

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

2.1.9

Generated by Doxygen 1.8.6

Thu Apr 17 2014 09:14:25

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	first_way	7
3.2.2.3	max	7
3.2.2.4	name	7
3.2.2.5	nb_max	7
3.2.2.6	number_after_comma	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	deleteCsuFile	9
4.1.2.2	openFileCsuExtension	10
4.1.2.3	readCsuFile	10
4.1.2.4	renameCsuFile	11
4.1.2.5	writeCsuFile	11
4.1.2.6	writeFileNewTurn	12
4.2	csu_files.h File Reference	12
4.2.1	Detailed Description	13
4.2.2	Macro Definition Documentation	13
4.2.2.1	FILE_EXTENSION	13
4.2.2.2	SIZE_MAX_FILE_NAME	13
4.2.2.3	STRING_CHECK_CSU_FILE	13
4.2.3	Function Documentation	13
4.2.3.1	deleteCsuFile	13
4.2.3.2	openFileCsuExtension	14
4.2.3.3	readCsuFile	14
4.2.3.4	renameCsuFile	15
4.2.3.5	writeCsuFile	15
4.2.3.6	writeFileNewTurn	16
4.3	csu_struct.c File Reference	16
4.3.1	Detailed Description	17
4.3.2	Function Documentation	17
4.3.2.1	addDistributorCsuStruct	17
4.3.2.2	closeCsuStruct	17
4.3.2.3	endNewTurn	18
4.3.2.4	exceedMaxNumber	18
4.3.2.5	maxNbTurn	18
4.3.2.6	newCsuStruct	18
4.3.2.7	rankCalculation	19

4.3.2.8	searchPlayerIndex	19
4.3.2.9	startNewTurn	20
4.4	csu_struct.h File Reference	20
4.4.1	Detailed Description	21
4.4.2	Macro Definition Documentation	21
4.4.2.1	SIZE_MAX_NAME	21
4.4.2.2	VERSION	21
4.4.3	Function Documentation	21
4.4.3.1	addDistributorCsuStruct	21
4.4.3.2	closeCsuStruct	22
4.4.3.3	endNewTurn	22
4.4.3.4	exceedMaxNumber	22
4.4.3.5	maxNbTurn	23
4.4.3.6	newCsuStruct	24
4.4.3.7	rankCalculation	24
4.4.3.8	searchPlayerIndex	24
4.4.3.9	startNewTurn	25
4.5	file_system_path.c File Reference	25
4.5.1	Detailed Description	26
4.5.2	Function Documentation	26
4.5.2.1	changeSystemPath	26
4.5.2.2	createFileSystemPath	26
4.5.2.3	readFileSystemPath	27
4.5.2.4	readHomePath	27
4.5.2.5	readHomePathSlash	28
4.5.2.6	readSystemPath	28
4.6	file_system_path.h File Reference	28
4.6.1	Detailed Description	29
4.6.2	Macro Definition Documentation	29
4.6.2.1	FILE_NAME_SYSTEM_PATH	29
4.6.2.2	MAIN_FOLDER_NAME	29
4.6.3	Function Documentation	29
4.6.3.1	changeSystemPath	29
4.6.3.2	createFileSystemPath	30
4.6.3.3	readFileSystemPath	30
4.6.3.4	readHomePath	31
4.6.3.5	readHomePathSlash	31
4.6.3.6	readSystemPath	31
4.7	game_config.c File Reference	32
4.7.1	Detailed Description	32

4.7.2	Function Documentation	33
4.7.2.1	addConfigListFile	33
4.7.2.2	closeListGameConfig	33
4.7.2.3	makeConfigListFile	33
4.7.2.4	newConfigFile	34
4.7.2.5	newListGameConfig	34
4.7.2.6	readConfigFile	35
4.7.2.7	readConfigListFile	35
4.7.2.8	removeConfigFile	36
4.7.2.9	removeConfigListFile	37
4.8	game_config.h File Reference	37
4.8.1	Detailed Description	38
4.8.2	Macro Definition Documentation	38
4.8.2.1	CONFIGURATION_FILE_NAME	38
4.8.2.2	CONFIGURATION_FOLDER_NAME	38
4.8.3	Function Documentation	38
4.8.3.1	addConfigListFile	38
4.8.3.2	closeListGameConfig	39
4.8.3.3	makeConfigListFile	39
4.8.3.4	newConfigFile	39
4.8.3.5	newListGameConfig	40
4.8.3.6	readConfigFile	40
4.8.3.7	readConfigListFile	41
4.8.3.8	removeConfigFile	42
4.8.3.9	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	libcsuper_initialize	49
4.12.2.7	myAlloc	49
4.12.2.8	myRealloc	50
4.12.2.9	openFile	50
4.12.2.10	readFileSize	51
4.12.2.11	wrongChoice	51
4.13	share.h File Reference	51
4.13.1	Detailed Description	52
4.13.2	Macro Definition Documentation	52
4.13.2.1	_	52
4.13.2.2	FALSE	52
4.13.2.3	TRUE	52
4.13.3	Function Documentation	52
4.13.3.1	addFileCsuExtension	52
4.13.3.2	clearScreen	53
4.13.3.3	closeFile	53
4.13.3.4	compareFloatAscending	53
4.13.3.5	compareFloatDescending	54
4.13.3.6	libcsuper_initialize	54
4.13.3.7	myAlloc	54
4.13.3.8	myRealloc	55
4.13.3.9	openFile	55
4.13.3.10	readFileSize	55
4.13.3.11	wrongChoice	55

Index	57
-----------------------	----

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	12
csu_struct.c	Management of the csu files	16
csu_struct.h	Management of the csu files header	20
file_system_path.c	Fonctions qui l'emrankment des fichiers sauvegardes	25
file_system_path.h	Prototypes des fonctions qui l'emrankment des fichiers sauvegardes	28
game_config.c	Game configuration	32
game_config.h	Game configurations	37
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	51

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 [number_after_comma](#)
- 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 [first_way](#)

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

3.2.2.3 char [max](#)

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

3.2.2.4 char [name](#)[[SIZE_MAX_NAME](#)]

The name of the game configuration

3.2.2.5 float [nb_max](#)

Number maximum or minimum that can reach a player.

3.2.2.6 char [number_after_comma](#)

The number of digit which are display

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)
- int [deleteCsuFile](#) (char *file_name)
- int [renameCsuFile](#) (char *old_name, char *new_name)

4.1.1 Detailed Description

Files management.

Author

Remi BERTHO

Date

16/04/14

Version

2.2.0

4.1.2 Function Documentation

4.1.2.1 int deleteCsuFile (char * file_name)

Delete a csu file

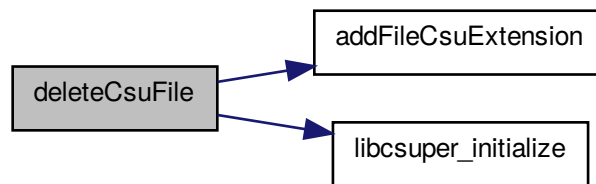
Parameters

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

Returns

TRUE if everything is OK, FALSE otherwise

Here is the call graph for this function:

**4.1.2.2 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.

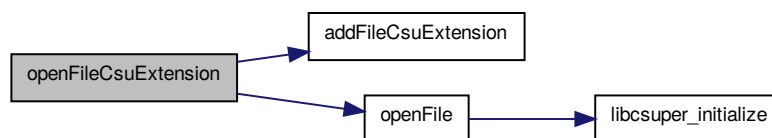
Parameters

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:

**4.1.2.3 csuStruct * readCsuFile (char * file_name)**

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

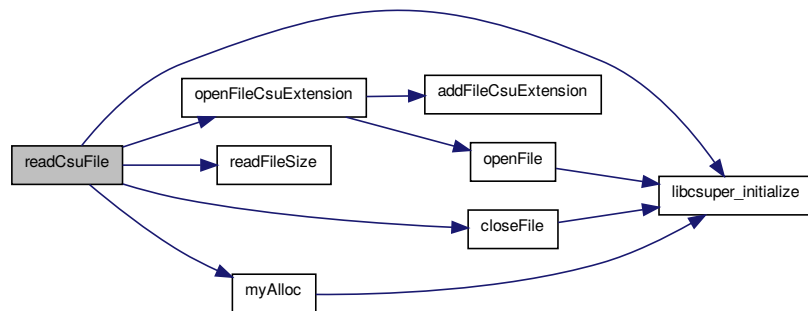
Parameters

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:



4.1.2.4 int renameCsuFile (char * old_name, char * new_name)

Rename a csu file.

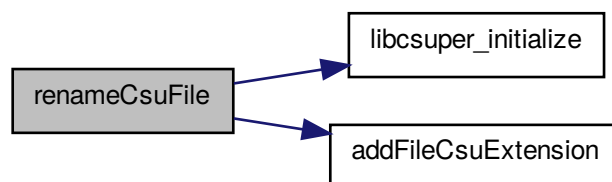
Parameters

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

Returns

TRUE if everything is OK, FALSE otherwise

Here is the call graph for this function:



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

Write a csu file

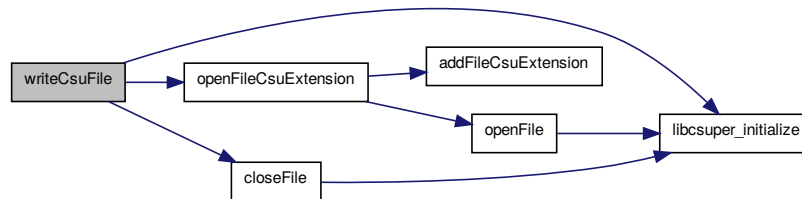
Parameters

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

Returns

TRUE if everything is OK, FALSE otherwise

Here is the call graph for this function:



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

Update the file with the new scores

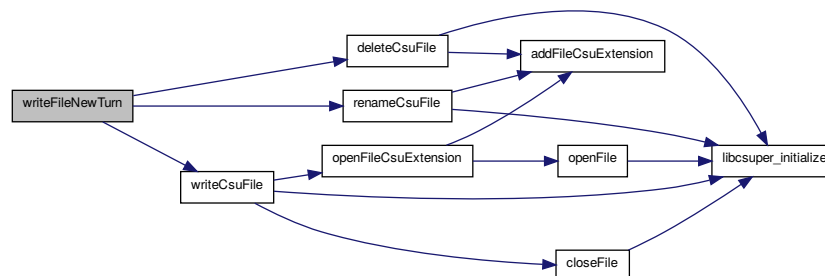
Parameters

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

Returns

TRUE if everything is OK, 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)`
- `int deleteCsuFile (char *file_name)`
- `int renameCsuFile (char *old_name, char *new_name)`

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 `int deleteCsuFile (char * file_name)`

Delete a csu file

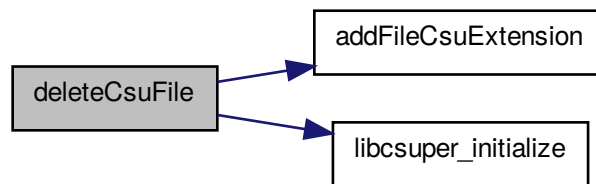
Parameters

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

Returns

TRUE if everything is OK, FALSE otherwise

Here is the call graph for this function:

**4.2.3.2 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.

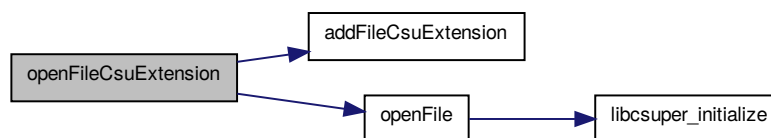
Parameters

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:

**4.2.3.3 csuStruct* readCsuFile (char * file_name)**

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

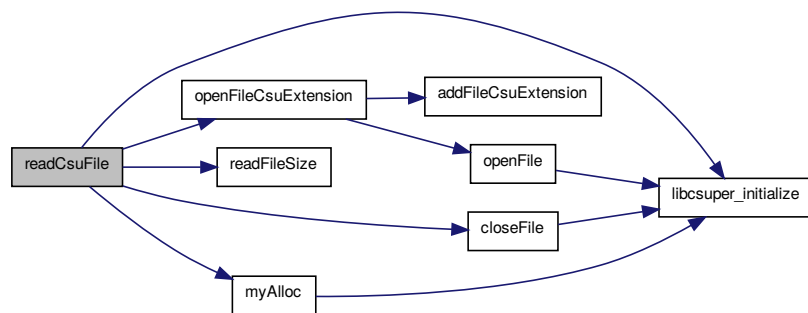
Parameters

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:



4.2.3.4 int renameCsuFile (char * old_name, char * new_name)

Rename a csu file.

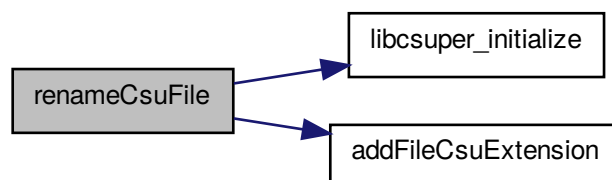
Parameters

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

Returns

TRUE if everything is OK, FALSE otherwise

Here is the call graph for this function:



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

Write a csu file

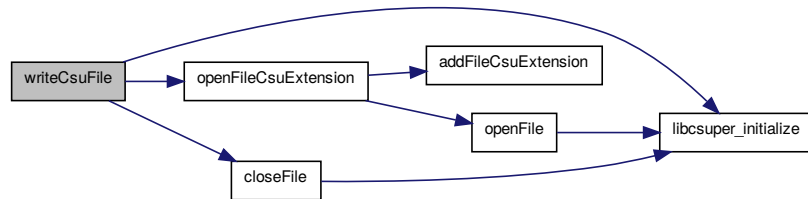
Parameters

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

Returns

TRUE if everything is OK, FALSE otherwise

Here is the call graph for this function:



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

Update the file with the new scores

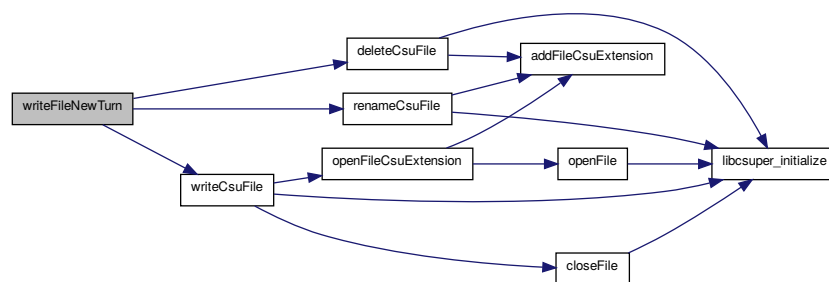
Parameters

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

Returns

TRUE if everything is OK, 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

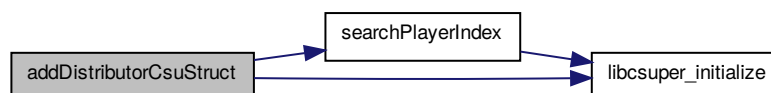
4.3.2.1 void [addDistributorCsuStruct](#) ([csuStruct](#) * *ptr_csu_struct*, char * *distributor_name*)

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
---------	------------------------	--

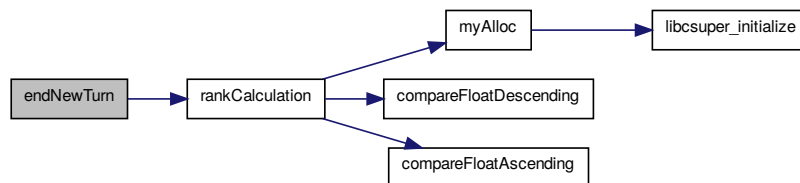
4.3.2.3 void endNewTurn (csuStruct * ptr_csu_struct, int index_player)

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

TRUE if someone exceed, 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

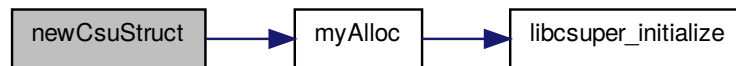
4.3.2.6 csuStruct * newCsuStruct (float nb_player, game_config config)

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:



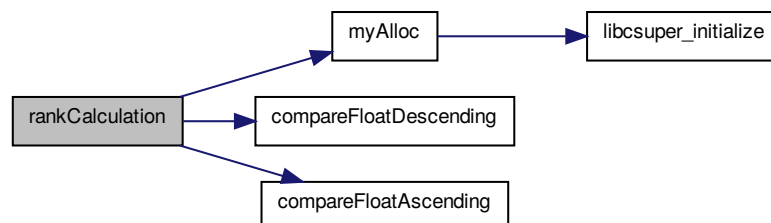
4.3.2.7 void rankCalculation (csuStruct * ptr_csu_struct)

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:



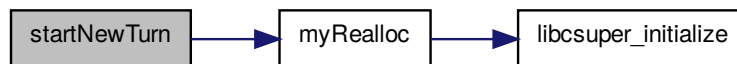
4.3.2.9 void startNewTurn (csuStruct * ptr_csu_struct, int index_player)

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

Parameters

in, out	*ptr_csu_struct	a pointer on a csuStruct
in, out	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 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

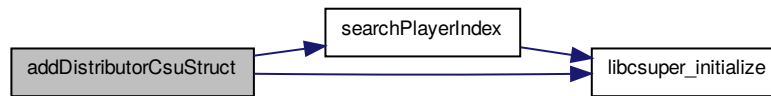
4.4.3.1 void addDistributorCsuStruct (csuStruct * ptr_csu_struct, char * distributor_name)

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
---------	-----------------	--

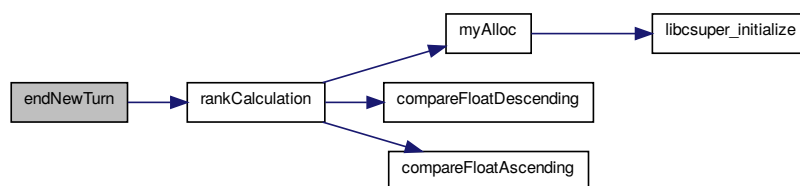
4.4.3.3 void endNewTurn (csuStruct * ptr_csu_struct, int index_player)

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

TRUE if someone exceed, 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

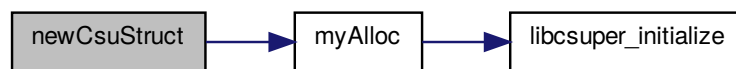
4.4.3.6 `csuStruct* newCsuStruct (float nb_player, game_config config)`

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:



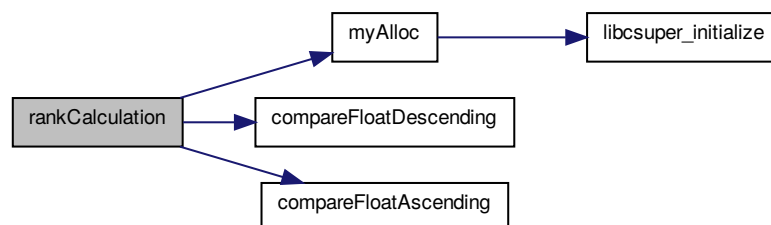
4.4.3.7 `void rankCalculation (csuStruct * ptr_csu_struct)`

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:



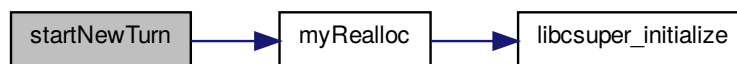
4.4.3.9 void startNewTurn (csuStruct * ptr_csu_struct, int index_player)

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

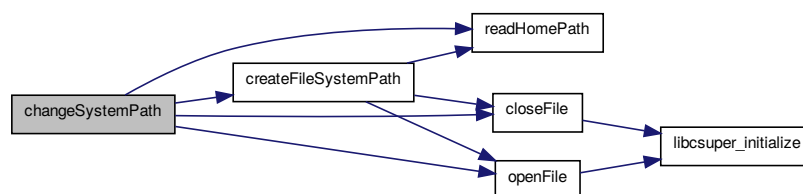
Parameters

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

Returns

TRUE if everything is OK, FALSE otherwise

Here is the call graph for this function:



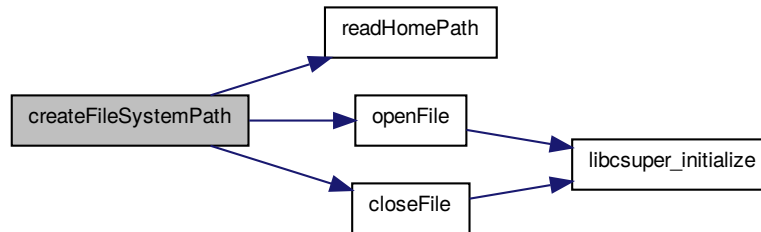
4.5.2.2 void createFileSystemPath ()

Create the folder and the file which contain the system path

Returns

TRUE if everything is OK, 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

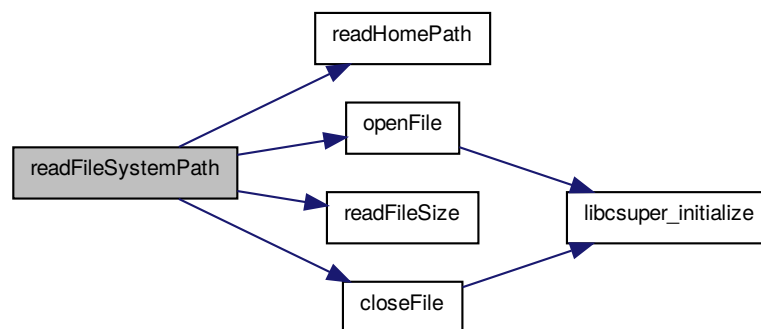
Parameters

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

Returns

TRUE if everything is OK, 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.

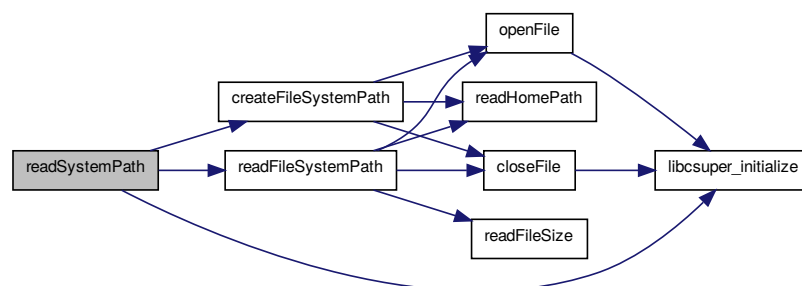
Parameters

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

Returns

TRUE if everything is OK, 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

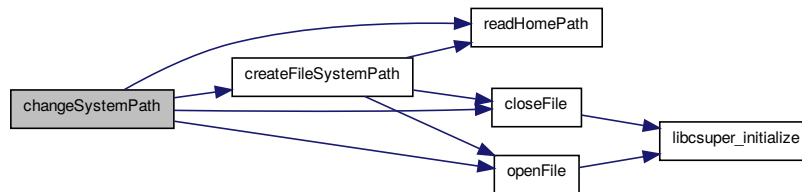
Parameters

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

Returns

TRUE if everything is OK, FALSE otherwise

Here is the call graph for this function:

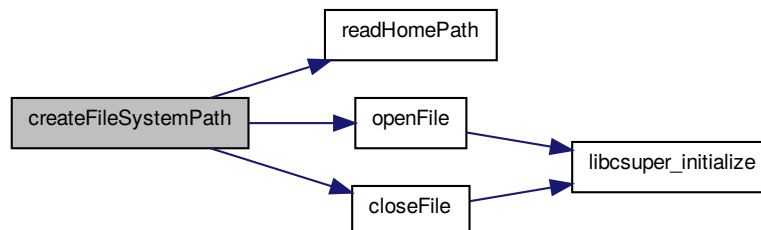
**4.6.3.2 int createFileSystemPath ()**

Create the folder and the file which contain the system path

Returns

TRUE if everything is OK, 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

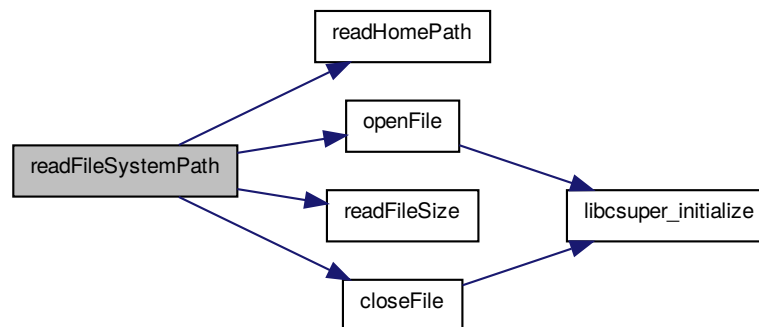
Parameters

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

Returns

TRUE if everything is OK, 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.

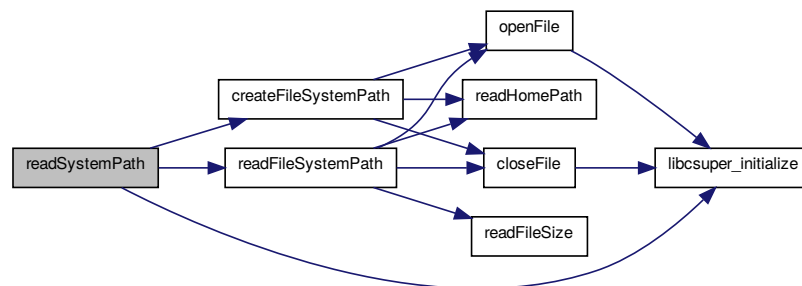
Parameters

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

Returns

TRUE if everything is OK, 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)

4.7.1 Detailed Description

Game configuration.

Author

Remi BERTHO

Date

16/04/14

Version

2.2.0

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.

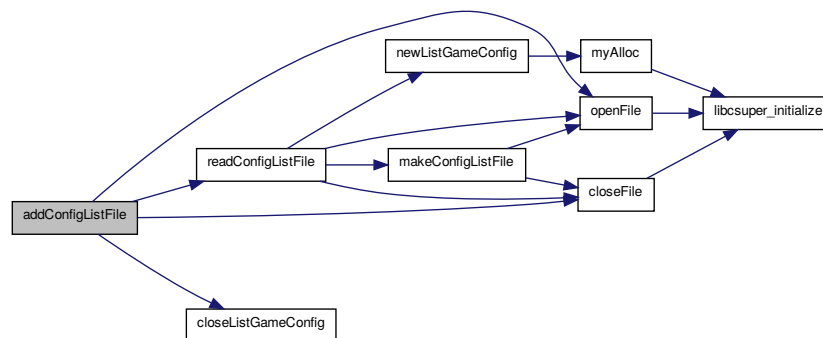
Parameters

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

TRUE if everything is OK, 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 makeConfigListFile (char * home_path)

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

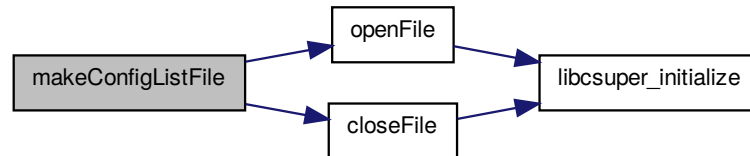
Parameters

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

Returns

TRUE if everything is OK, FALSE otherwise

Here is the call graph for this function:



4.7.2.4 int newConfigFile (game_config config, char * home_path)

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

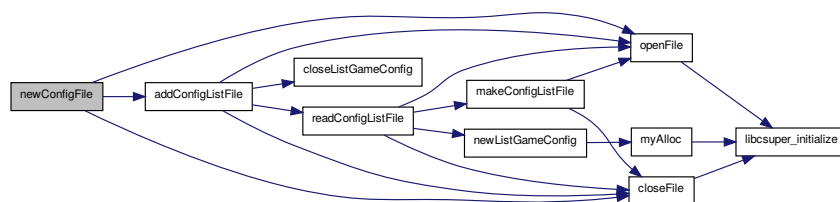
Parameters

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

Returns

TRUE if everything is OK, FALSE otherwise

Here is the call graph for this function:



4.7.2.5 list_game_config * newListGameConfig (int nb_config)

Create a list of game configuration.

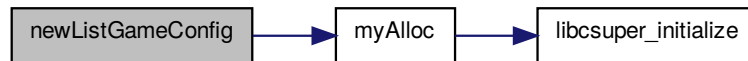
Parameters

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

Returns

une [list_game_config](#)

Here is the call graph for this function:



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

Read a game configuration file and close the list of game configuration

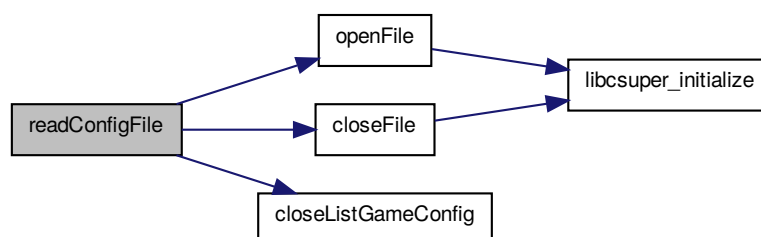
Parameters

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:



4.7.2.7 `list_game_config * readConfigListFile (char * home_path)`

Read the file which contain the list of game configuration.

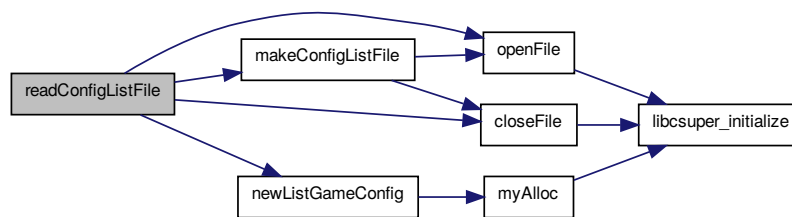
Parameters

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

Returns

a [list_game_config](#)

Here is the call graph for this function:



4.7.2.8 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

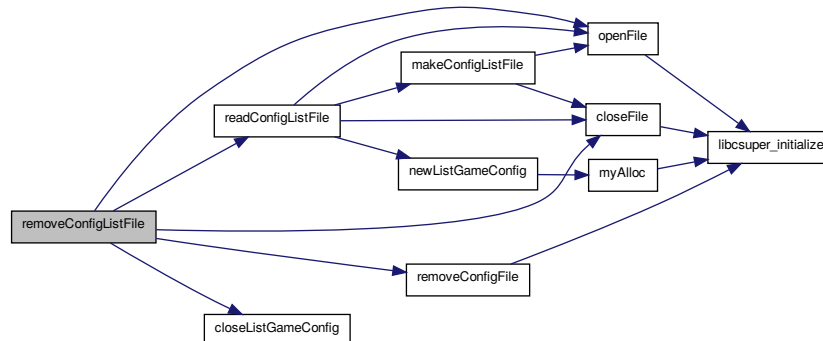
TRUE if everything is OK, FALSE otherwise

Here is the call graph for this function:



4.7.2.9 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"

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)

4.8.1 Detailed Description

Game configurations.

Author

Remi BERTHO

Date

16/04/14

Version

2.2.0

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

4.8.3 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.

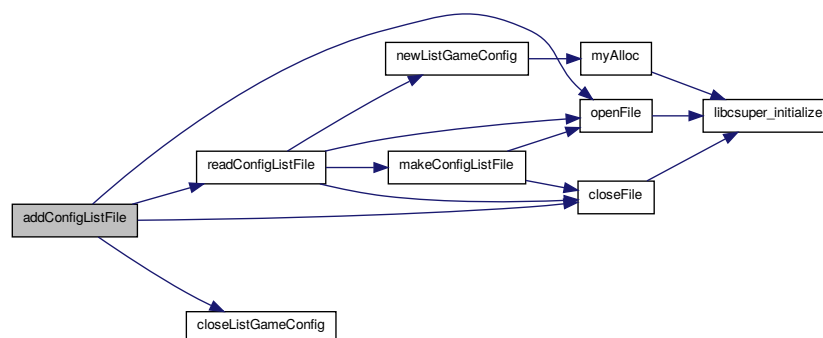
Parameters

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

TRUE if everything is OK, 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	*ptr_list_config	a pointer on a list of game configuration
----	------------------	---

4.8.3.3 int makeConfigListFile (char * home_path)

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

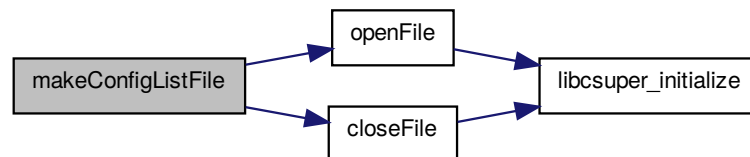
Parameters

in	*home_path	the path to the home directory
----	------------	--------------------------------

Returns

TRUE if everything is OK, FALSE otherwise

Here is the call graph for this function:



4.8.3.4 int newConfigFile (game_config config, char * home_path)

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

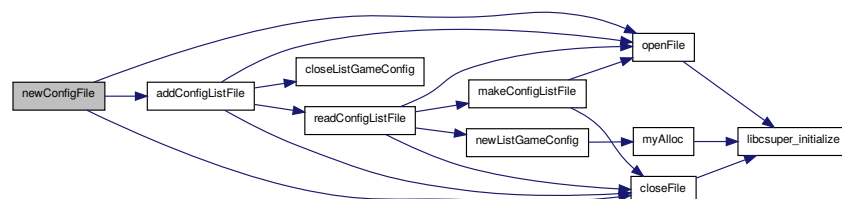
Parameters

in	config	the gale configuration
in	home_path	the path to the home directory

Returns

TRUE if everything is OK, FALSE otherwise

Here is the call graph for this function:



4.8.3.5 `list_game_config* newListGameConfig (int nb_config)`

Create a list of game configuration.

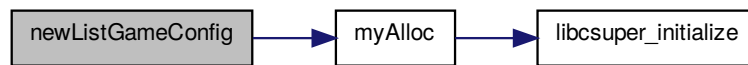
Parameters

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

Returns

une `list_game_config`

Here is the call graph for this function:



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

Read a game configuration file and close the list of game configuration

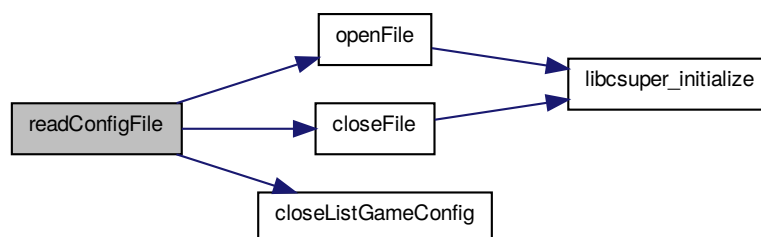
Parameters

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

Returns

a `list_game_config`

Here is the call graph for this function:



4.8.3.7 list_game_config* readConfigListFile (char * *home_path*)

Read the file which contain the list of game configuration.

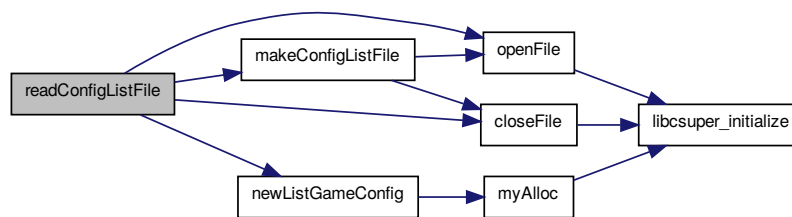
Parameters

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

Returns

a [list_game_config](#)

Here is the call graph for this function:



4.8.3.8 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

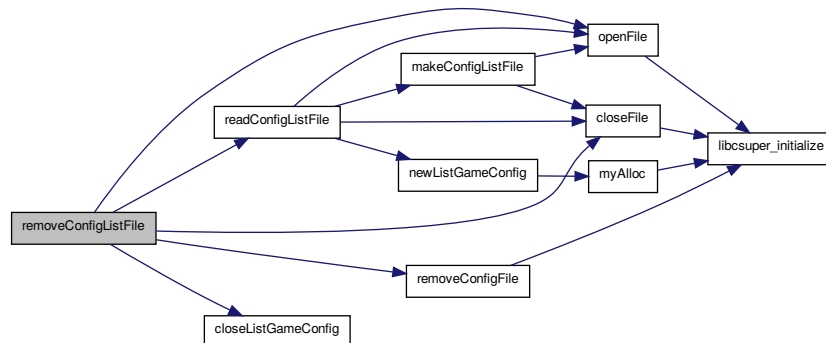
TRUE if everything is OK, FALSE otherwise

Here is the call graph for this function:



4.8.3.9 `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

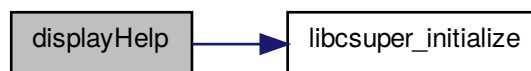
2.2.0

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

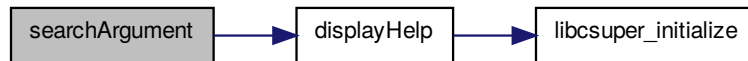
Parameters

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

TRUE if the function founded an argument, 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"`

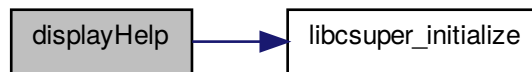
Define the reduce argument which call to read a file to "-r"

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

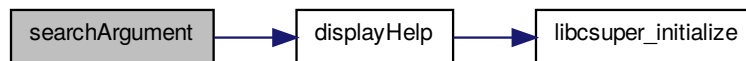
Parameters

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

TRUE if the function founded an argument, 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)

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

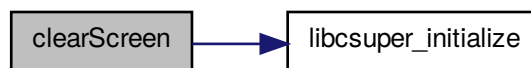
Parameters

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 void libcsuper_initialize ()

Initialize libcsuper with gettext.

4.12.2.7 void * myAlloc (int size_alloue)

Allocate a memory block and check if everything is OK.

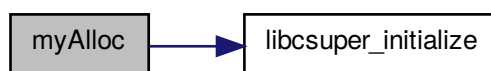
Parameters

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

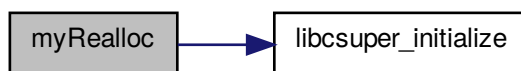
Returns

a pointer on the allocate memory block

Here is the call graph for this function:

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

Here is the call graph for this function:

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

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

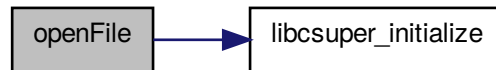
Parameters

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:

**4.12.2.10 int readFileSize (FILE * ptr_file)**

Read the size of the file

Parameters

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

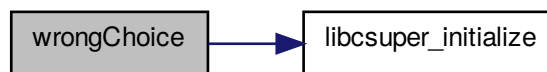
Returns

the size of the file

4.12.2.11 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 TRUE 1`
- `#define 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)

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 FALSE 0`

Definit FALSE a 0

4.13.2.3 `#define TRUE 1`

Definit TRUE a 1

4.13.3 Function Documentation

4.13.3.1 void `addFileCsuExtension` (char * *file_name*)

Add the csu file extension

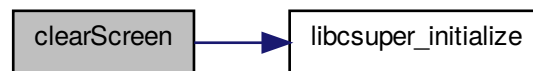
Parameters

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

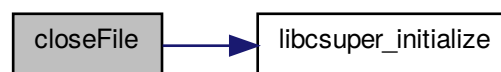
Parameters

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 void libcsuper_initialize ()

Initialize libcsuper with gettext.

4.13.3.7 void* myAlloc (int *size_alloue*)

Allocate a memory block and check if everything is OK.

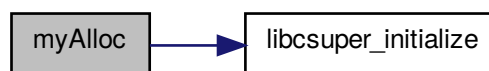
Parameters

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

Returns

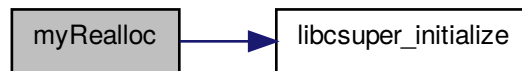
a pointer on the allocate memory block

Here is the call graph for this function:



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

Here is the call graph for this function:



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

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

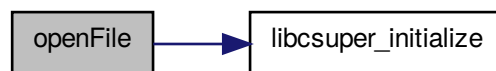
Parameters

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:



4.13.3.10 int readFileSize (FILE * *ptr_file*)

Read the size of the file

Parameters

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

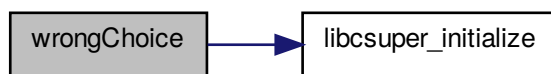
Returns

the size of the file

4.13.3.11 void wrongChoice ()

Display an error message.

Here is the call graph for this function:



Index

- - share.h, 52
- addConfigListFile
 - game_config.c, 33
 - game_config.h, 38
- addDistributorCsuStruct
 - csu_struct.c, 17
 - csu_struct.h, 21
- addFileCsuExtension
 - share.c, 48
 - share.h, 52
- begin_score
 - game_config, 7
- changeSystemPath
 - file_system_path.c, 26
 - file_system_path.h, 29
- clearScreen
 - share.c, 48
 - share.h, 53
- closeCsuStruct
 - csu_struct.c, 17
 - csu_struct.h, 22
- closeFile
 - share.c, 48
 - share.h, 53
- closeListGameConfig
 - game_config.c, 33
 - game_config.h, 39
- compareFloatAscending
 - share.c, 49
 - share.h, 53
- compareFloatDescending
 - share.c, 49
 - share.h, 54
- config
 - csuStruct, 5
- createFileSystemPath
 - file_system_path.c, 26
 - file_system_path.h, 30
- csu_files.c, 9
 - deleteCsuFile, 9
 - openFileCsuExtension, 10
 - readCsuFile, 10
 - renameCsuFile, 11
 - writeCsuFile, 11
 - writeFileNewTurn, 12
- csu_files.h, 12
 - deleteCsuFile, 13
 - FILE_EXTENSION, 13
 - openFileCsuExtension, 14
 - readCsuFile, 14
 - renameCsuFile, 15
 - writeCsuFile, 15
 - writeFileNewTurn, 16
- csu_struct.c, 16
 - addDistributorCsuStruct, 17
 - closeCsuStruct, 17
 - endNewTurn, 18
 - exceedMaxNumber, 18
 - maxNbTurn, 18
 - newCsuStruct, 18
 - rankCalculation, 19
 - searchPlayerIndex, 19
 - startNewTurn, 20
- csu_struct.h, 20
 - addDistributorCsuStruct, 21
 - closeCsuStruct, 22
 - endNewTurn, 22
 - exceedMaxNumber, 22
 - maxNbTurn, 22
 - newCsuStruct, 24
 - rankCalculation, 24
 - SIZE_MAX_NAME, 21
 - searchPlayerIndex, 24
 - startNewTurn, 25
 - VERSION, 21
- 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
- deleteCsuFile
 - csu_files.c, 9
 - csu_files.h, 13
- displayHelp

- main_argument.c, [44](#)
- main_argument.h, [46](#)
- distributor
 - csuStruct, [5](#)
- endNewTurn
 - csu_struct.c, [18](#)
 - csu_struct.h, [22](#)
- exceedMaxNumber
 - csu_struct.c, [18](#)
 - csu_struct.h, [22](#)
- FALSE
 - share.h, [52](#)
- FILE_EXTENSION
 - csu_files.h, [13](#)
- file_system_path.c, [25](#)
 - changeSystemPath, [26](#)
 - createFileSystemPath, [26](#)
 - readFileSystemPath, [27](#)
 - readHomePath, [27](#)
 - readHomePathSlash, [28](#)
 - readSystemPath, [28](#)
- file_system_path.h, [28](#)
 - changeSystemPath, [29](#)
 - createFileSystemPath, [30](#)
 - readFileSystemPath, [30](#)
 - readHomePath, [31](#)
 - readHomePathSlash, [31](#)
 - readSystemPath, [31](#)
- first_way
 - game_config, [7](#)
- game_config, [7](#)
 - begin_score, [7](#)
 - first_way, [7](#)
 - max, [7](#)
 - name, [7](#)
 - nb_max, [7](#)
 - number_after_comma, [7](#)
 - turn_by_turn, [7](#)
 - use_distributor, [7](#)
- game_config.c, [32](#)
 - addConfigListFile, [33](#)
 - closeListGameConfig, [33](#)
 - makeConfigListFile, [33](#)
 - newConfigFile, [34](#)
 - newListGameConfig, [34](#)
 - readConfigFile, [35](#)
 - readConfigListFile, [35](#)
 - removeConfigFile, [36](#)
 - removeConfigListFile, [36](#)
- game_config.h, [37](#)
 - addConfigListFile, [38](#)
 - closeListGameConfig, [39](#)
 - makeConfigListFile, [39](#)
 - newConfigFile, [39](#)
 - newListGameConfig, [40](#)
 - readConfigFile, [40](#)
 - readConfigListFile, [40](#)
 - removeConfigFile, [42](#)
 - removeConfigListFile, [42](#)
- HELP
 - main_argument.h, [46](#)
- libcsuper.h, [43](#)
- libcsuper_initialize
 - share.c, [49](#)
 - share.h, [54](#)
- list_game_config, [8](#)
 - name_game_config, [8](#)
 - nb_config, [8](#)
- MAIN_FOLDER_NAME
 - file_system_path.h, [29](#)
- 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, [33](#)
 - game_config.h, [39](#)
- max
 - game_config, [7](#)
- maxNbTurn
 - csu_struct.c, [18](#)
 - csu_struct.h, [22](#)
- month
 - csuStruct, [6](#)
- myAlloc
 - share.c, [49](#)
 - share.h, [54](#)
- myRealloc
 - share.c, [50](#)
 - share.h, [54](#)
- 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, 34
 game_config.h, 39
newCsuStruct
 csu_struct.c, 18
 csu_struct.h, 24
newListGameConfig
 game_config.c, 34
 game_config.h, 40
number_after_comma
 game_config, 7

OPEN_FILE
 main_argument.h, 46
openFile
 share.c, 50
 share.h, 55
openFileCsuExtension
 csu_files.c, 10
 csu_files.h, 14

player_names
 csuStruct, 6
point
 csuStruct, 6

READ_FILE
 main_argument.h, 46
rank
 csuStruct, 6
rankCalculation
 csu_struct.c, 19
 csu_struct.h, 24
readConfigFile
 game_config.c, 35
 game_config.h, 40
readConfigListFile
 game_config.c, 35
 game_config.h, 40
readCsuFile
 csu_files.c, 10
 csu_files.h, 14
readFileSize
 share.c, 51
 share.h, 55
readFileSystemPath
 file_system_path.c, 27
 file_system_path.h, 30
readHomePath
 file_system_path.c, 27
 file_system_path.h, 31
readHomePathSlash
 file_system_path.c, 28
 file_system_path.h, 31
readSystemPath
 file_system_path.c, 28
 file_system_path.h, 31
removeConfigFile
 game_config.c, 36
 game_config.h, 42
removeConfigListFile
 game_config.c, 36
 game_config.h, 42
renameCsuFile
 csu_files.c, 11
 csu_files.h, 15

SIZE_MAX_NAME
 csu_struct.h, 21
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, 19
 csu_struct.h, 24
share.c, 47
 addFileCsuExtension, 48
 clearScreen, 48
 closeFile, 48
 compareFloatAscending, 49
 compareFloatDescending, 49
 libcsuper_initialize, 49
 myAlloc, 49
 myRealloc, 50
 openFile, 50
 readFileSize, 51
 wrongChoice, 51
share.h, 51
 _, 52
 addFileCsuExtension, 52
 clearScreen, 53
 closeFile, 53
 compareFloatAscending, 53
 compareFloatDescending, 54
 FALSE, 52
 libcsuper_initialize, 54
 myAlloc, 54
 myRealloc, 54
 openFile, 55
 readFileSize, 55
 TRUE, 52
 wrongChoice, 55
size_max_name
 csuStruct, 6
startNewTurn
 csu_struct.c, 20
 csu_struct.h, 25

TRUE
 share.h, 52

total_points
 csuStruct, [6](#)

turn_by_turn
 game_config, [7](#)

use_distributor
 game_config, [7](#)

VERSION
 csu_struct.h, [21](#)

version
 csuStruct, [6](#)

writeCsuFile
 csu_files.c, [11](#)
 csu_files.h, [15](#)

writeFileNewTurn
 csu_files.c, [12](#)
 csu_files.h, [16](#)

wrongChoice
 share.c, [51](#)
 share.h, [55](#)

year
 csuStruct, [6](#)