

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

## 2.2.1

Generated by Doxygen 1.8.6

Wed Apr 30 2014 19:40:22



# Contents

<b>1</b>	<b>Data Structure Index</b>	<b>1</b>
1.1	Data Structures . . . . .	1
<b>2</b>	<b>File Index</b>	<b>3</b>
2.1	File List . . . . .	3
<b>3</b>	<b>Data Structure Documentation</b>	<b>5</b>
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
<b>4</b>	<b>File Documentation</b>	<b>9</b>
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
4.8.3	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	38
4.8.3.5	makeConfigListFile	39
4.8.3.6	newConfigFile	39
4.8.3.7	newListGameConfig	40
4.8.3.8	readConfigFile	40
4.8.3.9	readConfigListFile	41
4.8.3.10	removeConfigFile	41
4.8.3.11	removeConfigListFile	42
4.9	libcsuper.h File Reference	42
4.9.1	Detailed Description	43
4.10	main_argument.c File Reference	43
4.10.1	Detailed Description	43
4.10.2	Function Documentation	43
4.10.2.1	displayHelp	43
4.10.2.2	searchArgument	44
4.11	main_argument.h File Reference	44
4.11.1	Detailed Description	45
4.11.2	Macro Definition Documentation	45
4.11.2.1	HELP	45
4.11.2.2	OPEN_FILE	45
4.11.2.3	READ_FILE	45
4.11.2.4	STRING_HELP	45
4.11.2.5	STRING_HELP_RED	45

4.11.2.6	STRING_OPEN_FILE	45
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	46
4.12	share.c File Reference	47
4.12.1	Detailed Description	47
4.12.2	Function Documentation	47
4.12.2.1	addFileCsuExtension	47
4.12.2.2	clearScreen	48
4.12.2.3	closeFile	48
4.12.2.4	compareFloatAscending	48
4.12.2.5	compareFloatDescending	48
4.12.2.6	deleteFile	49
4.12.2.7	libcsuper_initialize	49
4.12.2.8	myAlloc	49
4.12.2.9	myRealloc	50
4.12.2.10	openFile	50
4.12.2.11	readFileSize	51
4.12.2.12	renameFile	52
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	FALSE	53
4.13.2.3	TRUE	54
4.13.3	Function Documentation	54
4.13.3.1	addFileCsuExtension	54
4.13.3.2	clearScreen	54
4.13.3.3	closeFile	54
4.13.3.4	compareFloatAscending	55
4.13.3.5	compareFloatDescending	56
4.13.3.6	deleteFile	56
4.13.3.7	libcsuper_initialize	56
4.13.3.8	myAlloc	56
4.13.3.9	myRealloc	57
4.13.3.10	openFile	57

4.13.3.11 readFileSize . . . . .	58
4.13.3.12 renameFile . . . . .	58
4.13.3.13 wrongChoice . . . . .	59
<b>Index</b>	<b>60</b>



# Chapter 1

## Data Structure Index

### 1.1 Data Structures

Here are the data structures with brief descriptions:

<a href="#">csuStruct</a>	5
<a href="#">game_config</a>	7
<a href="#">list_game_config</a>	8



## Chapter 2

# File Index

### 2.1 File List

Here is a list of all files with brief descriptions:

<a href="#">csu_files.c</a>	Files management . . . . .	9
<a href="#">csu_files.h</a>	Files management . . . . .	11
<a href="#">csu_struct.c</a>	Management of the csu files . . . . .	14
<a href="#">csu_struct.h</a>	Management of the csu files header . . . . .	18
<a href="#">file_system_path.c</a>	Fonctions qui l'emrankment des fichiers sauvegardes . . . . .	23
<a href="#">file_system_path.h</a>	Prototypes des fonctions qui l'emrankment des fichiers sauvegardes . . . . .	26
<a href="#">game_config.c</a>	Game configuration . . . . .	30
<a href="#">game_config.h</a>	Game configurations . . . . .	36
<a href="#">libcsuper.h</a>	Inclusion of all header files of libcsuper . . . . .	42
<a href="#">main_argument.c</a>	Begin csuper . . . . .	43
<a href="#">main_argument.h</a>	Begin csuper . . . . .	44
<a href="#">share.c</a>	Essential function of libcsuper . . . . .	47
<a href="#">share.h</a>	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.

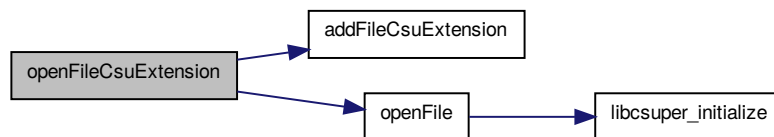
**Parameters**

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.

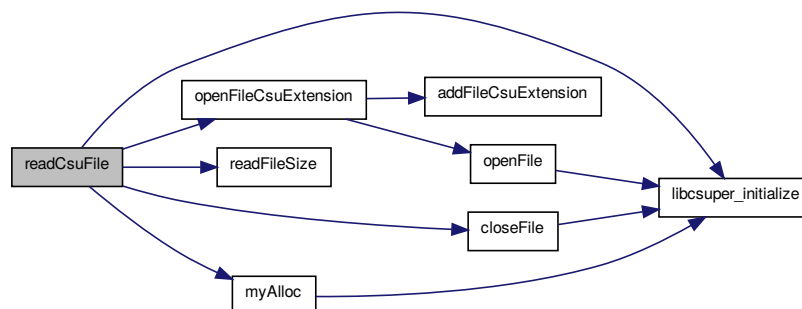
**Parameters**

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

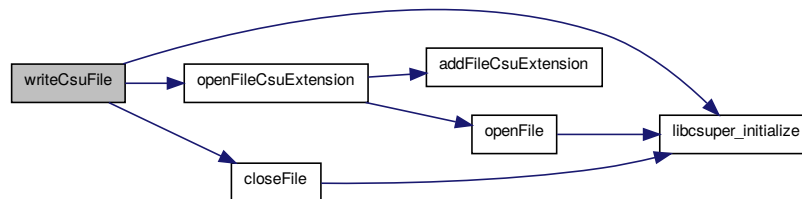
## Parameters

in	<i>*file_name</i>	the filename
in	<i>*ptr_csu_struct</i>	a pointer on a <a href="#">csuStruct</a>

## Returns

TRUE if everything is OK, 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

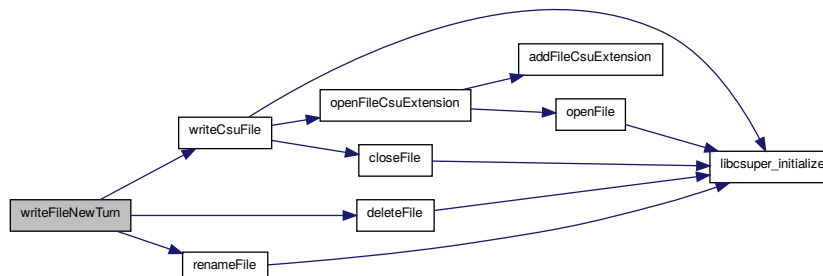
## Parameters

in	<i>*file_name</i>	the filename
in	<i>*ptr_csu_struct</i>	a pointer on a <a href="#">csuStruct</a>

## 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)

### 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.

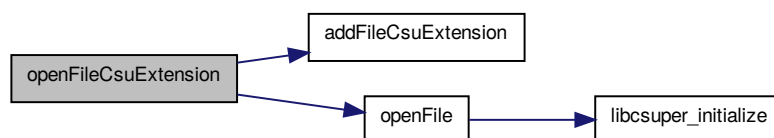
## Parameters

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.

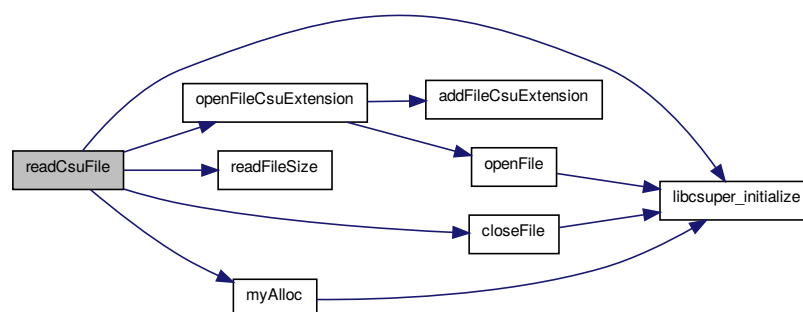
## Parameters

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

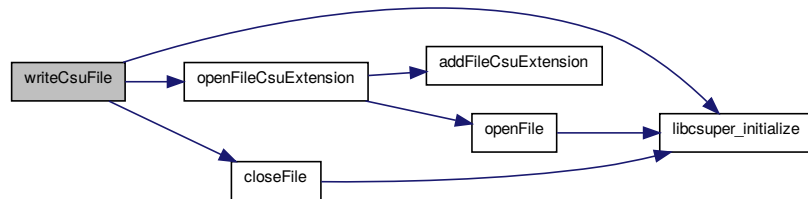
## Parameters

in	<i>*file_name</i>	the filename
in	<i>*ptr_csu_struct</i>	a pointer on a <a href="#">csuStruct</a>

## Returns

TRUE if everything is OK, 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

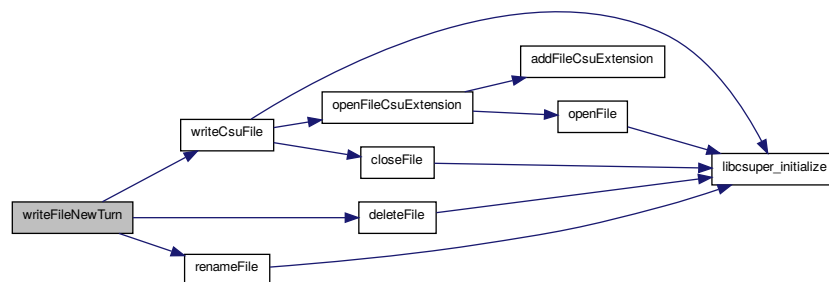
## Parameters

in	<i>*file_name</i>	the filename
in	<i>*ptr_csu_struct</i>	a pointer on a <a href="#">csuStruct</a>

## 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

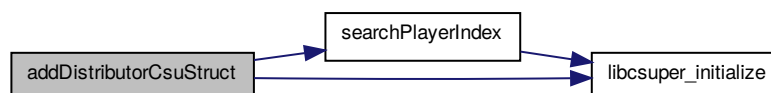
#### 4.3.2.1 void addDistributorCsuStruct ( [csuStruct](#) \* *ptr\_csu\_struct*, char \* *distributor\_name* )

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 <a href="#">csuStruct</a>

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 <a href="#">csuStruct</a>
---------	------------------------	--------------------------------------------

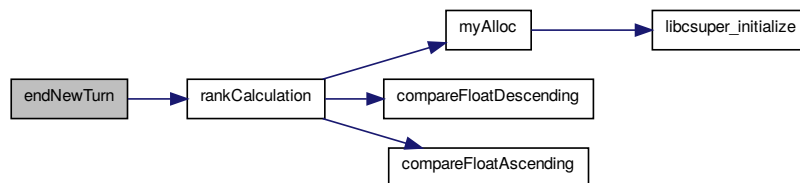
4.3.2.3 void endNewTurn ( [csuStruct](#) \* *ptr\_csu\_struct*, int *index\_player* )

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 <a href="#">csuStruct</a>
in, out	<i>index_player</i>	<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:

4.3.2.4 int exceedMaxNumber ( [csuStruct](#) \* *ptr\_csu\_struct* )

Check if someone exceed the maximum number

## Parameters

in	<i>*ptr_csu_struct</i>	a pointer on a <a href="#">csuStruct</a>
----	------------------------	------------------------------------------

## 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 <a href="#">csuStruct</a>
----	------------------------	------------------------------------------

## Returns

the maximal number of turn

4.3.2.6 [csuStruct](#) \* newCsuStruct ( float *nb\_player*, *game\_config config* )

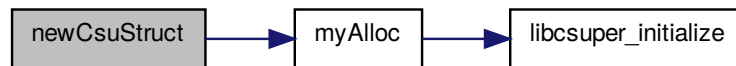
Create a new [csuStruct](#) from a game configuration and the number of player.



## 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:



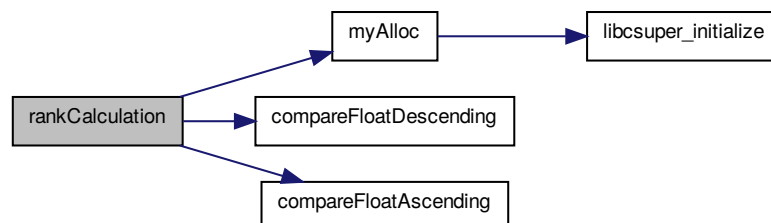
## 4.3.2.7 void rankCalculation ( csuStruct \* ptr\_csu\_struct )

Calculate the rank

## Parameters

in, out	<i>*ptr_csu_struct</i>	a pointer on a <a href="#">csuStruct</a>
---------	------------------------	------------------------------------------

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 <a href="#">csuStruct</a>

**Returns**

the index, -1 if there is not found

Here is the call graph for this function:



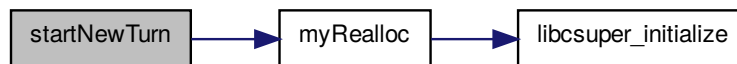
#### 4.3.2.9 void startNewTurn ( csuStruct \* ptr\_csu\_struct, int index\_player )

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

**Parameters**

in, out	*ptr_csu_struct	a pointer on a <a href="#">csuStruct</a>
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

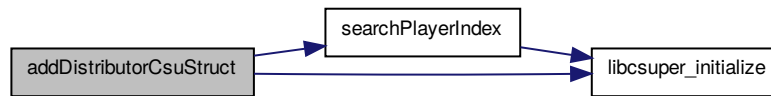
#### 4.4.3.1 void addDistributorCsuStruct ( csuStruct \* ptr\_csu\_struct, char \* distributor\_name )

Add the distributor on the structure

#### Parameters

in	* <i>distributor_</i> - <i>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 <a href="#">csuStruct</a>
---------	-----------------	--------------------------------------------

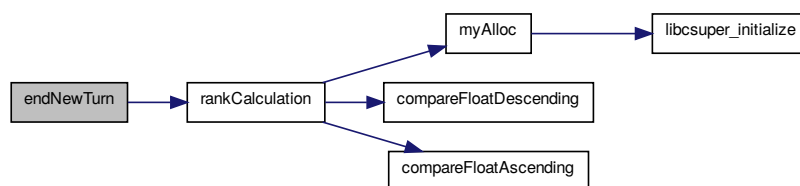
#### 4.4.3.3 void endNewTurn ( csuStruct \* ptr\_csu\_struct, int index\_player )

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 <a href="#">csuStruct</a>
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 <a href="#">csuStruct</a>
----	-----------------	------------------------------------------

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 <a href="#">csuStruct</a>
----	------------------------	------------------------------------------

## Returns

the maximal number of turn

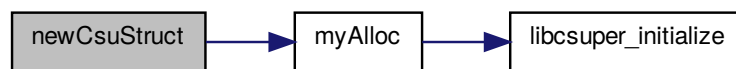
#### 4.4.3.6 `csuStruct* newCsuStruct ( float nb_player, game_config config )`

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:



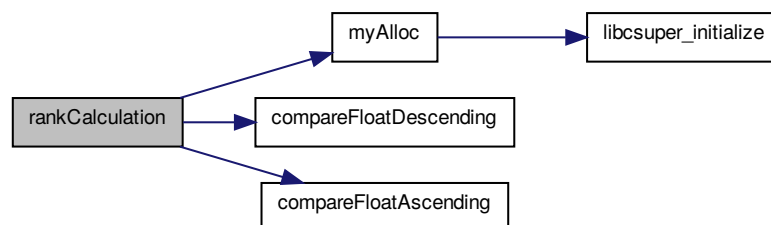
#### 4.4.3.7 `void rankCalculation ( csuStruct * ptr_csu_struct )`

Calculate the rank

## Parameters

in, out	<i>*ptr_csu_struct</i>	a pointer on a <a href="#">csuStruct</a>
---------	------------------------	------------------------------------------

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 <a href="#">csuStruct</a>

## Returns

the index, -1 if there is not found

Here is the call graph for this function:



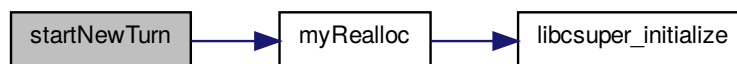
## 4.4.3.9 void startNewTurn ( csuStruct \* ptr\_csu\_struct, int index\_player )

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

## Parameters

in, out	<i>*ptr_csu_struct</i>	a pointer on a <a href="#">csuStruct</a>
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

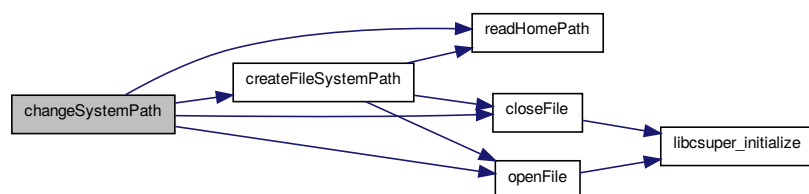
Parameters

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 ( )

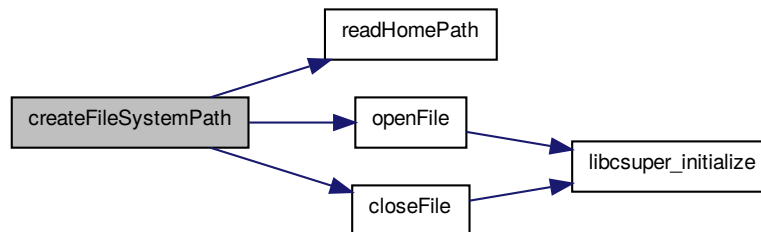
Create the folder and the file which contain the system path



**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

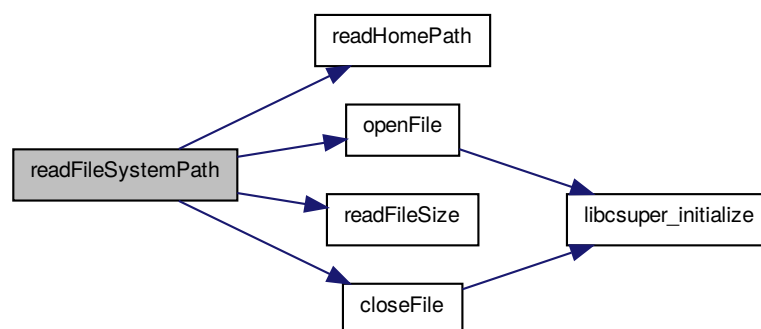
**Parameters**

<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**

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

4.5.2.5 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.

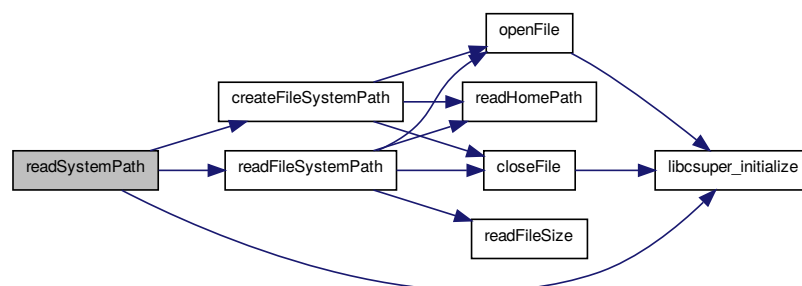
**Parameters**

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 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

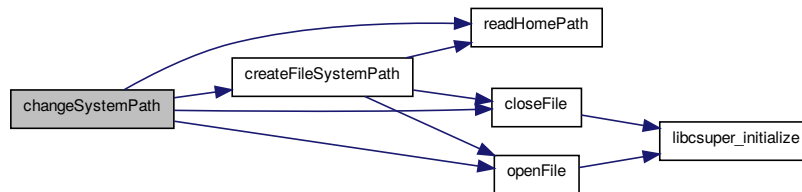
#### Parameters

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

**Returns**

TRUE if everything is OK, FALSE otherwise

Here is the call graph for this function:

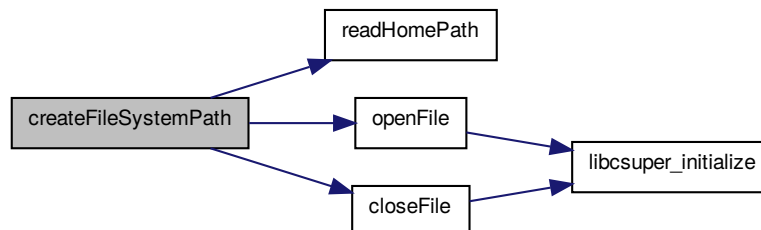
**4.6.3.2 int createFileSystemPath ( )**

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

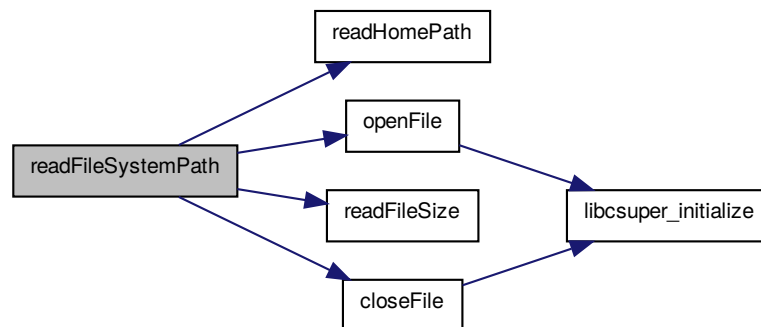
**Parameters**

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.

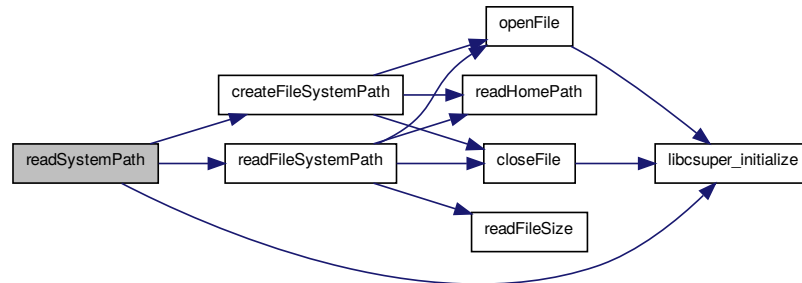
## Parameters

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)
- 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**

2.2.1

## 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.

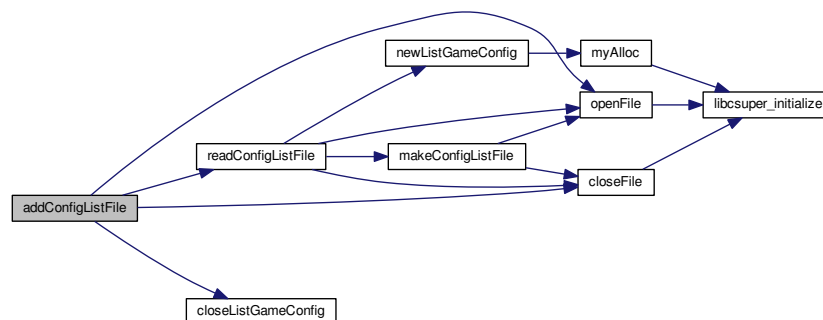
#### Parameters

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 exportConfigFile ( char \* home\_path, char \* file\_name )

Export all config file into a file.

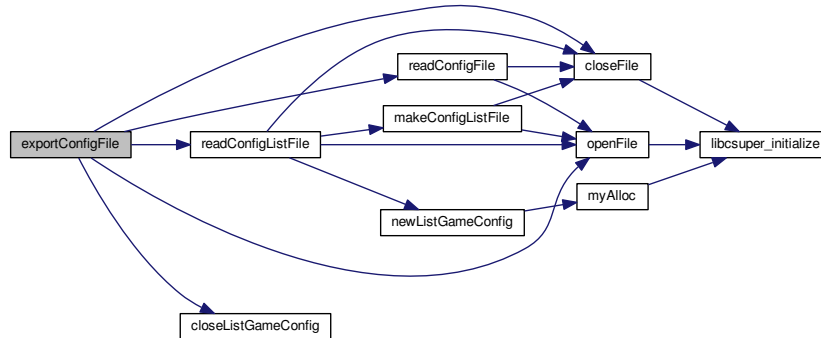
#### Parameters

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.

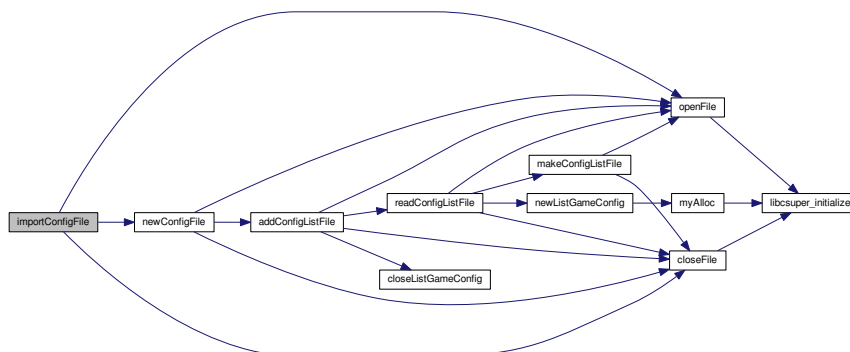
## Parameters

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



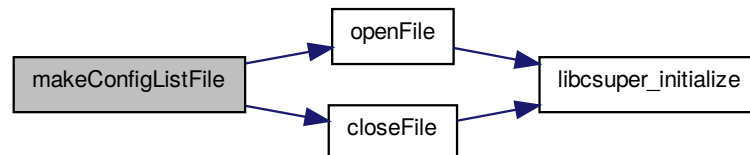
## Parameters

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

## Returns

TRUE if everything is OK, 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.

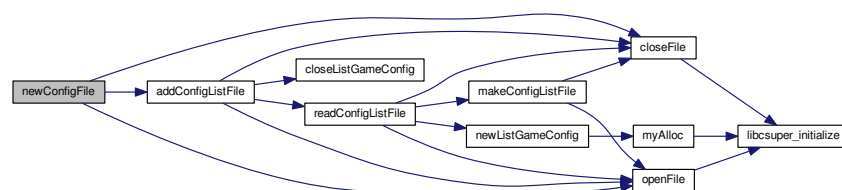
## Parameters

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

## Returns

TRUE if everything is OK, 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.

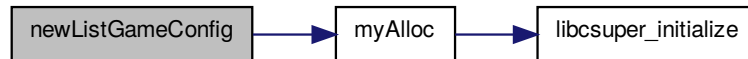
## Parameters

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.

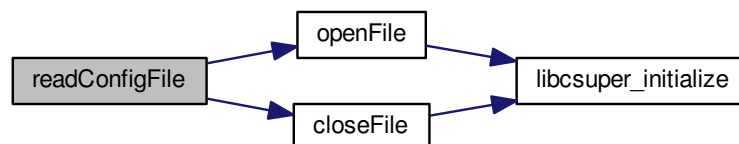
## Parameters

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.

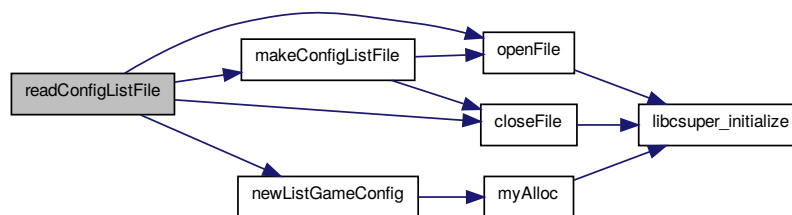
## Parameters

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

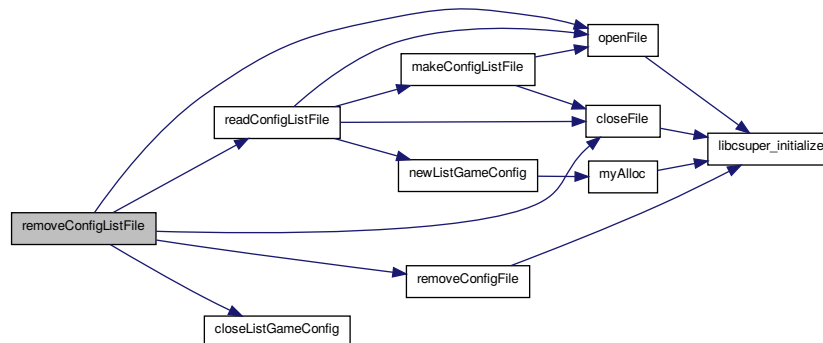
TRUE if everything is OK, FALSE otherwise

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"

### 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

2.2.1

### 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.

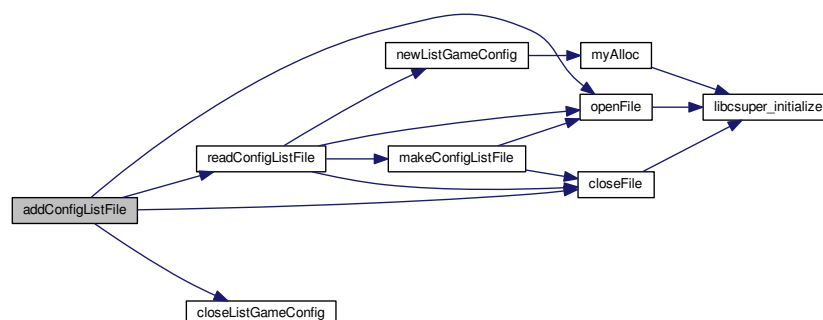
#### Parameters

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	<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.

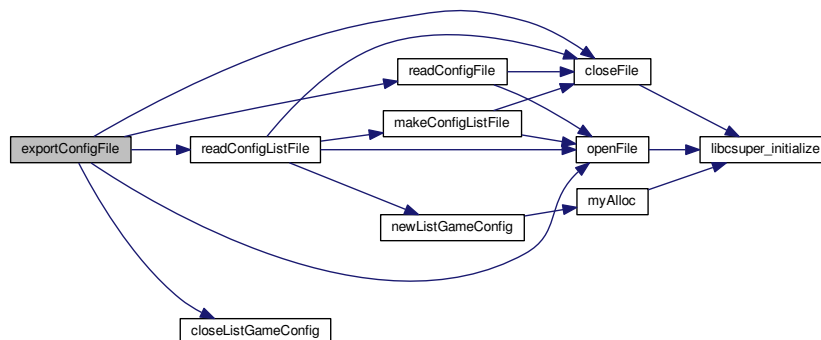
##### Parameters

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.

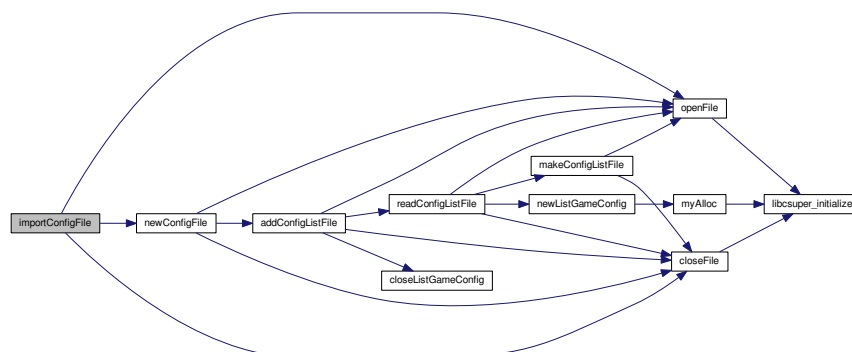
##### Parameters

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

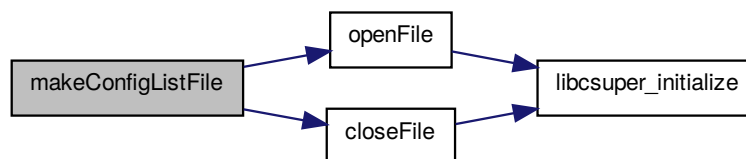
## Parameters

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

## Returns

TRUE if everything is OK, 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.

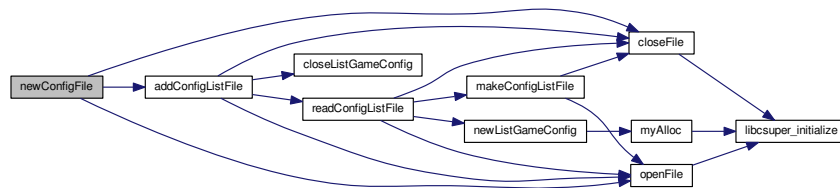
## Parameters

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

## Returns

TRUE if everything is OK, 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.

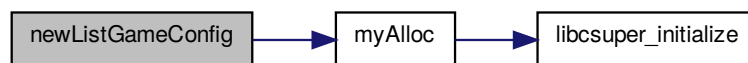
## Parameters

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

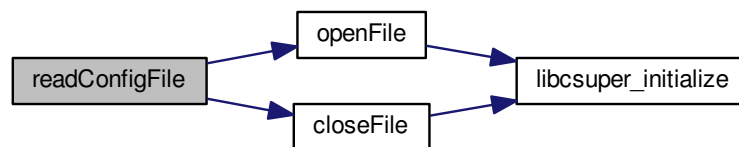


in	<i>index_read</i>	the index of the game configuration to be read
in	<i>ptr_list_config</i>	a pointer on the game configuration list
in	<i>ptr_config</i>	a pointer on a game configuration
in	<i>home_path</i>	the path to the home directory

**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.

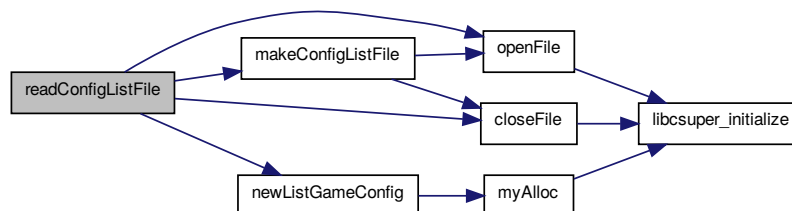
**Parameters**

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

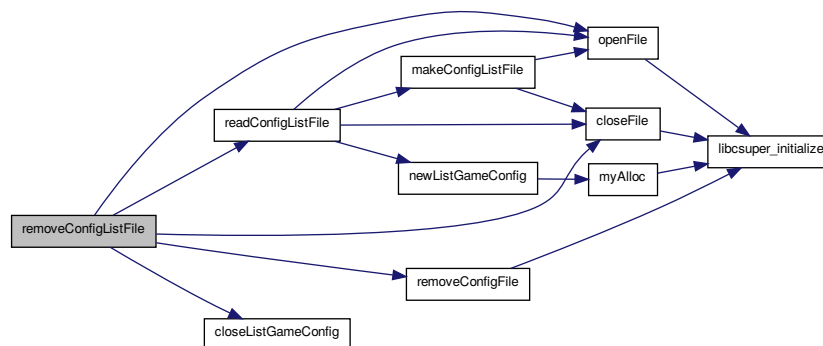
TRUE if everything is OK, FALSE otherwise

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**

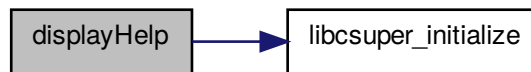
2.2.0

### 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

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"

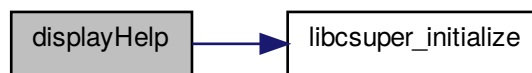
Define the reduce argument which call to read a file to "-r"

### 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

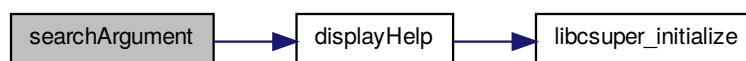
##### Parameters

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)
- 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

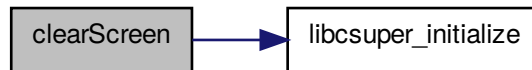
#### Parameters

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

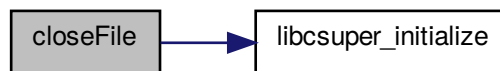
##### Parameters

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

TRUE if everything is OK, 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.

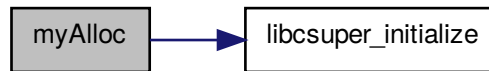
## Parameters

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

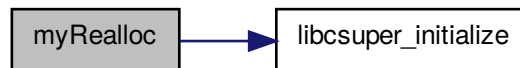
**Returns**

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.

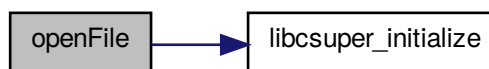
**Parameters**

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.

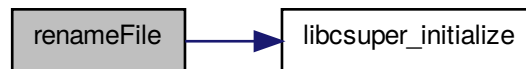
**Parameters**

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

**Returns**

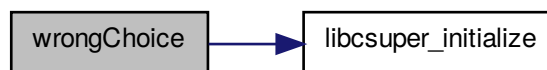
TRUE if everything is OK, FALSE otherwise

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 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)
- 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 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

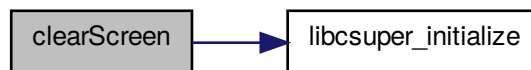
Parameters

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 int deleteFile ( char \* *file\_name* )**

Delete a file

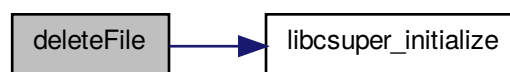
**Parameters**

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

**Returns**

TRUE if everything is OK, FALSE otherwise

Here is the call graph for this function:

**4.13.3.7 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.



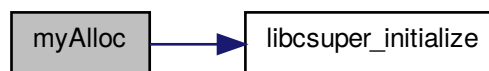
## Parameters

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

## Returns

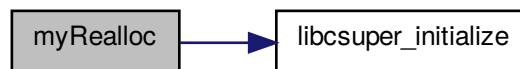
a pointer on the allocate memory block

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.

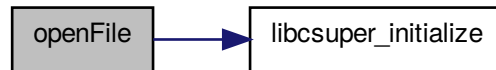
## Parameters

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.

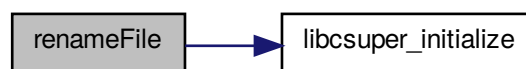
**Parameters**

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

**Returns**

TRUE if everything is OK, FALSE otherwise

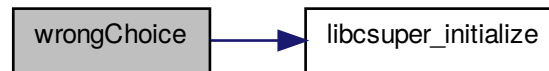
Here is the call graph for this function:



#### 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, [47](#)
  - share.h, [54](#)
- begin\_score
  - game\_config, [7](#)
- changeSystemPath
  - file\_system\_path.c, [24](#)
  - file\_system\_path.h, [27](#)
- clearScreen
  - share.c, [47](#)
  - share.h, [54](#)
- closeCsuStruct
  - csu\_struct.c, [15](#)
  - csu\_struct.h, [20](#)
- closeFile
  - share.c, [48](#)
  - share.h, [54](#)
- closeListGameConfig
  - game\_config.c, [31](#)
  - game\_config.h, [37](#)
- compareFloatAscending
  - share.c, [48](#)
  - share.h, [54](#)
- compareFloatDescending
  - share.c, [48](#)
  - share.h, [56](#)
- 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, [56](#)
- displayHelp
  - main\_argument.c, [43](#)
  - 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

FALSE  
    share.h, 53

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, 37  
    exportConfigFile, 38  
    importConfigFile, 38  
    makeConfigListFile, 39  
    newConfigFile, 39  
    newListGameConfig, 40  
    readConfigFile, 40  
    readConfigListFile, 41  
    removeConfigFile, 41  
    removeConfigListFile, 42

HELP  
    main\_argument.h, 45

importConfigFile  
    game\_config.c, 32  
    game\_config.h, 38

libcsuper.h, 42  
libcsuper\_initialize  
    share.c, 49  
    share.h, 56

list\_game\_config, 8  
    name\_game\_config, 8  
    nb\_config, 8

MAIN\_FOLDER\_NAME  
    file\_system\_path.h, 27

main\_argument.c, 43  
    displayHelp, 43  
    searchArgument, 44

main\_argument.h, 44  
    displayHelp, 46  
    HELP, 45  
    OPEN\_FILE, 45  
    READ\_FILE, 45  
    STRING\_HELP, 45  
    STRING\_HELP\_RED, 45  
    STRING\_OPEN\_FILE, 45  
    STRING\_READ\_FILE, 46  
    searchArgument, 46

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, 49  
    share.h, 56

myRealloc  
    share.c, 50  
    share.h, 57

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, 39
- newCsuStruct
  - csu\_struct.c, 16
  - csu\_struct.h, 22
- newListGameConfig
  - game\_config.c, 33
  - game\_config.h, 40
- OPEN\_FILE
  - main\_argument.h, 45
- openFile
  - share.c, 50
  - share.h, 57
- openFileCsuExtension
  - csu\_files.c, 9
  - csu\_files.h, 12
- player\_names
  - csuStruct, 6
- point
  - csuStruct, 6
- READ\_FILE
  - main\_argument.h, 45
- rank
  - csuStruct, 6
- rankCalculation
  - csu\_struct.c, 17
  - csu\_struct.h, 22
- readConfigFile
  - game\_config.c, 34
  - game\_config.h, 40
- readConfigListFile
  - game\_config.c, 34
  - game\_config.h, 41
- readCsuFile
  - csu\_files.c, 10
  - csu\_files.h, 13
- readFileSize
  - share.c, 50
  - share.h, 58
- readFileSystemPath
  - file\_system\_path.c, 25
  - file\_system\_path.h, 28
- readHomePath
  - file\_system\_path.c, 25
  - file\_system\_path.h, 29
- 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, 41
- removeConfigListFile
  - game\_config.c, 35
  - game\_config.h, 42
- renameFile
  - share.c, 52
  - share.h, 58
- SIZE\_MAX\_NAME
  - csu\_struct.h, 19
- STRING\_HELP
  - main\_argument.h, 45
- STRING\_HELP\_RED
  - main\_argument.h, 45
- STRING\_OPEN\_FILE
  - main\_argument.h, 45
- STRING\_READ\_FILE
  - main\_argument.h, 46
- searchArgument
  - main\_argument.c, 44
  - main\_argument.h, 46
- searchPlayerIndex
  - csu\_struct.c, 17
  - csu\_struct.h, 22
- share.c, 47
  - addFileCsuExtension, 47
  - clearScreen, 47
  - closeFile, 48
  - compareFloatAscending, 48
  - compareFloatDescending, 48
  - deleteFile, 49
  - libcsuper\_initialize, 49
  - myAlloc, 49
  - myRealloc, 50
  - openFile, 50
  - readFileSize, 50
  - renameFile, 52
  - wrongChoice, 52
- share.h, 52
  - \_, 53
  - addFileCsuExtension, 54
  - clearScreen, 54
  - closeFile, 54
  - compareFloatAscending, 54
  - compareFloatDescending, 56
  - deleteFile, 56
  - FALSE, 53
  - libcsuper\_initialize, 56
  - myAlloc, 56
  - myRealloc, 57
  - openFile, 57
  - readFileSize, 58

- renameFile, [58](#)
- TRUE, [53](#)
- wrongChoice, [58](#)
- size\_max\_name
  - csuStruct, [6](#)
- startNewTurn
  - csu\_struct.c, [18](#)
  - csu\_struct.h, [23](#)
- TRUE
  - share.h, [53](#)
- 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, [58](#)
- year
  - csuStruct, [6](#)