed, MY_FALSE otherwise

4.4.3.5 int maxNbTurn (csuStruct * *ptr_csu_struct*)

Search the maximal number of turn

Parameters

in	<i>*ptr_csu_struct</i>	a pointer on a csuStruct
----	------------------------	--

Returns

the maximal number of turn

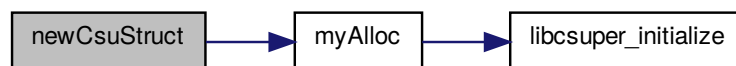
4.4.3.6 `csuStruct* newCsuStruct (float nb_player, game_config config)`

Create a new [csuStruct](#) from a game configuration and the number of player.

Parameters

in	<i>nb_player</i>	the number of player
in	<i>config</i>	the game configuration

Here is the call graph for this function:



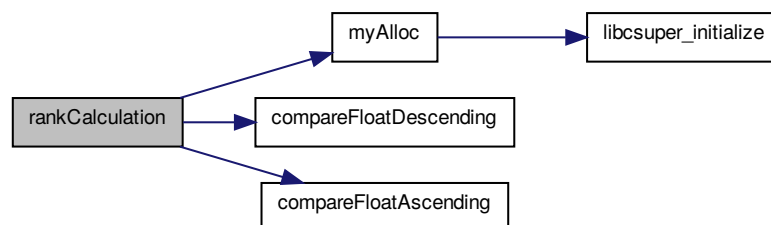
4.4.3.7 `void rankCalculation (csuStruct * ptr_csu_struct)`

Calculate the rank

Parameters

in, out	<i>*ptr_csu_struct</i>	a pointer on a csuStruct
---------	------------------------	--

Here is the call graph for this function:



4.4.3.8 `int searchPlayerIndex (csuStruct * ptr_csu_struct, char * player_name)`

Search the index of a person

Parameters

in	<i>*player_name</i>	the name of the player
in	<i>*ptr_csu_struct</i>	a pointer on a csuStruct

Returns

the index, -1 if there is not found

Here is the call graph for this function:



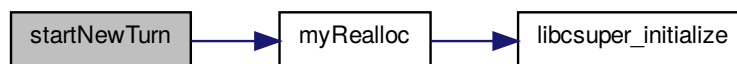
4.4.3.9 void startNewTurn (csuStruct * ptr_csu_struct, int index_player)

Reallocate the memory for the point to begin a new turn.

Parameters

in, out	<i>*ptr_csu_struct</i>	a pointer on a csuStruct
in, out	<i>index_player</i>	the index of the player who begin a new turn, -1 if everybody begin a new turn

Here is the call graph for this function:



4.5 file_system_path.c File Reference

Fonctions qui l'emrankment des fichiers sauvegardes.

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

Functions

- int [createFileSystemPath](#) ()
- int [readFileSystemPath](#) (char *file_name)
- int [readSystemPath](#) (char *file_name)
- int [changeSystemPath](#) (char *new_path)

- void `readHomePath` (char *path)
- void `readHomePathSlash` (char *path)

4.5.1 Detailed Description

Fonctions qui l'emrankment des fichiers sauvegardes.

Author

Remi BERTHO

Date

13/02/14

Version

2.0

4.5.2 Function Documentation

4.5.2.1 int `changeSystemPath` (char * *new_path*)

Change the system path

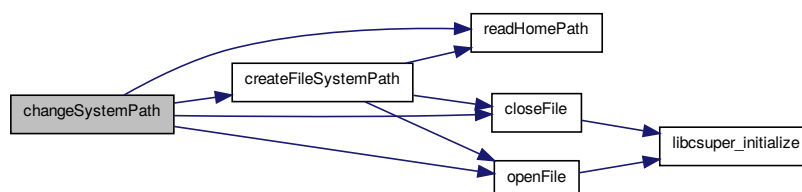
Parameters

<code>in, out</code>	<code>*new_path</code>	le nomveau chemin
----------------------	------------------------	-------------------

Returns

MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:



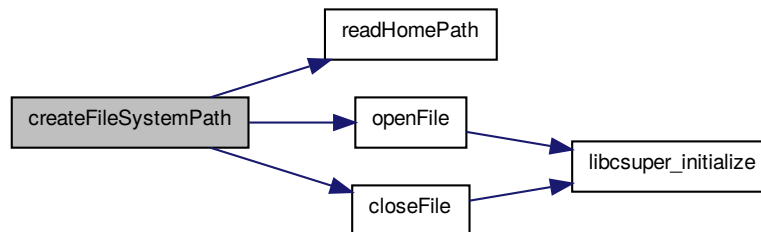
4.5.2.2 void `createFileSystemPath` ()

Create the folder and the file which contain the system path

Returns

MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:

**4.5.2.3 int readFileSystemPath (char * file_name)**

Read the system path and the path read to the filename

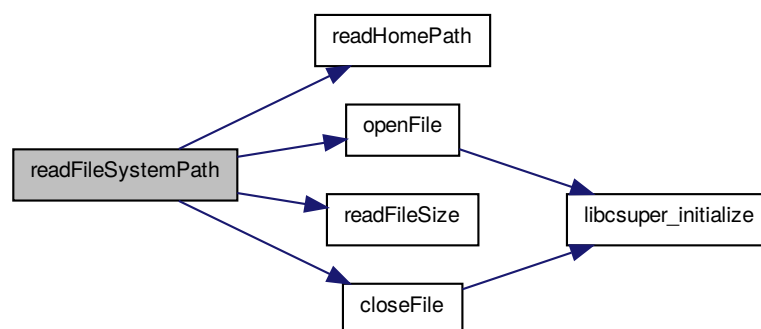
Parameters

in, out	*file_name	the filename
---------	------------	--------------

Returns

MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:

**4.5.2.4 void readHomePath (char * path)**

Read the home path

Parameters

<i>in, out</i>	<i>path</i>	the path
----------------	-------------	----------

Read the home path with a slash at the end

Parameters

<i>in, out</i>	<i>path</i>	the path
----------------	-------------	----------

4.5.2.5 void readHomePathSlash (char * *path*)

4.5.2.6 int readSystemPath (char * *file_name*)

Add the system path, if the file system path doesn't exist, it create it.

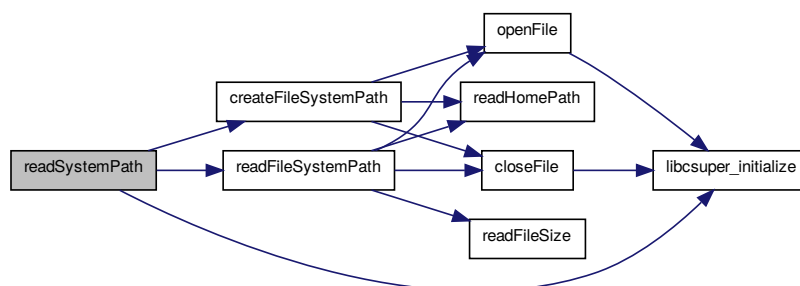
Parameters

<i>in, out</i>	* <i>file_name</i>	the filename
----------------	--------------------	--------------

Returns

MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:



4.6 file_system_path.h File Reference

Prototypes des fonctions qui l'emrankment des fichiers sauvegardes.

```
#include <sys/stat.h>
#include <sys/types.h>
#include "csu_struct.h"
#include "csu_files.h"
```

Macros

- `#define FILE_NAME_SYSTEM_PATH "system_path.txt"`
- `#define MAIN_FOLDER_NAME ".csuper"`

Functions

- int [createFileSystemPath](#) ()
- int [readFileSystemPath](#) (char *file_name)
- int [readSystemPath](#) (char *file_name)
- int [changeSystemPath](#) (char *new_path)
- void [readHomePath](#) (char *path)
- void [readHomePathSlash](#) (char *path)

4.6.1 Detailed Description

Prototypes des fonctions qui l'emrankment des fichiers sauvegardes.

Author

Remi BERTHO

Date

16/04/14

Version

2.2.0

4.6.2 Macro Definition Documentation

4.6.2.1 #define FILE_NAME_SYSTEM_PATH "system_path.txt"

Define filename of the file which contain the system path to "system_path.txt"

4.6.2.2 #define MAIN_FOLDER_NAME ".csuper"

Define the folder name of the csuper preferences

4.6.3 Function Documentation

4.6.3.1 int changeSystemPath (char * new_path)

Change the system path

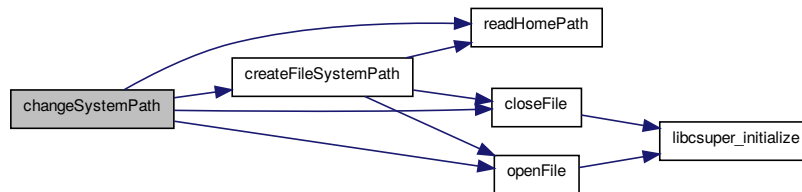
Parameters

in, out	*new_path	le nomveau chemin
---------	-----------	-------------------

Returns

MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:

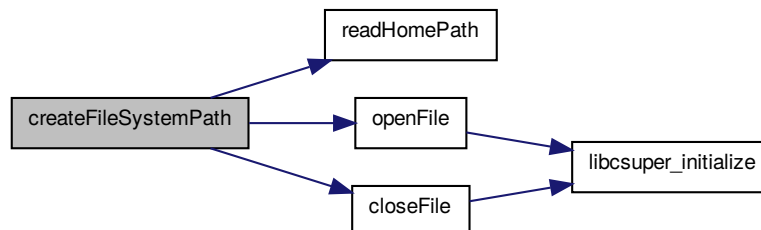
**4.6.3.2 int createFileSystemPath ()**

Create the folder and the file which contain the system path

Returns

MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:

**4.6.3.3 int readFileSystemPath (char * file_name)**

Read the system path and the path read to the filename

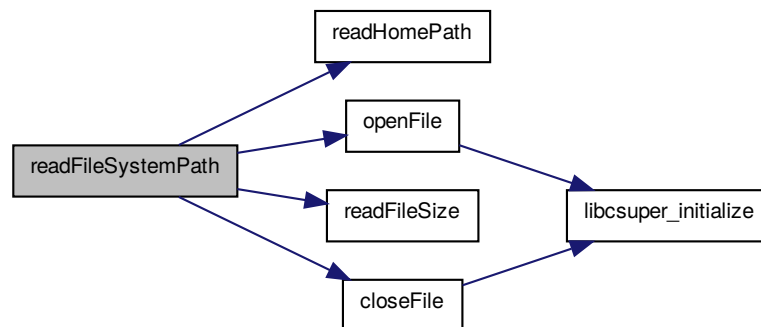
Parameters

in, out	<i>*file_name</i>	the filename
---------	-------------------	--------------

Returns

MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:



4.6.3.4 void readHomePath (char * path)

Read the home path

Parameters

in, out	<i>path</i>	the path
---------	-------------	----------

Read the home path with a slash at the end

Parameters

in, out	<i>path</i>	the path
---------	-------------	----------

4.6.3.5 void readHomePathSlash (char * path)

4.6.3.6 int readSystemPath (char * file_name)

Add the system path, if the file system path doesn't exist, it create it.

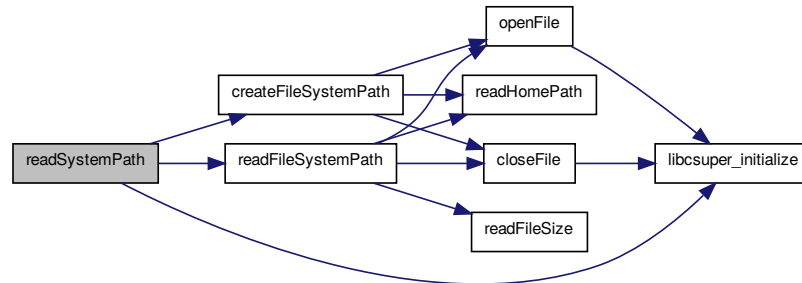
Parameters

in, out	<i>*file_name</i>	the filename
---------	-------------------	--------------

Returns

MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:



4.7 game_config.c File Reference

Game configuration.

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

Functions

- [list_game_config](#) * newListGameConfig (int nb_config)
- void closeListGameConfig (list_game_config *ptr_list_config)
- int makeConfigListFile (char *home_path)
- [list_game_config](#) * readConfigListFile (char *home_path)
- int addConfigListFile (char *new_config_name, char *home_path)
- int removeConfigListFile (int index_delete, [list_game_config](#) *ptr_list_config, char *home_path)
- int newConfigFile ([game_config](#) config, char *home_path)
- int removeConfigFile (char *config_name, char *home_path)
- int readConfigFile (int index_read, [list_game_config](#) *ptr_list_config, [game_config](#) *ptr_config, char *home_path)
- int exportConfigFile (char *home_path, char *file_name)
- int importConfigFile (char *home_path, char *file_name)

4.7.1 Detailed Description

Game configuration.

Author

Remi BERTHO

Date

29/04/14

Version

2.4.0

4.7.2 Function Documentation

4.7.2.1 int addConfigListFile (char * new_config_name, char * home_path)

Add a new game configuration into the file which contain the list of game configuration.

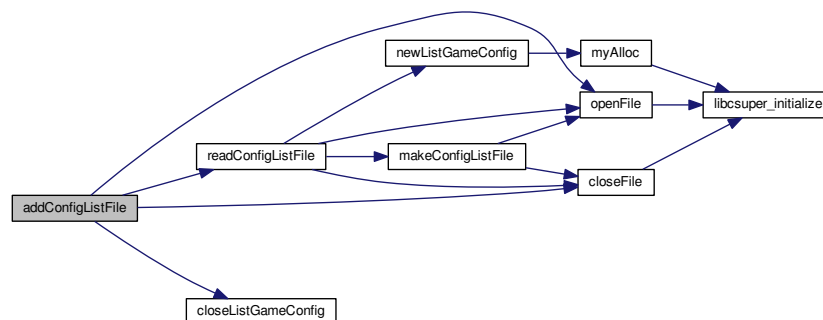
Parameters

in	<i>new_config_name</i>	the name of the new game configuration
in	<i>home_path</i>	the path to the home directory

Returns

MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:



4.7.2.2 void closeListGameConfig (list_game_config * ptr_list_config)

Free a list of game configuration

Parameters

in	<i>*ptr_list_config</i>	a pointer on a list of game configuration
----	-------------------------	---

4.7.2.3 int exportConfigFile (char * home_path, char * file_name)

Export all config file into a file.

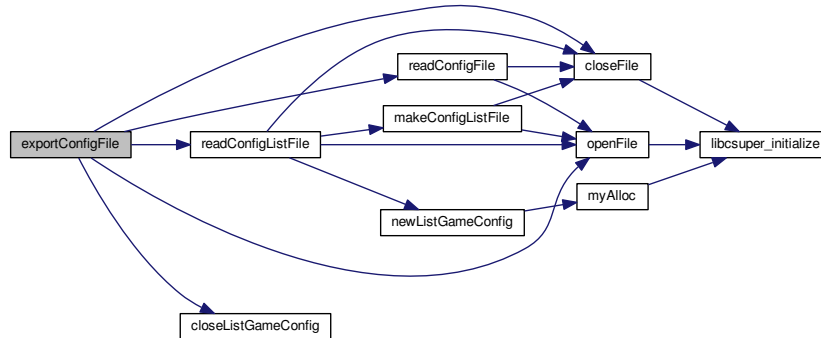
Parameters

in	<i>file_name</i>	the filename of the exported file.
in	<i>home_path</i>	the path to the home directory

Returns

a [list_game_config](#)

Here is the call graph for this function:



4.7.2.4 int importConfigFile (char * *home_path*, char * *file_name*)

Import all config file from a file.

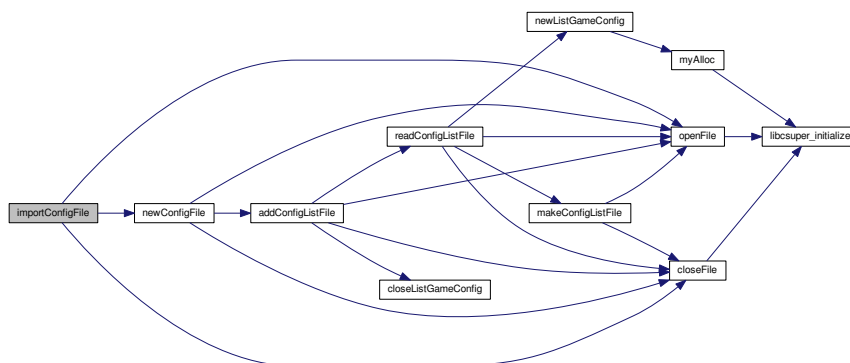
Parameters

in	<i>file_name</i>	the filename of the exported file.
in	<i>home_path</i>	the path to the home directory

Returns

a [list_game_config](#)

Here is the call graph for this function:



4.7.2.5 int makeConfigListFile (char * *home_path*)

Create the folder which contain the games configurations and the files which contain the list of games configurations

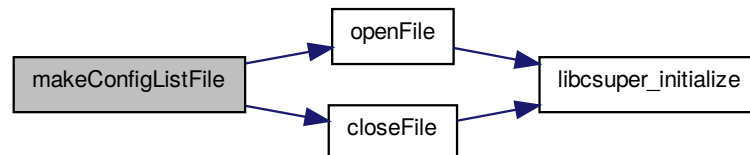
Parameters

in	<i>*home_path</i>	the path to the home directory
----	-------------------	--------------------------------

Returns

MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:



4.7.2.6 int newConfigFile (game_config config, char * home_path)

Create a game configuration file and put it into the game configuration file list.

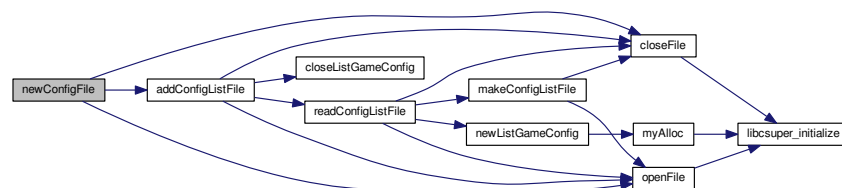
Parameters

in	<i>config</i>	the gale configuration
in	<i>home_path</i>	the path to the home directory

Returns

MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:



4.7.2.7 list_game_config * newListGameConfig (int nb_config)

Create a list of game configuration.

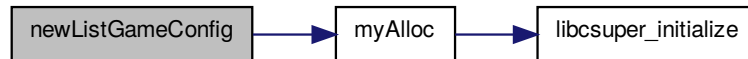
Parameters

in	<i>nb_config</i>	the number of game configuration
----	------------------	----------------------------------

Returns

une [list_game_config](#)

Here is the call graph for this function:



4.7.2.8 `int readConfigFile (int index_read, list_game_config * ptr_list_config, game_config * ptr_config, char * home_path)`

Read a game configuration file.

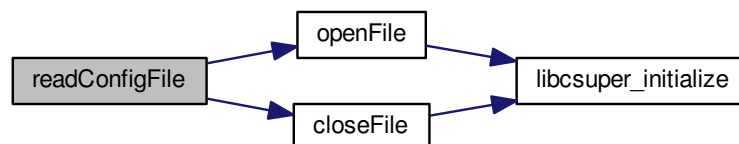
Parameters

in	<i>index_read</i>	the index of the game configuration to be read
in	<i>ptr_list_config</i>	a pointer on the game configuration list
in	<i>ptr_config</i>	a pointer on a game configuration
in	<i>home_path</i>	the path to the home directory

Returns

a [list_game_config](#)

Here is the call graph for this function:



4.7.2.9 `list_game_config * readConfigListFile (char * home_path)`

Read the file which contain the list of game configuration.

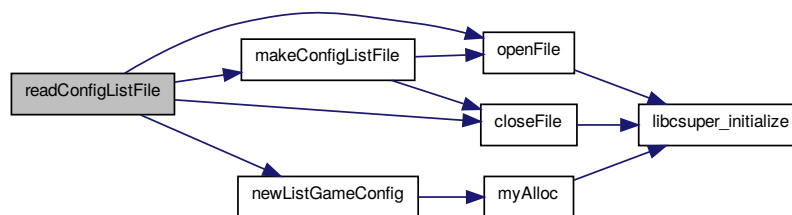
Parameters

in	<i>*home_path</i>	the path to the home directory
----	-------------------	--------------------------------

Returns

a [list_game_config](#)

Here is the call graph for this function:



4.7.2.10 int removeConfigFile (char * config_name, char * home_path)

Delete a game configuration.

Parameters

in	<i>config_name</i>	the name of the game configuration which will be deleted
in	<i>home_path</i>	the path to the home directory

Returns

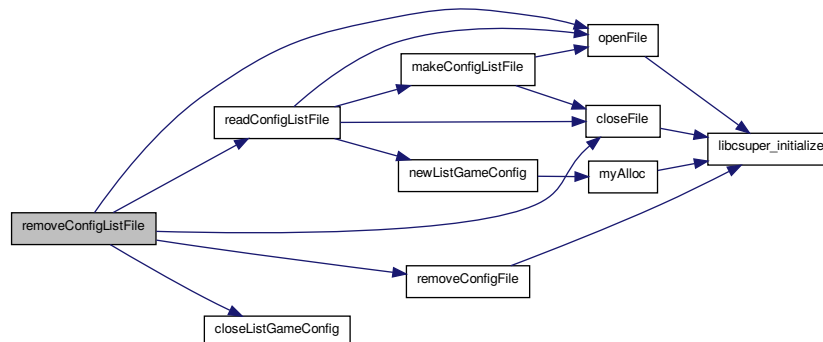
MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:



4.7.2.11 `int removeConfigListFile (int index_delete, list_game_config * ptr_list_config, char * home_path)`

Here is the call graph for this function:



4.8 game_config.h File Reference

Game configurations.

```
#include <math.h>
#include "csu_struct.h"
#include "file_system_path.h"
```

Data Structures

- struct [list_game_config](#)

Macros

- #define [CONFIGURATION_FOLDER_NAME](#) "config"
- #define [CONFIGURATION_FILE_NAME](#) "configuration"
- #define [STRING_CHECK_GAME_CONFIG](#) "Csuper_Game_Configuration"

Functions

- [list_game_config * newListGameConfig](#) (int nb_config)
- void [closeListGameConfig](#) (list_game_config *ptr_list_config)
- int [makeConfigListFile](#) (char *home_path)
- [list_game_config * readConfigListFile](#) (char *home_path)
- int [addConfigListFile](#) (char *new_config_name, char *home_path)
- int [removeConfigListFile](#) (int index_delete, list_game_config *ptr_list_config, char *home_path)
- int [newConfigFile](#) (game_config config, char *home_path)
- int [removeConfigFile](#) (char *config_name, char *home_path)
- int [readConfigFile](#) (int index_read, list_game_config *ptr_list_config, game_config *ptr_config, char *home_path)
- int [exportConfigFile](#) (char *home_path, char *file_name)
- int [importConfigFile](#) (char *home_path, char *file_name)

4.8.1 Detailed Description

Game configurations.

Author

Remi BERTHO

Date

29/04/14

Version

2.4.0

4.8.2 Macro Definition Documentation

4.8.2.1 #define CONFIGURATION_FILE_NAME "configuration"

Define the name of the file which contain the list of the game configurations

4.8.2.2 #define CONFIGURATION_FOLDER_NAME "config"

Define the name of the folder which contain the game configurations

4.8.2.3 #define STRING_CHECK_GAME_CONFIG "Csuper_Game_Configuration"

String for checking if the file is game configuration file.

4.8.3 Function Documentation

4.8.3.1 int addConfigListFile (char * *new_config_name*, char * *home_path*)

Add a new game configuration into the file which contain the list of game configuration.

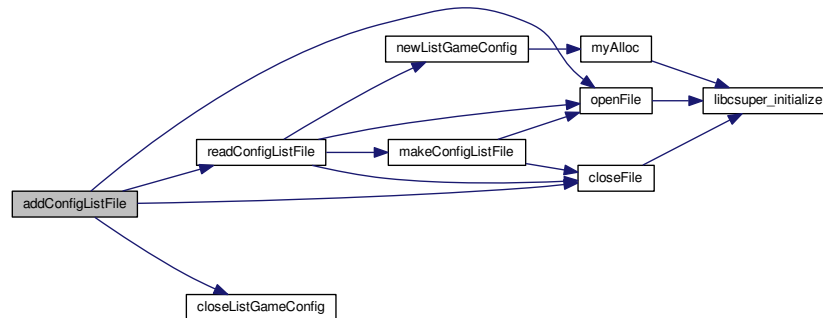
Parameters

in	<i>new_config_name</i>	the name of the new game configuration
in	<i>home_path</i>	the path to the home directory

Returns

MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:



4.8.3.2 void closeListGameConfig (list_game_config * ptr_list_config)

Free a list of game configuration

Parameters

in	<i>*ptr_list_config</i>	a pointer on a list of game configuration
----	-------------------------	---

4.8.3.3 int exportConfigFile (char * home_path, char * file_name)

Export all config file into a file.

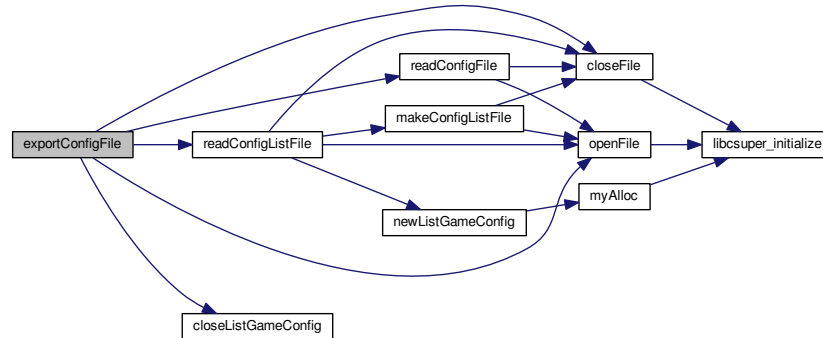
Parameters

in	<i>file_name</i>	the filename of the exported file.
in	<i>home_path</i>	the path to the home directory

Returns

a [list_game_config](#)

Here is the call graph for this function:



4.8.3.4 int importConfigFile (char * *home_path*, char * *file_name*)

Import all config file from a file.

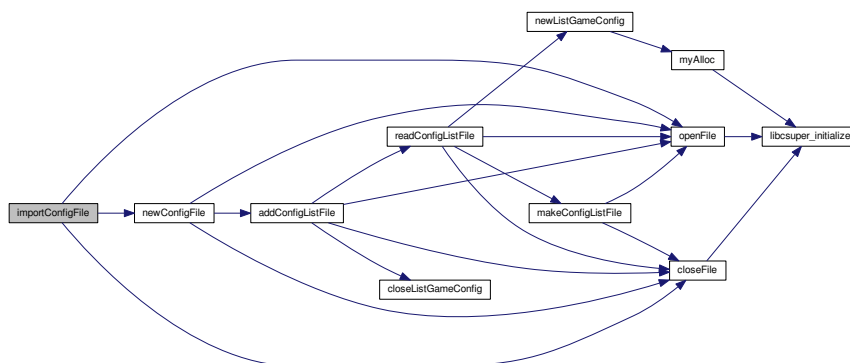
Parameters

in	<i>file_name</i>	the filename of the exported file.
in	<i>home_path</i>	the path to the home directory

Returns

a [list_game_config](#)

Here is the call graph for this function:



4.8.3.5 int makeConfigListFile (char * *home_path*)

Create the folder which contain the games configurations and the files which contain the list of games configurations

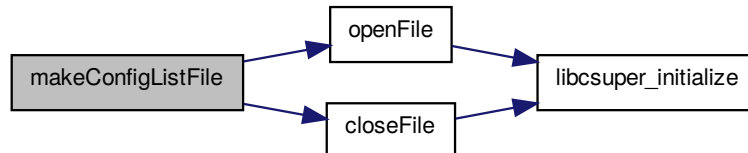
Parameters

in	<i>*home_path</i>	the path to the home directory
----	-------------------	--------------------------------

Returns

MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:



4.8.3.6 int newConfigFile (game_config config, char * home_path)

Create a game configuration file and put it into the game configuration file list.

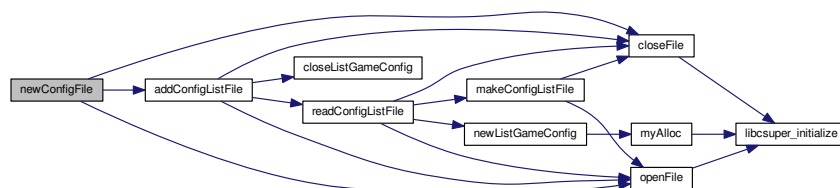
Parameters

in	<i>config</i>	the gale configuration
in	<i>home_path</i>	the path to the home directory

Returns

MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:



4.8.3.7 list_game_config* newListGameConfig (int nb_config)

Create a list of game configuration.

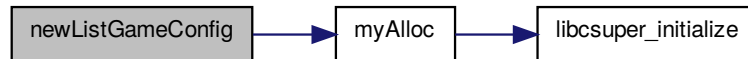
Parameters

in	<i>nb_config</i>	the number of game configuration
----	------------------	----------------------------------

Returns

une [list_game_config](#)

Here is the call graph for this function:



4.8.3.8 `int readConfigFile (int index_read, list_game_config * ptr_list_config, game_config * ptr_config, char * home_path)`

Read a game configuration file.

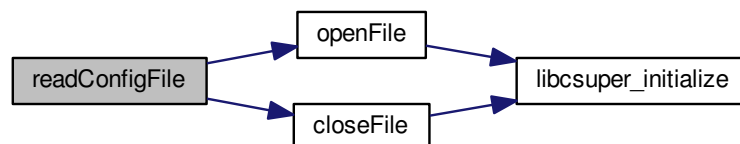
Parameters

in	<i>index_read</i>	the index of the game configuration to be read
in	<i>ptr_list_config</i>	a pointer on the game configuration list
in	<i>ptr_config</i>	a pointer on a game configuration
in	<i>home_path</i>	the path to the home directory

Returns

a [list_game_config](#)

Here is the call graph for this function:



4.8.3.9 `list_game_config* readConfigListFile (char * home_path)`

Read the file which contain the list of game configuration.

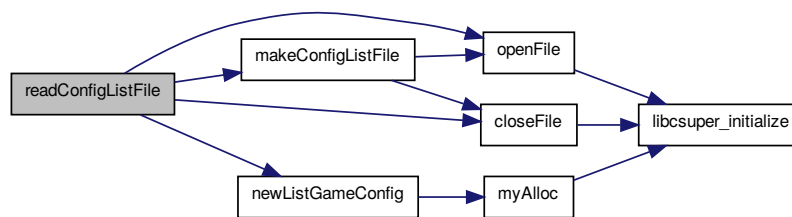
Parameters

in	<i>*home_path</i>	the path to the home directory
----	-------------------	--------------------------------

Returns

a [list_game_config](#)

Here is the call graph for this function:



4.8.3.10 int removeConfigFile (char * config_name, char * home_path)

Delete a game configuration.

Parameters

in	<i>config_name</i>	the name of the game configuration which will be deleted
in	<i>home_path</i>	the path to the home directory

Returns

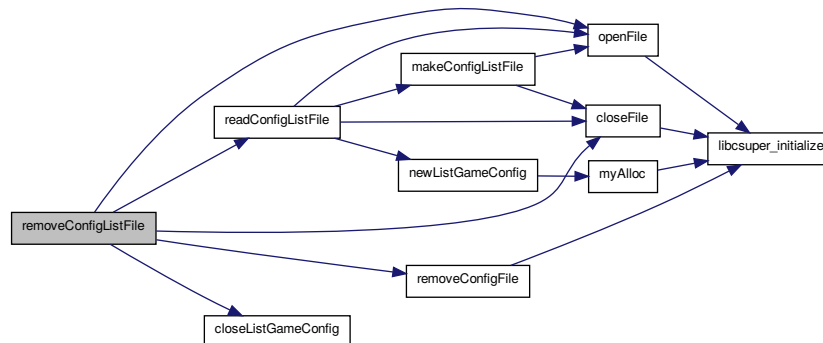
MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:



4.8.3.11 `int removeConfigListFile (int index_delete, list_game_config * ptr_list_config, char * home_path)`

Here is the call graph for this function:



4.9 libcsuper.h File Reference

Inclusion of all header files of libcsuper.

```
#include "csu_struct.h"
#include "share.h"
#include "csu_files.h"
#include "file_system_path.h"
#include "main_argument.h"
#include "game_config.h"
```

4.9.1 Detailed Description

Inclusion of all header files of libcsuper.

Author

Remi BERTHO

Date

05/04/14

Version

2.2.0

4.10 main_argument.c File Reference

Begin csuper.

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

Functions

- int [searchArgument](#) (int argc, char *argv[], int *function, int *file_place)
- void [displayHelp](#) ()

4.10.1 Detailed Description

Begin csuper.

Author

Remi BERTHO

Date

16/04/14

Version

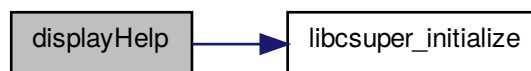
2.2.0

4.10.2 Function Documentation

4.10.2.1 void displayHelp ()

Display the help

Here is the call graph for this function:



4.10.2.2 int searchArgument (int argc, char * argv[], int * function, int * file_place)

Search the argument passed to the main function

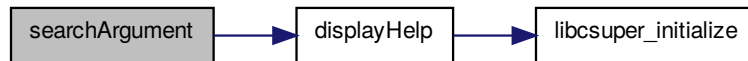
Parameters

in	<i>argc</i>	the number of argument
in	<i>argv</i>	the array of argument
in	<i>function</i>	integer which determine which function run
in	<i>file_place</i>	integer which determine the index of the filename

Returns

MY_TRUE if the function founded an argument, MY_FALSE otherwise

Here is the call graph for this function:



4.11 main_argument.h File Reference

Begin csuper.

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

Macros

- `#define STRING_READ_FILE "--read"`
- `#define STRING_READ_FILE_RED "-r"`
- `#define READ_FILE 0`
- `#define STRING_OPEN_FILE "--open"`
- `#define STRING_OPEN_FILE_RED "-o"`
- `#define OPEN_FILE 1`
- `#define STRING_HELP "--help"`
- `#define STRING_HELP_RED "-h"`
- `#define HELP 2`

Functions

- `int searchArgument (int argc, char *argv[], int *function, int *file_place)`
- `void displayHelp ()`

4.11.1 Detailed Description

Begin csuper.

Author

Remi BERTHO

Date

16/04/14

Version

2.2.0

4.11.2 Macro Definition Documentation

4.11.2.1 `#define HELP 2`

Define the call help to 2

4.11.2.2 `#define OPEN_FILE 1`

Define the call to read a file to 1

4.11.2.3 `#define READ_FILE 0`

Define the call to read a file to 0

4.11.2.4 `#define STRING_HELP "--help"`

Define the argument which call help to "--help"

4.11.2.5 `#define STRING_HELP_RED "-h"`

Define the reduce argument which call help to "-h"

4.11.2.6 `#define STRING_OPEN_FILE "--open"`

Define the argument which call to open a file to "--open"

4.11.2.7 `#define STRING_OPEN_FILE_RED "-o"`

Define the reduce argument which call to open a file to "-o"

4.11.2.8 `#define STRING_READ_FILE "--read"`

Define the argument which call to read a file to "--read"

4.11.2.9 `#define STRING_READ_FILE_RED "-r"`

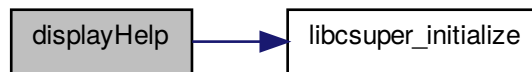
Define the reduce argument which call to read a file to "-r"

4.11.3 Function Documentation

4.11.3.1 `void displayHelp ()`

Display the help

Here is the call graph for this function:



4.11.3.2 int searchArgument (int argc, char * argv[], int * function, int * file_place)

Search the argument passed to the main function

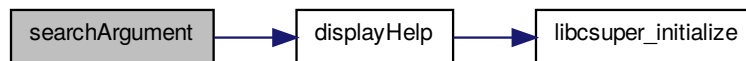
Parameters

in	<i>argc</i>	the number of argument
in	<i>argv</i>	the array of argument
in	<i>function</i>	integer which determine which function run
in	<i>file_place</i>	integer which determine the index of the filename

Returns

MY_TRUE if the function founded an argument, MY_FALSE otherwise

Here is the call graph for this function:



4.12 share.c File Reference

Essential function of libcsuper.

```
#include "share.h"
#include "csu_files.h"
```

Functions

- void [libcsuper_initialize](#) ()
- void [wrongChoice](#) ()
- void [clearScreen](#) ()
- int [compareFloatAscending](#) (void const *a, void const *b)
- int [compareFloatDescending](#) (void const *a, void const *b)
- FILE * [openFile](#) (char file_name[], char mode[])

- int [closeFile](#) (FILE *ptr_file)
- int [readFileSize](#) (FILE *ptr_file)
- void * [myAlloc](#) (int size_alloue)
- void [myRealloc](#) (void **ptr, int size_alloue)
- void [addFileCsuExtension](#) (char *file_name)
- int [deleteFile](#) (char *file_name)
- int [renameFile](#) (char *old_name, char *new_name)

4.12.1 Detailed Description

Essential function of libcsuper.

Author

Remi BERTHO

Date

15/04/14

Version

2.2.0

4.12.2 Function Documentation

4.12.2.1 void [addFileCsuExtension](#) (char * *file_name*)

Add the csu file extension

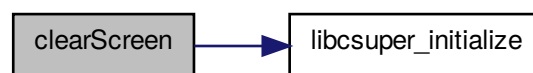
Parameters

in	<i>file_name</i>	the filename
----	------------------	--------------

4.12.2.2 void [clearScreen](#) ()

Clear the terminal.

Here is the call graph for this function:



4.12.2.3 int [closeFile](#) (FILE * *ptr_file*)

Close the file

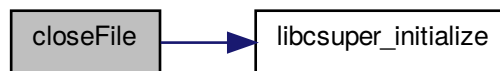
Parameters

in	<i>*ptr_file</i>	the file
----	------------------	----------

Returns

0 if everything is OK, 1 otherwise

Here is the call graph for this function:

4.12.2.4 int compareFloatAscending (void const * *a*, void const * *b*)

Compare 2 float

Parameters

in	<i>*a</i>	a pointer on a float
in	<i>*b</i>	a pointer on a float

Returns

1 if $a > b$, 0 if $a = b$ and -1 if $a < b$

4.12.2.5 int compareFloatDescending (void const * *a*, void const * *b*)

Compare 2 float

Parameters

in	<i>*a</i>	a pointer on a float
in	<i>*b</i>	a pointer on a float

Returns

1 if $a < b$, 0 if $a = b$ and -1 if $a > b$

4.12.2.6 int deleteFile (char * *file_name*)

Delete a file

Parameters

in	<i>*file_name</i>	the filename
----	-------------------	--------------

Returns

MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:

**4.12.2.7 void libcsuper_initialize ()**

Initialize libcsuper with gettext.

4.12.2.8 void * myAlloc (int *size_alloue*)

Allocate a memory block and check if everything is OK.

Parameters

in	<i>size_alloue</i>	the size
----	--------------------	----------

Returns

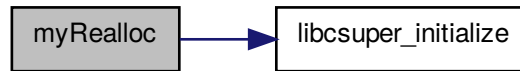
a pointer on the allocate memory block

Here is the call graph for this function:



4.12.2.9 void myRealloc (void ** *ptr*, int *size_alloue*)

Here is the call graph for this function:



4.12.2.10 FILE * openFile (char *file_name*[], char *mode*[])

Open a file with his name and with a specific mode.

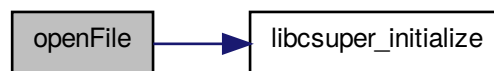
Parameters

in	<i>file_name</i> []	the filename
in	<i>mode</i> []	the mode

Returns

a pointer to the open file, NULL if there was a problem

Here is the call graph for this function:



4.12.2.11 int readFileSize (FILE * *ptr_file*)

Read the size of the file

Parameters

in	<i>*ptr_file</i>	the file
----	------------------	----------

Returns

the size of the file

4.12.2.12 int renameFile (char * *old_name*, char * *new_name*)

Rename a file.

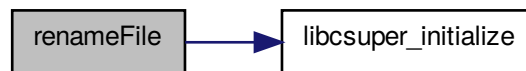
Parameters

in	<i>*old_name</i>	the old name of the file
in	<i>*new_name</i>	the new name of the file

Returns

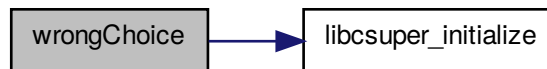
MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:

**4.12.2.13 void wrongChoice ()**

Display an error message.

Here is the call graph for this function:

**4.13 share.h File Reference**

Header for the essential function of libcsuper.

```

#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <string.h>
#include <libintl.h>
  
```

Macros

- `#define MY_TRUE 1`
- `#define MY_FALSE 0`
- `#define _(String) dgettext ("libcsuper", String)`

Functions

- void [libcsuper_initialize](#) ()
- void [wrongChoice](#) ()
- void [clearScreen](#) ()
- int [compareFloatDescending](#) (void const *a, void const *b)
- int [compareFloatAscending](#) (void const *a, void const *b)
- FILE * [openFile](#) (char nome[], char mode[])
- int [closeFile](#) (FILE *ptr_file)
- int [readFileSize](#) (FILE *ptr_file)
- void * [myAlloc](#) (int size_alloue)
- void [myRealloc](#) (void **ptr, int size_alloue)
- void [addFileCsuExtension](#) (char *file_name)
- int [deleteFile](#) (char *file_name)
- int [renameFile](#) (char *old_name, char *new_name)

4.13.1 Detailed Description

Header for the essential function of libcsuper.

Author

Remi BERTHO

Date

15/04/14

Version

2.2.0

4.13.2 Macro Definition Documentation

4.13.2.1 `#define _(String) dgettext ("libcsuper", String)`

Define the `_` for gettext.

4.13.2.2 `#define MY_FALSE 0`

Definit MY_FALSE a 0

4.13.2.3 `#define MY_TRUE 1`

Definit MY_TRUE a 1

4.13.3 Function Documentation

4.13.3.1 `void addFileCsuExtension (char * file_name)`

Add the csu file extension

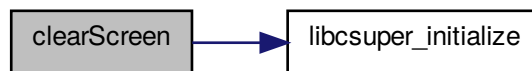
Parameters

in	<i>file_name</i>	the filename
----	------------------	--------------

4.13.3.2 void clearScreen ()

Clear the terminal.

Here is the call graph for this function:



4.13.3.3 int closeFile (FILE * ptr_file)

Close the file

Parameters

in	<i>*ptr_file</i>	the file
----	------------------	----------

Returns

0 if everything is OK, 1 otherwise

Here is the call graph for this function:



4.13.3.4 int compareFloatAscending (void const * a, void const * b)

Compare 2 float

Parameters

in	<i>*a</i>	a pointer on a float
in	<i>*b</i>	a pointer on a float

Returns

1 if $a > b$, 0 if $a = b$ and -1 if $a < b$

4.13.3.5 int compareFloatDescending (void const * *a*, void const * *b*)

Compare 2 float

Parameters

in	<i>*a</i>	a pointer on a float
in	<i>*b</i>	a pointer on a float

Returns

1 if $a < b$, 0 if $a = b$ and -1 if $a > b$

4.13.3.6 int deleteFile (char * *file_name*)

Delete a file

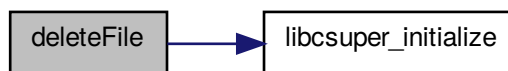
Parameters

in	<i>*file_name</i>	the filename
----	-------------------	--------------

Returns

MY_TRUE if everything is OK, MY_FALSE otherwise

Here is the call graph for this function:

**4.13.3.7 void libcsuper_initialize ()**

Initialize libcsuper with gettext.

4.13.3.8 void* myAlloc (int *size_alloue*)

Allocate a memory block and check if everything is OK.

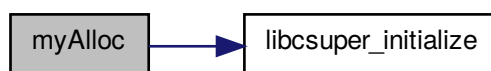
Parameters

in	<i>size_alloue</i>	the size
----	--------------------	----------

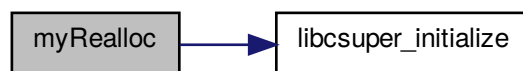
Returns

a pointer on the allocate memory block

Here is the call graph for this function:

**4.13.3.9 void myRealloc (void ** ptr, int size_alloue)**

Here is the call graph for this function:

**4.13.3.10 FILE* openFile (char file_name[], char mode[])**

Open a file with his name and with a specific mode.

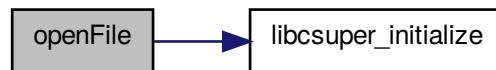
Parameters

in	<i>file_name[]</i>	the filename
in	<i>mode[]</i>	the mode

Returns

a pointer to the open file, NULL if there was a problem

Here is the call graph for this function:

**4.13.3.11 int readFileSize (FILE * ptr_file)**

Read the size of the file

Parameters

in	*ptr_file	the file
----	-----------	----------

Returns

the size of the file

4.13.3.12 int renameFile (char * old_name, char * new_name)

Rename a file.

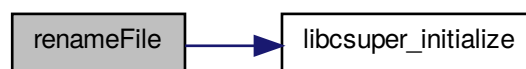
Parameters

in	*old_name	the old name of the file
in	*new_name	the new name of the file

Returns

MY_TRUE if everything is OK, MY_FALSE otherwise

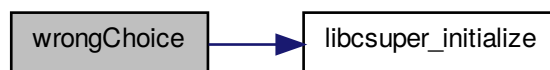
Here is the call graph for this function:



4.13.3.13 void wrongChoice ()

Display an error message.

Here is the call graph for this function:



Index

- - share.h, [53](#)
- addConfigListFile
 - game_config.c, [31](#)
 - game_config.h, [37](#)
- addDistributorCsuStruct
 - csu_struct.c, [15](#)
 - csu_struct.h, [19](#)
- addFileCsuExtension
 - share.c, [48](#)
 - share.h, [53](#)
- begin_score
 - game_config, [7](#)
- changeSystemPath
 - file_system_path.c, [24](#)
 - file_system_path.h, [27](#)
- clearScreen
 - share.c, [48](#)
 - share.h, [54](#)
- closeCsuStruct
 - csu_struct.c, [15](#)
 - csu_struct.h, [20](#)
- closeFile
 - share.c, [48](#)
 - share.h, [54](#)
- closeListGroupConfig
 - game_config.c, [31](#)
 - game_config.h, [38](#)
- compareFloatAscending
 - share.c, [49](#)
 - share.h, [54](#)
- compareFloatDescending
 - share.c, [49](#)
 - share.h, [55](#)
- config
 - csuStruct, [5](#)
- createFileSystemPath
 - file_system_path.c, [24](#)
 - file_system_path.h, [28](#)
- csu_files.c, [9](#)
 - openFileCsuExtension, [9](#)
 - readCsuFile, [10](#)
 - writeCsuFile, [10](#)
 - writeFileNewTurn, [11](#)
- csu_files.h, [11](#)
 - FILE_EXTENSION, [12](#)
 - openFileCsuExtension, [12](#)
 - readCsuFile, [13](#)
 - writeCsuFile, [13](#)
 - writeFileNewTurn, [14](#)
- csu_struct.c, [14](#)
 - addDistributorCsuStruct, [15](#)
 - closeCsuStruct, [15](#)
 - endNewTurn, [16](#)
 - exceedMaxNumber, [16](#)
 - maxNbTurn, [16](#)
 - newCsuStruct, [16](#)
 - rankCalculation, [17](#)
 - searchPlayerIndex, [17](#)
 - startNewTurn, [18](#)
- csu_struct.h, [18](#)
 - addDistributorCsuStruct, [19](#)
 - closeCsuStruct, [20](#)
 - endNewTurn, [20](#)
 - exceedMaxNumber, [20](#)
 - maxNbTurn, [20](#)
 - newCsuStruct, [22](#)
 - rankCalculation, [22](#)
 - SIZE_MAX_NAME, [19](#)
 - searchPlayerIndex, [22](#)
 - startNewTurn, [23](#)
 - VERSION, [19](#)
- csuStruct, [5](#)
 - config, [5](#)
 - day, [5](#)
 - distributor, [5](#)
 - month, [6](#)
 - nb_player, [6](#)
 - nb_turn, [6](#)
 - player_names, [6](#)
 - point, [6](#)
 - rank, [6](#)
 - size_max_name, [6](#)
 - total_points, [6](#)
 - version, [6](#)
 - year, [6](#)
- day
 - csuStruct, [5](#)
- decimal_place
 - game_config, [7](#)
- deleteFile
 - share.c, [49](#)
 - share.h, [55](#)
- displayHelp
 - main_argument.c, [44](#)
 - main_argument.h, [46](#)

- distributor
 - csuStruct, 5
- endNewTurn
 - csu_struct.c, 16
 - csu_struct.h, 20
- exceedMaxNumber
 - csu_struct.c, 16
 - csu_struct.h, 20
- exportConfigFile
 - game_config.c, 31
 - game_config.h, 38
- FILE_EXTENSION
 - csu_files.h, 12
- file_system_path.c, 23
 - changeSystemPath, 24
 - createFileSystemPath, 24
 - readFileSystemPath, 25
 - readHomePath, 25
 - readHomePathSlash, 26
 - readSystemPath, 26
- file_system_path.h, 26
 - changeSystemPath, 27
 - createFileSystemPath, 28
 - readFileSystemPath, 28
 - readHomePath, 29
 - readHomePathSlash, 29
 - readSystemPath, 29
- first_way
 - game_config, 7
- game_config, 7
 - begin_score, 7
 - decimal_place, 7
 - first_way, 7
 - max, 7
 - name, 7
 - nb_max, 7
 - turn_by_turn, 7
 - use_distributor, 7
- game_config.c, 30
 - addConfigListFile, 31
 - closeListGameConfig, 31
 - exportConfigFile, 31
 - importConfigFile, 32
 - makeConfigListFile, 32
 - newConfigFile, 33
 - newListGameConfig, 33
 - readConfigFile, 34
 - readConfigListFile, 34
 - removeConfigFile, 35
 - removeConfigListFile, 35
- game_config.h, 36
 - addConfigListFile, 37
 - closeListGameConfig, 38
 - exportConfigFile, 38
 - importConfigFile, 39
 - makeConfigListFile, 39
 - newConfigFile, 40
 - newListGameConfig, 40
 - readConfigFile, 41
 - readConfigListFile, 41
 - removeConfigFile, 42
 - removeConfigListFile, 42
- HELP
 - main_argument.h, 46
- importConfigFile
 - game_config.c, 32
 - game_config.h, 39
- libcsuper.h, 43
 - libcsuper_initialize
 - share.c, 50
 - share.h, 55
- list_game_config, 8
 - name_game_config, 8
 - nb_config, 8
- MAIN_FOLDER_NAME
 - file_system_path.h, 27
- MY_FALSE
 - share.h, 53
- MY_TRUE
 - share.h, 53
- main_argument.c, 43
 - displayHelp, 44
 - searchArgument, 44
- main_argument.h, 45
 - displayHelp, 46
 - HELP, 46
 - OPEN_FILE, 46
 - READ_FILE, 46
 - STRING_HELP, 46
 - STRING_HELP_RED, 46
 - STRING_OPEN_FILE, 46
 - STRING_READ_FILE, 46
 - searchArgument, 47
- makeConfigListFile
 - game_config.c, 32
 - game_config.h, 39
- max
 - game_config, 7
- maxNbTurn
 - csu_struct.c, 16
 - csu_struct.h, 20
- month
 - csuStruct, 6
- myAlloc
 - share.c, 50
 - share.h, 55
- myRealloc
 - share.c, 50
 - share.h, 56
- name

- game_config, 7
- name_game_config
 - list_game_config, 8
- nb_config
 - list_game_config, 8
- nb_max
 - game_config, 7
- nb_player
 - csuStruct, 6
- nb_turn
 - csuStruct, 6
- newConfigFile
 - game_config.c, 33
 - game_config.h, 40
- newCsuStruct
 - csu_struct.c, 16
 - csu_struct.h, 22
- newListGameConfig
 - game_config.c, 33
 - game_config.h, 40
- OPEN_FILE
 - main_argument.h, 46
- openFile
 - share.c, 51
 - share.h, 56
- openFileCsuExtension
 - csu_files.c, 9
 - csu_files.h, 12
- player_names
 - csuStruct, 6
- point
 - csuStruct, 6
- READ_FILE
 - main_argument.h, 46
- rank
 - csuStruct, 6
- rankCalculation
 - csu_struct.c, 17
 - csu_struct.h, 22
- readConfigFile
 - game_config.c, 34
 - game_config.h, 41
- readConfigListFile
 - game_config.c, 34
 - game_config.h, 41
- readCsuFile
 - csu_files.c, 10
 - csu_files.h, 13
- readFileSize
 - share.c, 51
 - share.h, 57
- readFileSystemPath
 - file_system_path.c, 25
 - file_system_path.h, 28
- readHomePath
 - file_system_path.c, 25
- readHomePathSlash
 - file_system_path.c, 26
 - file_system_path.h, 29
- readSystemPath
 - file_system_path.c, 26
 - file_system_path.h, 29
- removeConfigFile
 - game_config.c, 35
 - game_config.h, 42
- removeConfigListFile
 - game_config.c, 35
 - game_config.h, 42
- renameFile
 - share.c, 51
 - share.h, 57
- SIZE_MAX_NAME
 - csu_struct.h, 19
- STRING_HELP
 - main_argument.h, 46
- STRING_HELP_RED
 - main_argument.h, 46
- STRING_OPEN_FILE
 - main_argument.h, 46
- STRING_READ_FILE
 - main_argument.h, 46
- searchArgument
 - main_argument.c, 44
 - main_argument.h, 47
- searchPlayerIndex
 - csu_struct.c, 17
 - csu_struct.h, 22
- share.c, 47
 - addFileCsuExtension, 48
 - clearScreen, 48
 - closeFile, 48
 - compareFloatAscending, 49
 - compareFloatDescending, 49
 - deleteFile, 49
 - libcsuper_initialize, 50
 - myAlloc, 50
 - myRealloc, 50
 - openFile, 51
 - readFileSize, 51
 - renameFile, 51
 - wrongChoice, 52
- share.h, 52
 - _, 53
 - addFileCsuExtension, 53
 - clearScreen, 54
 - closeFile, 54
 - compareFloatAscending, 54
 - compareFloatDescending, 55
 - deleteFile, 55
 - libcsuper_initialize, 55
 - MY_FALSE, 53
 - MY_TRUE, 53
 - myAlloc, 55

- myRealloc, [56](#)
 - openFile, [56](#)
 - readFileSize, [57](#)
 - renameFile, [57](#)
 - wrongChoice, [57](#)
- size_max_name
 - csuStruct, [6](#)
- startNewTurn
 - csu_struct.c, [18](#)
 - csu_struct.h, [23](#)
- total_points
 - csuStruct, [6](#)
- turn_by_turn
 - game_config, [7](#)
- use_distributor
 - game_config, [7](#)
- VERSION
 - csu_struct.h, [19](#)
- version
 - csuStruct, [6](#)
- writeCsuFile
 - csu_files.c, [10](#)
 - csu_files.h, [13](#)
- writeFileNewTurn
 - csu_files.c, [11](#)
 - csu_files.h, [14](#)
- wrongChoice
 - share.c, [52](#)
 - share.h, [57](#)
- year
 - csuStruct, [6](#)