

Csuper - Compteur de Score Universel Permettant l'Exemption de Reflexion  
4.0.2

Generated by Doxygen 1.8.6

Wed Sep 10 2014 17:06:59



# 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 . . . . .	5
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 . . . . .	6
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_based	7
3.2.2.8	use_distributor	7
3.3	list_game_config Struct Reference	8
3.3.1	Detailed Description	8
3.3.2	Field Documentation	8
3.3.2.1	name_game_config	8
3.3.2.2	nb_config	8
3.4	main_window_size Struct Reference	8
3.4.1	Detailed Description	8
3.4.2	Field Documentation	8
3.4.2.1	height	8
3.4.2.2	is_maximize	8
3.4.2.3	width	9
3.5	toolbar_button_preferences_struct Struct Reference	9
3.5.1	Detailed Description	9
3.5.2	Field Documentation	9
3.5.2.1	about	9
3.5.2.2	copy	9
3.5.2.3	cut	9
3.5.2.4	delete	10
3.5.2.5	game_configuration_preferences	10
3.5.2.6	new	10
3.5.2.7	open	10
3.5.2.8	paste	10
3.5.2.9	preferences	10
3.5.2.10	properties	10
3.5.2.11	redo	10
3.5.2.12	save_as	10
3.5.2.13	separator_1	10
3.5.2.14	separator_2	10
3.5.2.15	separator_3	10
3.5.2.16	separator_4	11
3.5.2.17	separator_5	11
3.5.2.18	toolbar_button_preferences	11
3.5.2.19	undo	11
<b>4</b>	<b>File Documentation</b>	<b>13</b>
4.1	csu_files.c File Reference	13
4.1.1	Detailed Description	13
4.1.2	Function Documentation	13

4.1.2.1	<a href="#">openFileCsuExtension</a>	13
4.1.2.2	<a href="#">readCsuFile</a>	14
4.1.2.3	<a href="#">writeCsuFile</a>	14
4.1.2.4	<a href="#">writeFileNewTurn</a>	15
4.2	<a href="#">csu_files.h File Reference</a>	15
4.2.1	<a href="#">Detailed Description</a>	16
4.2.2	<a href="#">Macro Definition Documentation</a>	16
4.2.2.1	<a href="#">FILE_EXTENSION</a>	16
4.2.2.2	<a href="#">SIZE_MAX_FILE_NAME</a>	16
4.2.2.3	<a href="#">STRING_CHECK_CSU_FILE</a>	16
4.2.3	<a href="#">Function Documentation</a>	16
4.2.3.1	<a href="#">openFileCsuExtension</a>	16
4.2.3.2	<a href="#">readCsuFile</a>	17
4.2.3.3	<a href="#">writeCsuFile</a>	17
4.2.3.4	<a href="#">writeFileNewTurn</a>	18
4.3	<a href="#">csu_struct.c File Reference</a>	18
4.3.1	<a href="#">Detailed Description</a>	19
4.3.2	<a href="#">Function Documentation</a>	19
4.3.2.1	<a href="#">addDistributorCsuStruct</a>	19
4.3.2.2	<a href="#">closeCsuStruct</a>	19
4.3.2.3	<a href="#">copyCsuStruct</a>	20
4.3.2.4	<a href="#">differeentsPlayerName</a>	20
4.3.2.5	<a href="#">endNewTurn</a>	20
4.3.2.6	<a href="#">exceedMaxNumber</a>	21
4.3.2.7	<a href="#">maxNbTurn</a>	21
4.3.2.8	<a href="#">newCsuStruct</a>	21
4.3.2.9	<a href="#">rankCalculation</a>	22
4.3.2.10	<a href="#">searchIndexFromPosition</a>	23
4.3.2.11	<a href="#">searchPlayerIndex</a>	23
4.3.2.12	<a href="#">startNewTurn</a>	24
4.4	<a href="#">csu_struct.h File Reference</a>	24
4.4.1	<a href="#">Detailed Description</a>	25
4.4.2	<a href="#">Macro Definition Documentation</a>	25
4.4.2.1	<a href="#">SIZE_MAX_NAME</a>	25
4.4.2.2	<a href="#">VERSION</a>	25
4.4.3	<a href="#">Function Documentation</a>	25
4.4.3.1	<a href="#">addDistributorCsuStruct</a>	25
4.4.3.2	<a href="#">closeCsuStruct</a>	26
4.4.3.3	<a href="#">copyCsuStruct</a>	26
4.4.3.4	<a href="#">differeentsPlayerName</a>	26

4.4.3.5	<a href="#">endNewTurn</a>	27
4.4.3.6	<a href="#">exceedMaxNumber</a>	27
4.4.3.7	<a href="#">maxNbTurn</a>	27
4.4.3.8	<a href="#">newCsuStruct</a>	27
4.4.3.9	<a href="#">rankCalculation</a>	28
4.4.3.10	<a href="#">searchIndexFromPosition</a>	28
4.4.3.11	<a href="#">searchPlayerIndex</a>	29
4.4.3.12	<a href="#">startNewTurn</a>	29
4.5	<a href="#">file.c File Reference</a>	30
4.5.1	<a href="#">Detailed Description</a>	30
4.5.2	<a href="#">Function Documentation</a>	30
4.5.2.1	<a href="#">closeFile</a>	30
4.5.2.2	<a href="#">deleteFile</a>	31
4.5.2.3	<a href="#">openFile</a>	31
4.5.2.4	<a href="#">readFileSize</a>	32
4.5.2.5	<a href="#">renameFile</a>	32
4.6	<a href="#">file.h File Reference</a>	32
4.6.1	<a href="#">Detailed Description</a>	33
4.6.2	<a href="#">Function Documentation</a>	33
4.6.2.1	<a href="#">closeFile</a>	33
4.6.2.2	<a href="#">deleteFile</a>	33
4.6.2.3	<a href="#">openFile</a>	34
4.6.2.4	<a href="#">readFileSize</a>	34
4.6.2.5	<a href="#">renameFile</a>	35
4.7	<a href="#">filename.c File Reference</a>	35
4.7.1	<a href="#">Detailed Description</a>	35
4.7.2	<a href="#">Function Documentation</a>	36
4.7.2.1	<a href="#">addFileCsuExtension</a>	36
4.7.2.2	<a href="#">checkFilename</a>	36
4.7.2.3	<a href="#">checkPath</a>	36
4.7.2.4	<a href="#">getFolderFromFilename</a>	37
4.7.2.5	<a href="#">getSimpleFilenameFromFullFilename</a>	37
4.7.2.6	<a href="#">readHomePath</a>	37
4.7.2.7	<a href="#">readHomePathSlash</a>	38
4.8	<a href="#">filename.h File Reference</a>	38
4.8.1	<a href="#">Detailed Description</a>	38
4.8.2	<a href="#">Function Documentation</a>	38
4.8.2.1	<a href="#">addFileCsuExtension</a>	38
4.8.2.2	<a href="#">checkFilename</a>	39
4.8.2.3	<a href="#">checkPath</a>	39

4.8.2.4	<a href="#">getFolderFromFilename</a>	40
4.8.2.5	<a href="#">getSimpleFilenameFromFullFilename</a>	40
4.8.2.6	<a href="#">readHomePath</a>	40
4.8.2.7	<a href="#">readHomePathSlash</a>	40
4.9	<a href="#">game_config.c File Reference</a>	40
4.9.1	<a href="#">Detailed Description</a>	41
4.9.2	<a href="#">Function Documentation</a>	41
4.9.2.1	<a href="#">addConfigListFile</a>	41
4.9.2.2	<a href="#">closeListGameConfig</a>	42
4.9.2.3	<a href="#">exportConfigFile</a>	43
4.9.2.4	<a href="#">importConfigFile</a>	43
4.9.2.5	<a href="#">makeConfigListFile</a>	44
4.9.2.6	<a href="#">newConfigFile</a>	44
4.9.2.7	<a href="#">newListGameConfig</a>	45
4.9.2.8	<a href="#">readConfigFile</a>	45
4.9.2.9	<a href="#">readConfigListFile</a>	46
4.9.2.10	<a href="#">removeConfigFile</a>	46
4.9.2.11	<a href="#">removeConfigListFile</a>	47
4.10	<a href="#">game_config.h File Reference</a>	47
4.10.1	<a href="#">Detailed Description</a>	48
4.10.2	<a href="#">Macro Definition Documentation</a>	48
4.10.2.1	<a href="#">CONFIGURATION_FILE_NAME</a>	48
4.10.2.2	<a href="#">CONFIGURATION_FOLDER_NAME</a>	48
4.10.2.3	<a href="#">STRING_CHECK_GAME_CONFIG</a>	48
4.10.3	<a href="#">Function Documentation</a>	49
4.10.3.1	<a href="#">addConfigListFile</a>	49
4.10.3.2	<a href="#">closeListGameConfig</a>	50
4.10.3.3	<a href="#">exportConfigFile</a>	50
4.10.3.4	<a href="#">importConfigFile</a>	51
4.10.3.5	<a href="#">makeConfigListFile</a>	51
4.10.3.6	<a href="#">newConfigFile</a>	52
4.10.3.7	<a href="#">newListGameConfig</a>	52
4.10.3.8	<a href="#">readConfigFile</a>	53
4.10.3.9	<a href="#">readConfigListFile</a>	53
4.10.3.10	<a href="#">removeConfigFile</a>	54
4.10.3.11	<a href="#">removeConfigListFile</a>	55
4.11	<a href="#">libcsuper.h File Reference</a>	55
4.11.1	<a href="#">Detailed Description</a>	55
4.11.2	<a href="#">Macro Definition Documentation</a>	56
4.11.2.1	<a href="#">NOT_LIBCSUPER</a>	56

4.12	main_argument.c File Reference	56
4.12.1	Detailed Description	56
4.12.2	Function Documentation	56
4.12.2.1	displayHelp	56
4.12.2.2	searchArgument	56
4.13	main_argument.h File Reference	57
4.13.1	Detailed Description	57
4.13.2	Macro Definition Documentation	58
4.13.2.1	HELP	58
4.13.2.2	OPEN_FILE	58
4.13.2.3	READ_FILE	58
4.13.2.4	STRING_HELP	58
4.13.2.5	STRING_HELP_RED	58
4.13.2.6	STRING_OPEN_FILE	58
4.13.2.7	STRING_OPEN_FILE_RED	58
4.13.2.8	STRING_READ_FILE	58
4.13.2.9	STRING_READ_FILE_RED	58
4.13.3	Function Documentation	59
4.13.3.1	displayHelp	59
4.13.3.2	searchArgument	59
4.14	preferences_files.c File Reference	59
4.14.1	Detailed Description	60
4.14.2	Function Documentation	60
4.14.2.1	changeSystemPath	60
4.14.2.2	createFileMainWidowSize	60
4.14.2.3	createFileSystemPath	61
4.14.2.4	createFileToolbarButtonPreferences	61
4.14.2.5	createPreferencesFolder	62
4.14.2.6	differeentsToolbarButtonPreferencesStruct	62
4.14.2.7	readFileMainWidowSize	62
4.14.2.8	readFileSystemPath	63
4.14.2.9	readFileToolbarButtonPreferences	63
4.14.2.10	readSystemPath	64
4.15	preferences_files.h File Reference	64
4.15.1	Detailed Description	65
4.15.2	Macro Definition Documentation	65
4.15.2.1	FILENAME_MAIN_WINDOW_SIZE	65
4.15.2.2	FILENAME_SYSTEM_PATH	65
4.15.2.3	FILENAME_TOOLBAR_BUTTON_PREFERENCES	66
4.15.2.4	PREFERENCES_FOLDER_NAME	66



4.15.3	Function Documentation	66
4.15.3.1	changeSystemPath	66
4.15.3.2	createFileMainWidowSize	66
4.15.3.3	createFileSystemPath	67
4.15.3.4	createFileToolBarButtonPreferences	67
4.15.3.5	createPreferencesFolder	68
4.15.3.6	differeentsToolBarButtonPreferencesStruct	68
4.15.3.7	readFileMainWidowSize	68
4.15.3.8	readFileSystemPath	69
4.15.3.9	readFileToolBarButtonPreferences	69
4.15.3.10	readSystemPath	70
4.16	share.c File Reference	70
4.16.1	Detailed Description	71
4.16.2	Function Documentation	71
4.16.2.1	clearScreen	71
4.16.2.2	compareFloatAscending	71
4.16.2.3	compareFloatDescending	72
4.16.2.4	integerToYesNo	72
4.16.2.5	libcsuper_initialize	72
4.16.2.6	myAlloc	72
4.16.2.7	myRealloc	73
4.16.2.8	wrongChoice	73
4.17	share.h File Reference	73
4.17.1	Detailed Description	74
4.17.2	Macro Definition Documentation	74
4.17.2.1	_	74
4.17.3	Function Documentation	74
4.17.3.1	clearScreen	74
4.17.3.2	compareFloatAscending	75
4.17.3.3	compareFloatDescending	75
4.17.3.4	integerToYesNo	75
4.17.3.5	libcsuper_initialize	75
4.17.3.6	myAlloc	76
4.17.3.7	myRealloc	77
4.17.3.8	wrongChoice	77
	<b>Index</b>	<b>78</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>	6
<a href="#">list_game_config</a>	8
<a href="#">main_window_size</a>	8
<a href="#">toolbar_button_preferences_struct</a>	9



## 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 . . . . .	13
<a href="#">csu_files.h</a>	Files management . . . . .	15
<a href="#">csu_struct.c</a>	Management of the csu files . . . . .	18
<a href="#">csu_struct.h</a>	Management of the csu files header . . . . .	24
<a href="#">file.c</a>	Files function of libcsuper . . . . .	30
<a href="#">file.h</a>	Header for the files function of libcsuper . . . . .	32
<a href="#">filename.c</a>	Essential function of libcsuper . . . . .	35
<a href="#">filename.h</a>	Header for the essential function of libcsuper . . . . .	38
<a href="#">game_config.c</a>	Game configuration . . . . .	40
<a href="#">game_config.h</a>	Game configurations . . . . .	47
<a href="#">libcsuper.h</a>	Inclusion of all header files of libcsuper . . . . .	55
<a href="#">main_argument.c</a>	Begin csuper . . . . .	56
<a href="#">main_argument.h</a>	Begin csuper . . . . .	57
<a href="#">preferences_files.c</a>	Function which store preferences into files . . . . .	59
<a href="#">preferences_files.h</a>	Prototypes des fonctions qui l'emrankment des fichiers sauvegardes . . . . .	64
<a href="#">share.c</a>	Essential function of libcsuper . . . . .	70
<a href="#">share.h</a>	Header for the essential function of libcsuper . . . . .	73



## 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

#### 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\\_based](#)
- 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\_based

Is 1 if this is a turn-based game, 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 Detailed Description

Represent a list of game configuration

#### 3.3.2 Field Documentation

##### 3.3.2.1 char\*\* name\_game\_config

The list of the game configuration.

##### 3.3.2.2 int nb\_config

Number of game configuration.

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

- [game\\_config.h](#)

### 3.4 main\_window\_size Struct Reference

```
#include <preferences_files.h>
```

#### Data Fields

- int [width](#)
- int [height](#)
- int [is\\_maximize](#)

#### 3.4.1 Detailed Description

All component of the man window size

#### 3.4.2 Field Documentation

##### 3.4.2.1 int height

The height of the main window

##### 3.4.2.2 int is\_maximize

Said if the main window is maximize or not

#### 3.4.2.3 int width

The width of the main window

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

- [preferences\\_files.h](#)

## 3.5 toolbar\_button\_preferences\_struct Struct Reference

```
#include <preferences_files.h>
```

### Data Fields

- int [new](#)
- int [open](#)
- int [save\\_as](#)
- int [separator\\_1](#)
- int [undo](#)
- int [redo](#)
- int [separator\\_2](#)
- int [cut](#)
- int [copy](#)
- int [paste](#)
- int [delete](#)
- int [separator\\_3](#)
- int [properties](#)
- int [separator\\_4](#)
- int [preferences](#)
- int [game\\_configuration\\_preferences](#)
- int [toolbar\\_button\\_preferences](#)
- int [separator\\_5](#)
- int [about](#)

### 3.5.1 Detailed Description

Represent the toolbar button preferences

### 3.5.2 Field Documentation

#### 3.5.2.1 int about

The about button

#### 3.5.2.2 int copy

The copy button

#### 3.5.2.3 int cut

The cut button

#### 3.5.2.4 int delete

The delete button

#### 3.5.2.5 int game\_configuration\_preferences

The game configuration preferences button

#### 3.5.2.6 int new

The new button

#### 3.5.2.7 int open

The open button

#### 3.5.2.8 int paste

The paste button

#### 3.5.2.9 int preferences

The preferences button

#### 3.5.2.10 int properties

The properties button

#### 3.5.2.11 int redo

The redo button

#### 3.5.2.12 int save\_as

The save\_as button

#### 3.5.2.13 int separator\_1

The separator 1

#### 3.5.2.14 int separator\_2

The separator 2

#### 3.5.2.15 int separator\_3

The separator 3

#### 3.5.2.16 int separator\_4

The separator 4

#### 3.5.2.17 int separator\_5

The separator 5

#### 3.5.2.18 int toolbar\_button\_preferences

The toolbar button preferences button

#### 3.5.2.19 int undo

The undo button

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

- [preferences\\_files.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)
- bool [writeCsuFile](#) (char \*file\_name, [csuStruct](#) \*ptr\_csu\_struct)
- bool [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.

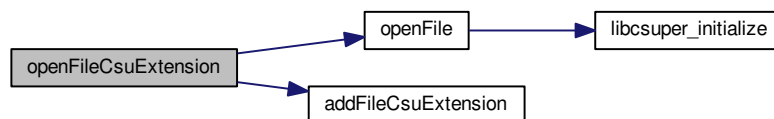
**Parameters**

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.

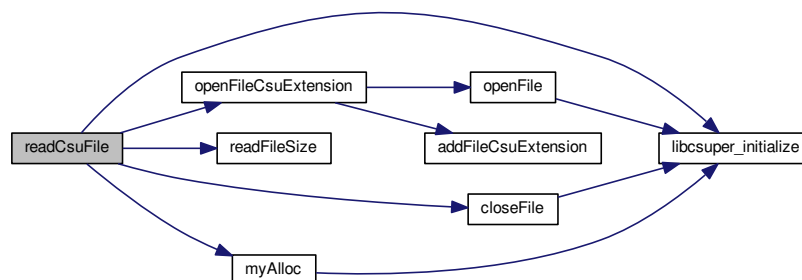
**Parameters**

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 bool writeCsuFile ( char \* file\_name, csuStruct \* ptr\_csu\_struct )**

Write a csu file



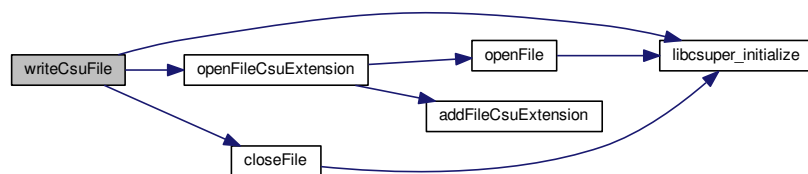
## Parameters

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 bool writeFileNewTurn ( char \* file\_name, csuStruct \* ptr\_csu\_struct )

Update the file with the new scores

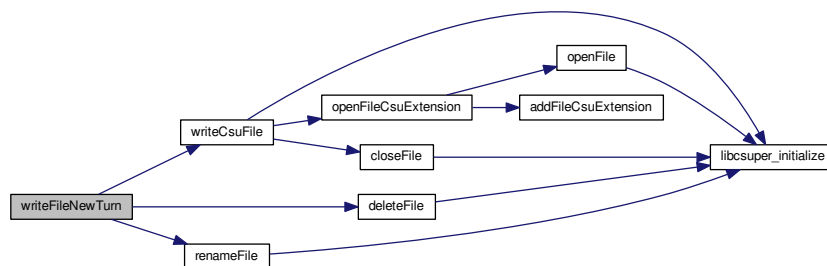
## Parameters

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

## Macros

- `#define SIZE_MAX_FILE_NAME 1024`
- `#define FILE_EXTENSION "csu"`
- `#define STRING_CHECK_CSU_FILE "CompteurScoreUniversel"`

## Functions

- `FILE * openFileCsuExtension (char file_name[], char mode[])`
- `csuStruct * readCsuFile (char *file_name)`
- `bool writeCsuFile (char *file_name, csuStruct *ptr_csu_struct)`
- `bool 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 1024`

Define the size maximum of a filename to 1024

#### 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.

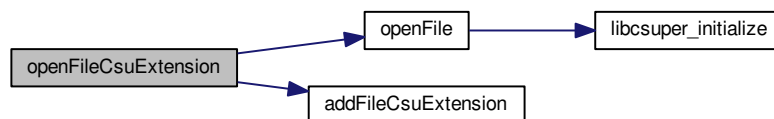
## Parameters

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.

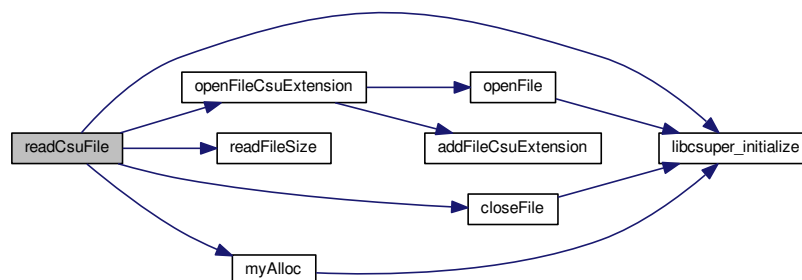
## Parameters

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 bool writeCsuFile ( char \* file\_name, csuStruct \* ptr\_csu\_struct )

Write a csu file

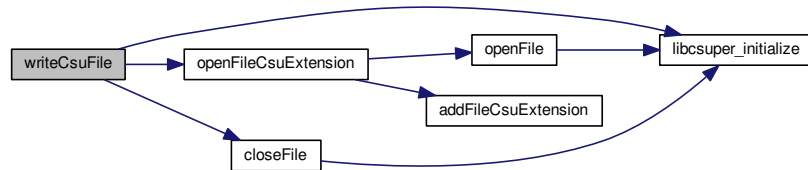
## Parameters

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 bool writeFileNewTurn ( char \* file\_name, csuStruct \* ptr\_csu\_struct )

Update the file with the new scores

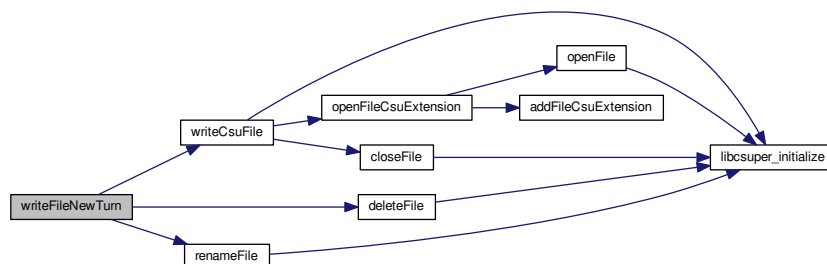
## Parameters

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`)
- int `searchIndexFromPosition` (`csuStruct *ptr_csu_struct`, int position, int \*nb)
- void `addDistributorCsuStruct` (`csuStruct *ptr_csu_struct`, char \*distributor\_name)
- bool `exceedMaxNumber` (`csuStruct *ptr_csu_struct`)
- int `maxNbTurn` (`csuStruct *ptr_csu_struct`)
- int `searchPlayerIndex` (`csuStruct *ptr_csu_struct`, char \*player\_name)
- bool `differentsPlayerName` (`csuStruct *ptr_csu_struct`)
- `csuStruct * copyCsuStruct` (`csuStruct *ptr_csu_struct`)

### 4.3.1 Detailed Description

Management of the csu files.

#### Author

Remi BERTHO

#### Date

15/06/14

#### Version

4.0.0

### 4.3.2 Function Documentation

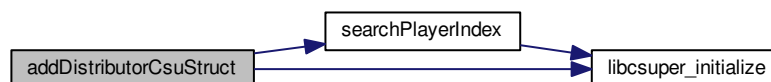
#### 4.3.2.1 void addDistributorCsuStruct ( `csuStruct * ptr_csu_struct`, char \* *distributor\_name* )

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 **csuStruct \* copyCsuStruct ( csuStruct \* ptr\_csu\_struct )**

Copy a csu structure

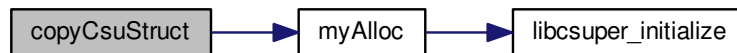
## Parameters

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

## Returns

a pointer on the new csu structure

Here is the call graph for this function:

4.3.2.4 **bool differentsPlayerName ( csuStruct \* ptr\_csu\_struct )**

Search the index of a person

## Parameters

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

## Returns

true if all player names are different, false otherwise

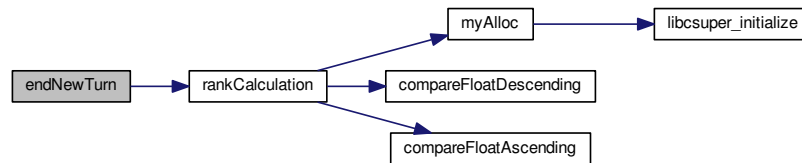
4.3.2.5 **void endNewTurn ( csuStruct \* ptr\_csu\_struct, int index\_player )**

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>	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:



#### 4.3.2.6 bool 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

#### 4.3.2.7 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

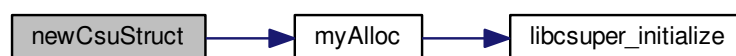
#### 4.3.2.8 csuStruct \* newCsuStruct ( float nb\_player, game\_config config )

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:



4.3.2.9 void rankCalculation ( csuStruct \* ptr\_csu\_struct )

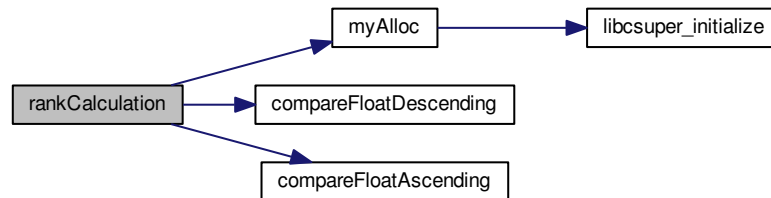
Calculate the rank



## Parameters

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

Here is the call graph for this function:



#### 4.3.2.10 int searchIndexFromPosition ( csuStruct \* ptr\_csu\_struct, int position, int \* nb )

Search the index in the array of the person who is the 'position' position

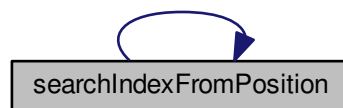
## Parameters

in, out	*ptr_csu_struct	a pointer on a <a href="#">csuStruct</a>
in, out	position	the position
in, out	nb	the nbth player who have the position will be selected

## Returns

the index or NULL if the position doesn't exist

Here is the call graph for this function:



#### 4.3.2.11 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:



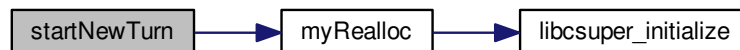
#### 4.3.2.12 void startNewTurn ( [csuStruct](#) \* *ptr\_csu\_struct*, int *index\_player* )

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:



## 4.4 [csu\\_struct.h](#) File Reference

Management of the csu files header.

```

#include <time.h>
#include <float.h>
#include "share.h"
#include "file.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`)
- int `searchIndexFromPosition` (`csuStruct *ptr_csu_struct`, int position, int \*nb)
- void `addDistributorCsuStruct` (`csuStruct *ptr_csu_struct`, char \*distributor\_name)
- bool `exceedMaxNumber` (`csuStruct *ptr_csu_struct`)
- int `maxNbTurn` (`csuStruct *ptr_csu_struct`)
- int `searchPlayerIndex` (`csuStruct *ptr_csu_struct`, char \*player\_name)
- bool `differentsPlayerName` (`csuStruct *ptr_csu_struct`)
- `csuStruct * copyCsuStruct` (`csuStruct *ptr_csu_struct`)

### 4.4.1 Detailed Description

Management of the csu files header.

#### Author

Remi BERTHO

#### Date

16/06/14

#### Version

4.0.0

### 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

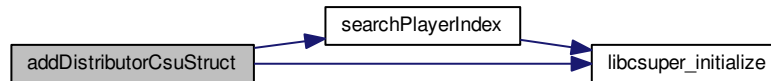
#### 4.4.3.1 void `addDistributorCsuStruct` ( `csuStruct * ptr_csu_struct`, char \* `distributor_name` )

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

Here is the call graph for this function:



#### 4.4.3.2 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>
---------	------------------------	--

#### 4.4.3.3 csuStruct\* copyCsuStruct ( csuStruct \* ptr\_csu\_struct )

Copy a csu structure

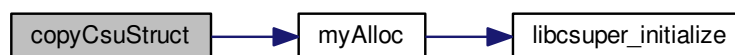
## Parameters

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

## Returns

a pointer on the new csu structure

Here is the call graph for this function:



#### 4.4.3.4 bool differsPlayerName ( csuStruct \* ptr\_csu\_struct )

Search the index of a person

## Parameters

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

## Returns

true if all player names are different, false otherwise

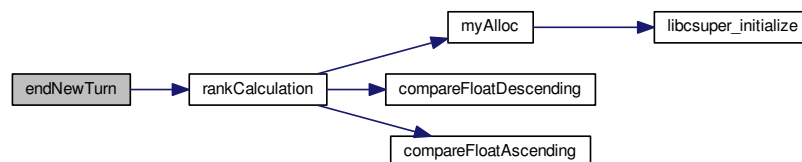
## 4.4.3.5 void endNewTurn ( csuStruct \* ptr\_csu\_struct, int index\_player )

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>	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:



## 4.4.3.6 bool 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

## 4.4.3.7 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

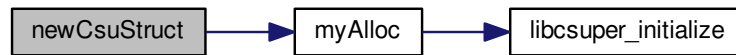
## 4.4.3.8 csuStruct\* newCsuStruct ( float nb\_player, game\_config config )

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:



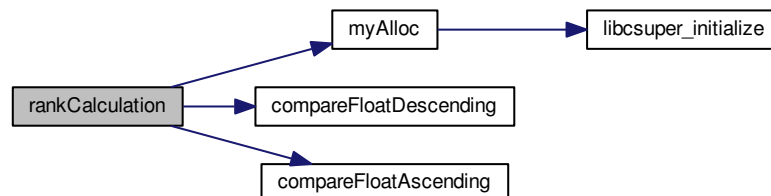
#### 4.4.3.9 void rankCalculation ( csuStruct \* ptr\_csu\_struct )

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:



#### 4.4.3.10 int searchIndexFromPosition ( csuStruct \* ptr\_csu\_struct, int position, int \* nb )

Search the index in the array of the person who is the 'position' position

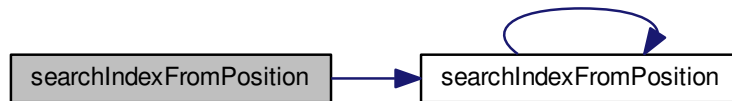
## Parameters

in, out	<i>*ptr_csu_struct</i>	a pointer on a <a href="#">csuStruct</a>
in, out	<i>position</i>	the position
in, out	<i>nb</i>	the nbth player who have the position will be selected

**Returns**

the index or NULL if the position doesn't exist

Here is the call graph for this function:



#### 4.4.3.11 `int searchPlayerIndex ( csuStruct * ptr_csu_struct, char * player_name )`

Search the index of a person

**Parameters**

in	<code>*player_name</code>	the name of the player
in	<code>*ptr_csu_struct</code>	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:



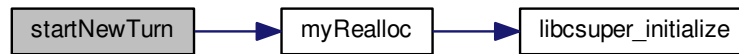
#### 4.4.3.12 `void startNewTurn ( csuStruct * ptr_csu_struct, int index_player )`

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

**Parameters**

in, out	<code>*ptr_csu_struct</code>	a pointer on a <a href="#">csuStruct</a>
in, out	<code>index_player</code>	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.c File Reference

Files function of libcsuper.

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

### Functions

- FILE \* [openFile](#) (char file\_name[], char mode[])
- int [closeFile](#) (FILE \*ptr\_file)
- int [readFileSize](#) (FILE \*ptr\_file)
- bool [deleteFile](#) (char \*file\_name)
- bool [renameFile](#) (char \*old\_name, char \*new\_name)

#### 4.5.1 Detailed Description

Files function of libcsuper.

##### Author

Remi BERTHO

##### Date

05/07/14

##### Version

4.0.1

#### 4.5.2 Function Documentation

##### 4.5.2.1 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:

**4.5.2.2 bool 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:

**4.5.2.3 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:

**4.5.2.4 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

**4.5.2.5 bool 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:

**4.6 file.h File Reference**

Header for the files function of libcsuper.

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

## Functions

- FILE \* [openFile](#) (char nome[], char mode[])
- int [closeFile](#) (FILE \*ptr\_file)
- int [readFileSize](#) (FILE \*ptr\_file)
- bool [deleteFile](#) (char \*file\_name)
- bool [renameFile](#) (char \*old\_name, char \*new\_name)

### 4.6.1 Detailed Description

Header for the files function of libcsuper.

#### Author

Remi BERTHO

#### Date

05/07/14

#### Version

4.0.1

### 4.6.2 Function Documentation

#### 4.6.2.1 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:



#### 4.6.2.2 bool 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:

**4.6.2.3 FILE\* openFile ( char *file\_name*[], char *mode*[] )**

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

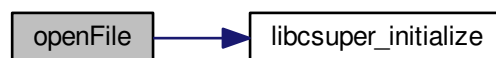
**Parameters**

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:

**4.6.2.4 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

**4.6.2.5 bool renameFile ( char \* *old\_name*, char \* *new\_name* )**

Rename a file.

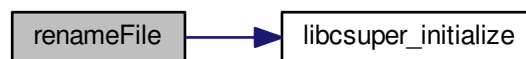
**Parameters**

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:

**4.7 filename.c File Reference**

Essential function of libcsuper.

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

**Functions**

- void [addFileCsuExtension](#) (char \*file\_name)
- bool [getFolderFromFilename](#) (char \*file\_name\_to\_folder)
- bool [getSimpleFilenameFromFullFilename](#) (char \*full\_filename, char \*simple\_filename)
- bool [checkPath](#) (char \*path)
- bool [checkFilename](#) (char \*filename, char \*folder)
- void [readHomePath](#) (char \*path)
- void [readHomePathSlash](#) (char \*path)

**4.7.1 Detailed Description**

Essential function of libcsuper.

**Author**

Remi BERTHO

**Date**

05/07/14

**Version**

4.0.1

**4.7.2 Function Documentation****4.7.2.1 void addFileCsuExtension ( char \* *file\_name* )**

Add the csu file extension

**Parameters**

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

**4.7.2.2 bool checkFilename ( char \* *filename*, char \* *folder* )**

Test if the filename is valid

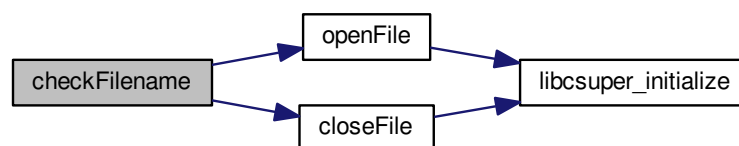
**Parameters**

in, out	* <i>filename</i>	the filename
in, out	* <i>folder</i>	the folder where the filename will be tested, may be ""

**Returns**

true if the filename is valid OK, false otherwise

Here is the call graph for this function:

**4.7.2.3 bool checkPath ( char \* *path* )**

Test if the path is valid

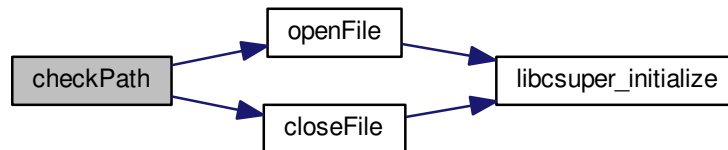
## Parameters

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

## Returns

true if the path is valid OK, false otherwise

Here is the call graph for this function:

4.7.2.4 bool getFolderFromFilename ( char \* *file\_name\_to\_folder* )

Transform a filename into his folder

## Parameters

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

## Returns

true if everything is OK, false otherwise

4.7.2.5 bool getSimpleFilenameFromFullFilename ( char \* *full\_filename*, char \* *simple\_filename* )

Transform a full filename into his simple filename (without the folder)

## Parameters

in	<i>full_filename</i>	the full filename
in	<i>simple_filename</i>	the full filename

## Returns

true if everything is OK, false otherwise

4.7.2.6 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
----------------	-------------	----------

4.7.2.7 void readHomePathSlash ( char \* *path* )

## 4.8 filename.h File Reference

Header for the essential function of libcsuper.

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

**Functions**

- void [addFileCsuExtension](#) (char \*file\_name)
- bool [getFolderFromFilename](#) (char \*file\_name\_to\_folder)
- bool [getSimpleFilenameFromFullFilename](#) (char \*full\_filename, char \*simple\_filename)
- bool [checkPath](#) (char \*path)
- bool [checkFilename](#) (char \*filename, char \*folder)
- void [readHomePath](#) (char \*path)
- void [readHomePathSlash](#) (char \*path)

### 4.8.1 Detailed Description

Header for the essential function of libcsuper.

**Author**

Remi BERTHO

**Date**

05/07/14

**Version**

4.0.1

### 4.8.2 Function Documentation

4.8.2.1 void addFileCsuExtension ( char \* *file\_name* )

Add the csu file extension



## Parameters

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

4.8.2.2 bool checkFilename ( char \* *filename*, char \* *folder* )

Test if the filename is valid

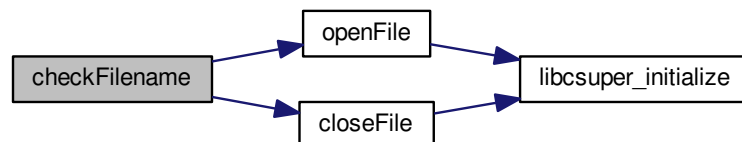
## Parameters

in, out	* <i>filename</i>	the filename
in, out	* <i>folder</i>	the folder where the filename will be tested, may be ""

## Returns

true if the filename is valid OK, false otherwise

Here is the call graph for this function:

4.8.2.3 bool checkPath ( char \* *path* )

Test if the path is valid

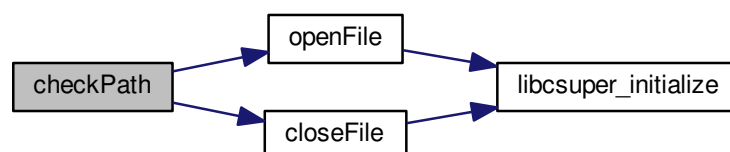
## Parameters

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

## Returns

true if the path is valid OK, false otherwise

Here is the call graph for this function:



#### 4.8.2.4 bool getFolderFromFilename ( char \* *file\_name\_to\_folder* )

Transform a filename into his folder

##### Parameters

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

##### Returns

true if everything is OK, false otherwise

#### 4.8.2.5 bool getSimpleFilenameFromFullFilename ( char \* *full\_filename*, char \* *simple\_filename* )

Transform a full filename into his simple filename (without the folder)

##### Parameters

in	<i>full_filename</i>	the full filename
in	<i>simple_filename</i>	the full filename

##### Returns

true if everything is OK, false otherwise

#### 4.8.2.6 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
---------	-------------	----------

#### 4.8.2.7 void readHomePathSlash ( char \* *path* )

## 4.9 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)
- bool [makeConfigListFile](#) (char \*home\_path)
- [list\\_game\\_config](#) \* [readConfigListFile](#) (char \*home\_path)
- bool [addConfigListFile](#) (char \*new\_config\_name, char \*home\_path)
- bool [removeConfigListFile](#) (int index\_delete, [list\\_game\\_config](#) \*ptr\_list\_config, char \*home\_path)

- bool [newConfigFile](#) ([game\\_config](#) config, char \*home\_path)
- bool [removeConfigFile](#) (char \*config\_name, char \*home\_path)
- bool [readConfigFile](#) (int index\_read, [list\\_game\\_config](#) \*ptr\_list\_config, [game\\_config](#) \*ptr\_config, char \*home\_path)
- bool [exportConfigFile](#) (char \*home\_path, char \*file\_name)
- bool [importConfigFile](#) (char \*home\_path, char \*file\_name)

### 4.9.1 Detailed Description

Game configuration.

#### Author

Remi BERTHO

#### Date

29/04/14

#### Version

2.4.0

### 4.9.2 Function Documentation

#### 4.9.2.1 bool addConfigListFile ( char \* new\_config\_name, char \* home\_path )

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

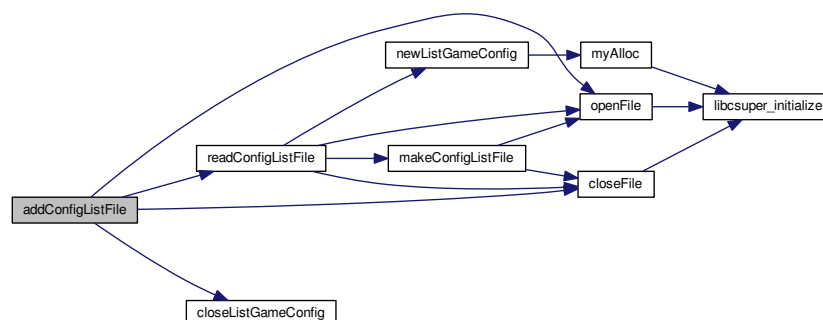
#### Parameters

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:



4.9.2.2 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
----	-------------------------	---

## 4.9.2.3 bool exportConfigFile ( char \* home\_path, char \* file\_name )

Export all config file into a file.

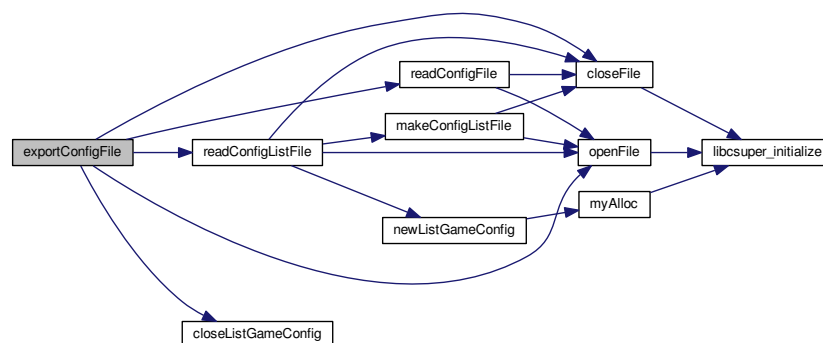
## Parameters

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:



## 4.9.2.4 bool importConfigFile ( char \* home\_path, char \* file\_name )

Import all config file from a file.

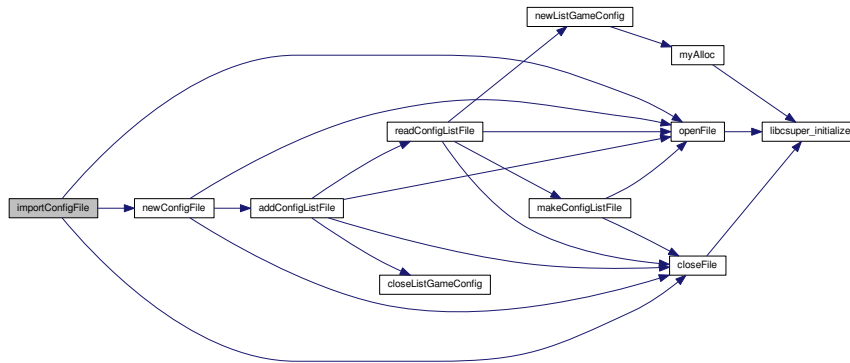
## Parameters

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:



#### 4.9.2.5 bool makeConfigListFile ( char \* home\_path )

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

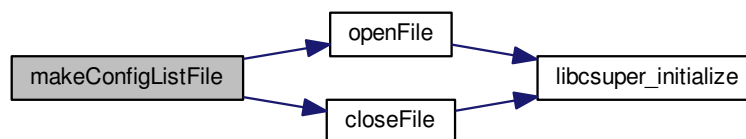
## Parameters

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:



#### 4.9.2.6 bool newConfigFile ( game\_config config, char \* home\_path )

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

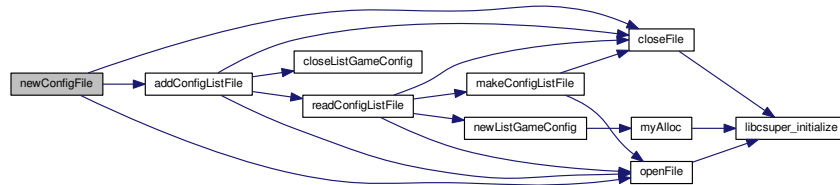
## Parameters

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:



## 4.9.2.7 list\_game\_config \* newListGameConfig ( int nb\_config )

Create a list of game configuration.

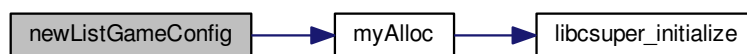
## Parameters

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

## Returns

une [list\\_game\\_config](#)

Here is the call graph for this function:



## 4.9.2.8 bool readConfigFile ( int index\_read, list\_game\_config \* ptr\_list\_config, game\_config \* ptr\_config, char \* home\_path )

Read a game configuration file.

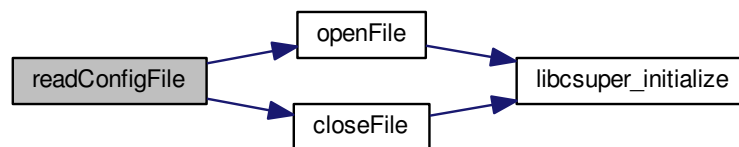
## Parameters

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:



#### 4.9.2.9 list\_game\_config \* readConfigListFile ( char \* home\_path )

Read the file which contain the list of game configuration.

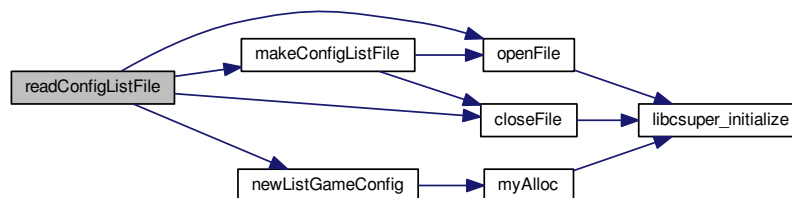
#### Parameters

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

#### Returns

a [list\\_game\\_config](#)

Here is the call graph for this function:



#### 4.9.2.10 bool 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

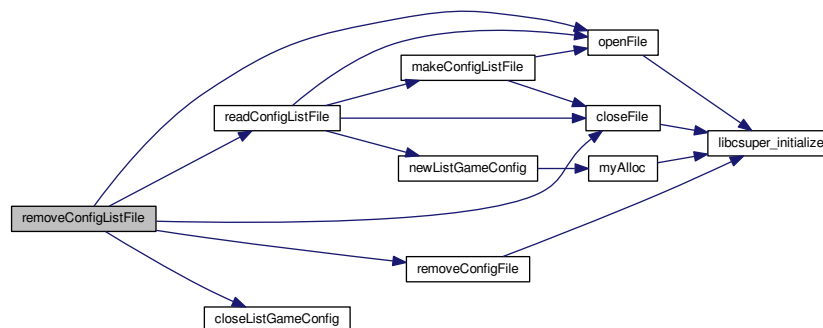
## Returns

true if everything is OK, false otherwise

Here is the call graph for this function:

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

Here is the call graph for this function:



## 4.10 game\_config.h File Reference

Game configurations.

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

## Data Structures

- struct [list\\_game\\_config](#)

## Macros

- `#define CONFIGURATION_FOLDER_NAME "config"`
- `#define CONFIGURATION_FILE_NAME "configuration"`
- `#define STRING_CHECK_GAME_CONFIG "Csuper_Game_Configuration"`

## Functions

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

### 4.10.1 Detailed Description

Game configurations.

#### Author

Remi BERTHO

#### Date

29/04/14

#### Version

2.4.0

### 4.10.2 Macro Definition Documentation

#### 4.10.2.1 `#define CONFIGURATION_FILE_NAME "configuration"`

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

#### 4.10.2.2 `#define CONFIGURATION_FOLDER_NAME "config"`

Define the name of the folder which contain the game configurations

#### 4.10.2.3 `#define STRING_CHECK_GAME_CONFIG "Csuper_Game_Configuration"`

String for checking if the file is game configuration file.

### 4.10.3 Function Documentation

#### 4.10.3.1 bool addConfigListFile ( char \* *new\_config\_name*, char \* *home\_path* )

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

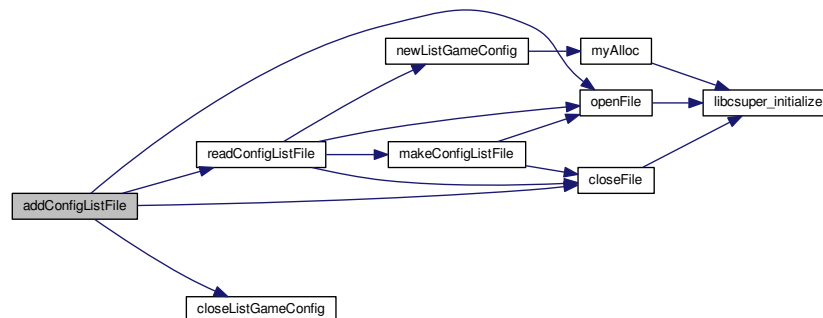
**Parameters**

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:



#### 4.10.3.2 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
----	-------------------------	---

#### 4.10.3.3 bool exportConfigFile ( char \* home\_path, char \* file\_name )

Export all config file into a file.

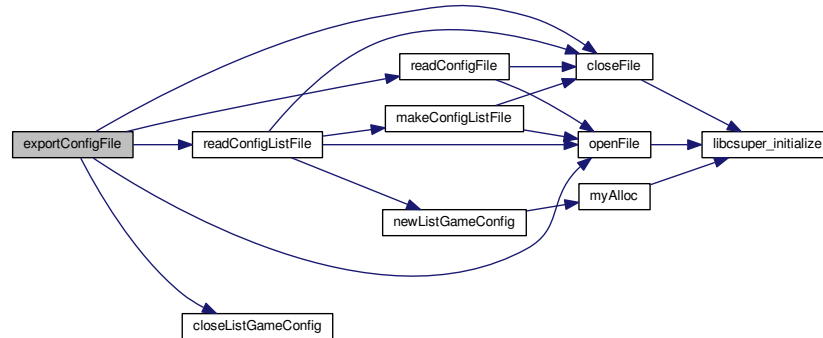
**Parameters**

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:



#### 4.10.3.4 bool importConfigFile ( char \* home\_path, char \* file\_name )

Import all config file from a file.

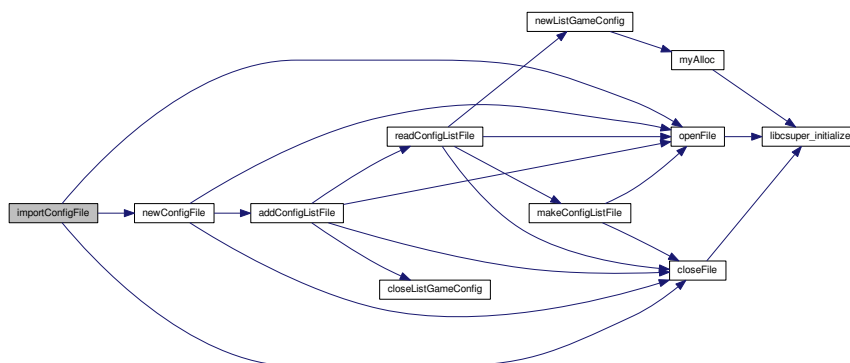
## Parameters

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:



#### 4.10.3.5 bool makeConfigListFile ( char \* home\_path )

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

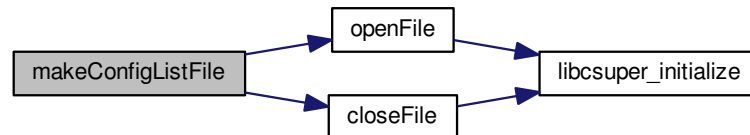
**Parameters**

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:



#### 4.10.3.6 bool newConfigFile ( game\_config config, char \* home\_path )

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

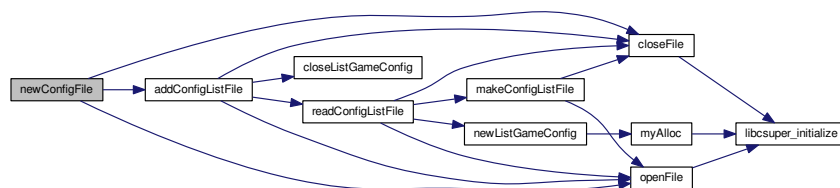
**Parameters**

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:



#### 4.10.3.7 list\_game\_config\* newListGameConfig ( int nb\_config )

Create a list of game configuration.

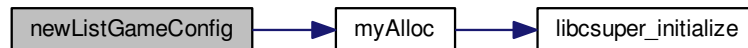
## Parameters

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

## Returns

une [list\\_game\\_config](#)

Here is the call graph for this function:



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

Read a game configuration file.

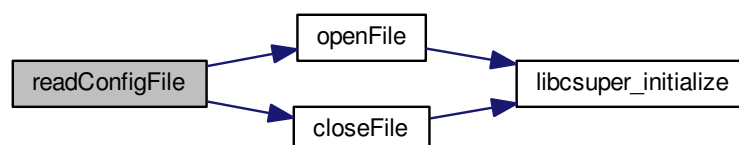
## Parameters

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:



**4.10.3.9** `list_game_config* readConfigListFile ( char * home_path )`

Read the file which contain the list of game configuration.

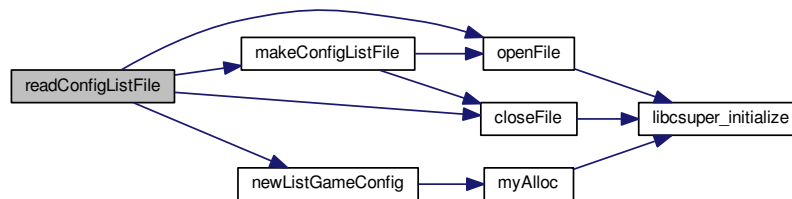
## Parameters

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

## Returns

a [list\\_game\\_config](#)

Here is the call graph for this function:



#### 4.10.3.10 bool 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

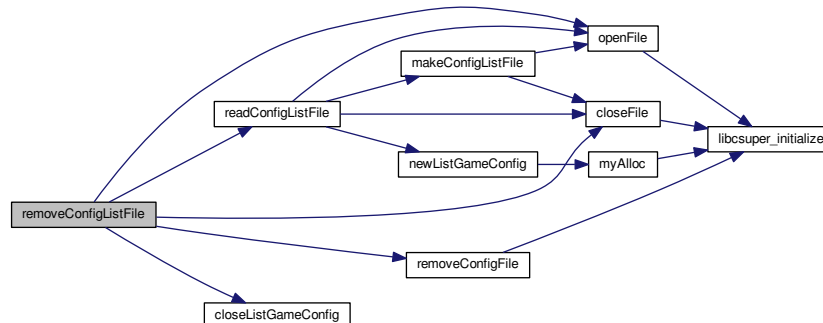
Here is the call graph for this function:





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

Here is the call graph for this function:



## 4.11 libcsuper.h File Reference

Inclusion of all header files of libcsuper.

```
#include "csu_struct.h"
#include "share.h"
#include "csu_files.h"
#include "preferences_files.h"
#include "main_argument.h"
#include "game_config.h"
#include "file.h"
#include "filename.h"
```

### Macros

- `#define NOT_LIBCSUPER`

### 4.11.1 Detailed Description

Inclusion of all header files of libcsuper.

#### Author

Remi BERTHO

#### Date

25/08/14

#### Version

4.0.2

## 4.11.2 Macro Definition Documentation

### 4.11.2.1 #define NOT\_LIBCSUPER

Define that we don't compile libcsuper

## 4.12 main\_argument.c File Reference

Begin csuper.

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

### Functions

- bool [searchArgument](#) (int argc, char \*argv[], int \*function, int \*file\_place)
- void [displayHelp](#) ()

### 4.12.1 Detailed Description

Begin csuper.

#### Author

Remi BERTHO

#### Date

16/04/14

#### Version

2.2.0

### 4.12.2 Function Documentation

#### 4.12.2.1 void displayHelp ( )

Display the help

Here is the call graph for this function:



#### 4.12.2.2 bool searchArgument ( int argc, char \* argv[], int \* function, int \* file\_place )

Search the argument passed to the main function

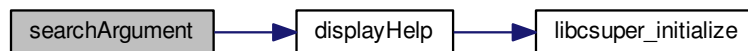
## Parameters

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:



## 4.13 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

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

## 4.13.1 Detailed Description

Begin csuper.

## Author

Remi BERTHO

**Date**

16/04/14

**Version**

2.2.0

**4.13.2 Macro Definition Documentation****4.13.2.1 #define HELP 2**

Define the call help to 2

**4.13.2.2 #define OPEN\_FILE 1**

Define the call to read a file to 1

**4.13.2.3 #define READ\_FILE 0**

Define the call to read a file to 0

**4.13.2.4 #define STRING\_HELP "--help"**

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

**4.13.2.5 #define STRING\_HELP\_RED "-h"**

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

**4.13.2.6 #define STRING\_OPEN\_FILE "--open"**

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

**4.13.2.7 #define STRING\_OPEN\_FILE\_RED "-o"**

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

**4.13.2.8 #define STRING\_READ\_FILE "--read"**

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

**4.13.2.9 #define STRING\_READ\_FILE\_RED "-r"**

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

### 4.13.3 Function Documentation

#### 4.13.3.1 void displayHelp ( )

Display the help

Here is the call graph for this function:



#### 4.13.3.2 bool searchArgument ( int argc, char \* argv[], int \* function, int \* file\_place )

Search the argument passed to the main function

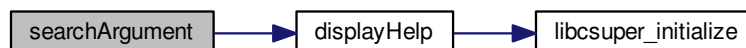
##### Parameters

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:



## 4.14 preferences\_files.c File Reference

Function which store preferences into files.

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

### Functions

- void [createPreferencesFolder](#) (char \*home\_path)
- bool [createFileToolbarButtonPreferences](#) (char \*home\_path, [toolbar\\_button\\_preferences\\_struct](#) toolbar)

- bool [readFileToolBarButtonPreferences](#) (char \*home\_path, toolbar\_button\_preferences\_struct \*toolbar)
- bool [differeentsToolBarButtonPreferencesStruct](#) (toolbar\_button\_preferences\_struct toolbar1, toolbar\_button\_preferences\_struct toolbar2)
- bool [createFileMainWidowSize](#) (char \*home\_path, main\_window\_size size)
- bool [readFileMainWidowSize](#) (char \*home\_path, main\_window\_size \*size)
- bool [createFileSystemPath](#) ()
- bool [readFileSystemPath](#) (char \*file\_name)
- bool [readSystemPath](#) (char \*file\_name)
- bool [changeSystemPath](#) (char \*new\_path)

#### 4.14.1 Detailed Description

Function which store preferences into files.

##### Author

Remi BERTHO

##### Date

05/07/14

##### Version

4.0.1

#### 4.14.2 Function Documentation

##### 4.14.2.1 bool changeSystemPath ( char \* new\_path )

Change the system path

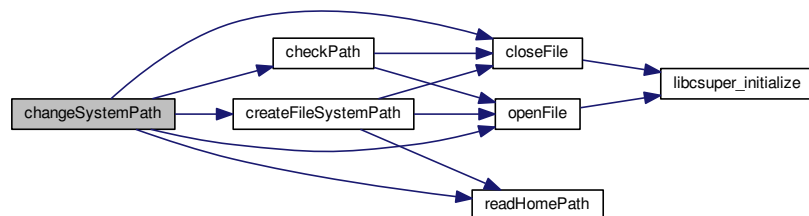
##### Parameters

in, out	*new_path	the new path
---------	-----------	--------------

##### Returns

true if everything is OK, false otherwise

Here is the call graph for this function:



##### 4.14.2.2 bool createFileMainWidowSize ( char \* home\_path, main\_window\_size size )

Create the file which contain the main window size

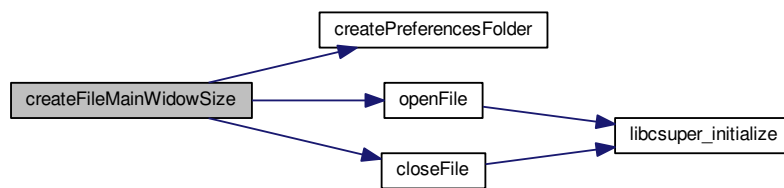
## Parameters

in	<i>home_path</i>	the path to the home directory
in	<i>size</i>	the size of the main window

## Returns

true if everything is OK, false otherwise

Here is the call graph for this function:



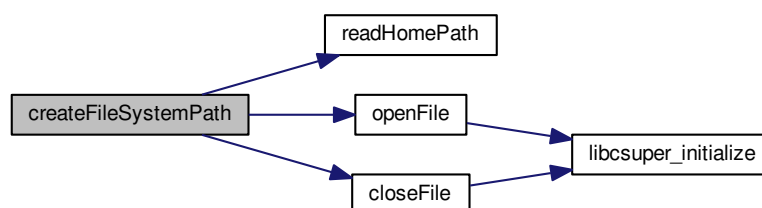
## 4.14.2.3 bool createFileSystemPath ( )

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:



## 4.14.2.4 bool createFileToolBarButtonPreferences ( char \* home\_path, toolbar\_button\_preferences\_struct toolbar )

Create the file which contain the preferences for the toolbar button

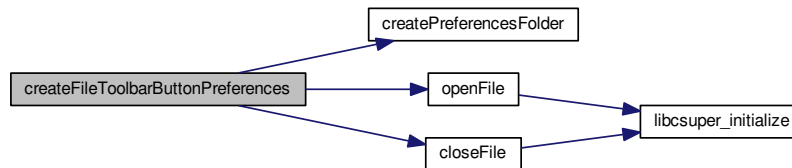
**Parameters**

in	<i>home_path</i>	the path to the home directory
in	<i>toolbar</i>	the toolbar button preferences

**Returns**

true if everything is OK, false otherwise

Here is the call graph for this function:



#### 4.14.2.5 void createPreferencesFolder ( char \* *home\_path* )

Create the folder which contain all preferences

**Parameters**

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

#### 4.14.2.6 bool differsToolbarButtonPreferencesStruct ( toolbar\_button\_preferences\_struct *toolbar1*, toolbar\_button\_preferences\_struct *toolbar2* )

Test if the two toolbar button preferences are different

**Parameters**

in	<i>toolbar1</i>	the first toolbar button preferences
in	<i>toolbar2</i>	the second toolbar button preferences

**Returns**

true if everything is OK, false otherwise

#### 4.14.2.7 bool readFileMainWidowSize ( char \* *home\_path*, main\_window\_size \* *size* )

Read the file which contain the main window size

**Parameters**

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

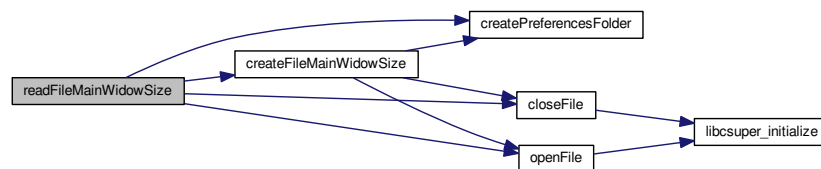


<i>in</i>	<i>size</i>	the size of the main window
-----------	-------------	-----------------------------

**Returns**

true if everything is OK, false otherwise

Here is the call graph for this function:

**4.14.2.8 bool readFileSystemPath ( char \* *file\_name* )**

Read the system path and the path read to the filename

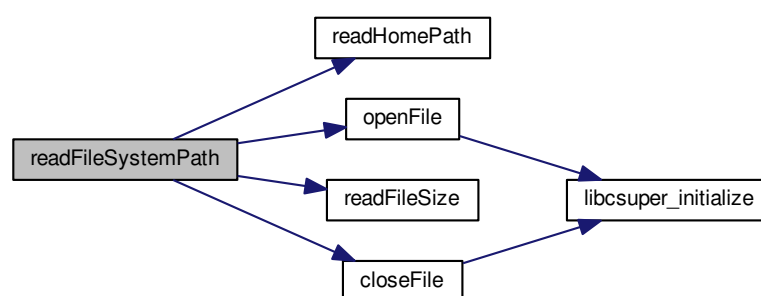
**Parameters**

<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:

**4.14.2.9 bool readFileToolBarButtonPreferences ( char \* *home\_path*, toolbar\_button\_preferences\_struct \* *toolbar* )**

Read the file which contain the preferences for the toolbar button

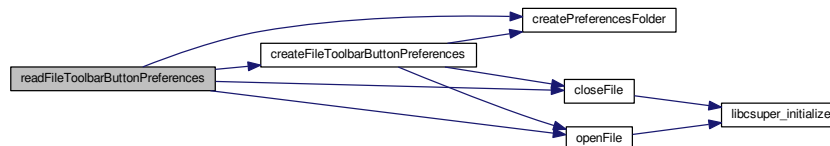
**Parameters**

in	<i>home_path</i>	the path to the home directory
in	<i>toolbar</i>	the toolbar button preferences

**Returns**

true if everything is OK, false otherwise

Here is the call graph for this function:

**4.14.2.10 bool readSystemPath ( char \* file\_name )**

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

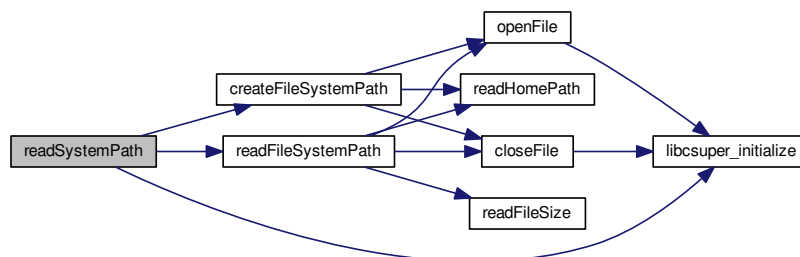
**Parameters**

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

**Returns**

true if everything is OK, false otherwise

Here is the call graph for this function:

**4.15 preferences\_files.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"
#include "filename.h"

```

## Data Structures

- struct [toolbar\\_button\\_preferences\\_struct](#)
- struct [main\\_window\\_size](#)

## Macros

- #define [FILENAME\\_SYSTEM\\_PATH](#) "system\_path.txt"
- #define [FILENAME\\_TOOLBAR\\_BUTTON\\_PREFERENCES](#) "toolbar\_button\_preferences.txt"
- #define [FILENAME\\_MAIN\\_WINDOW\\_SIZE](#) "main\_window\_size.txt"
- #define [PREFERENCES\\_FOLDER\\_NAME](#) ".csuper"

## Functions

- void [createPreferencesFolder](#) (char \*home\_path)
- bool [createFileToolbarButtonPreferences](#) (char \*home\_path, [toolbar\\_button\\_preferences\\_struct](#) toolbar)
- bool [readFileToolbarButtonPreferences](#) (char \*home\_path, [toolbar\\_button\\_preferences\\_struct](#) \*toolbar)
- bool [differeentsToolbarButtonPreferencesStruct](#) ([toolbar\\_button\\_preferences\\_struct](#) toolbar1, [toolbar\\_button\\_preferences\\_struct](#) toolbar2)
- bool [createFileMainWidowSize](#) (char \*home\_path, [main\\_window\\_size](#) size)
- bool [readFileMainWidowSize](#) (char \*home\_path, [main\\_window\\_size](#) \*size)
- bool [createFileSystemPath](#) ()
- bool [readFileSystemPath](#) (char \*file\_name)
- bool [readSystemPath](#) (char \*file\_name)
- bool [changeSystemPath](#) (char \*new\_path)

### 4.15.1 Detailed Description

Prototypes des fonctions qui l'emrankment des fichiers sauvegardes.

#### Author

Remi BERTHO

#### Date

05/07/14

#### Version

4.0.1

### 4.15.2 Macro Definition Documentation

4.15.2.1 #define [FILENAME\\_MAIN\\_WINDOW\\_SIZE](#) "main\_window\_size.txt"

4.15.2.2 #define [FILENAME\\_SYSTEM\\_PATH](#) "system\_path.txt"

Define filename of the file which contain the system path

#### 4.15.2.3 #define FILENAME\_TOOLBAR\_BUTTON\_PREFERENCES "toolbar\_button\_preferences.txt"

Define filename of the file which contain the toolbar button preferences

#### 4.15.2.4 #define PREFERENCES\_FOLDER\_NAME ".csuper"

Define the folder name of the csuper preferences

### 4.15.3 Function Documentation

#### 4.15.3.1 bool changeSystemPath ( char \* *new\_path* )

Change the system path

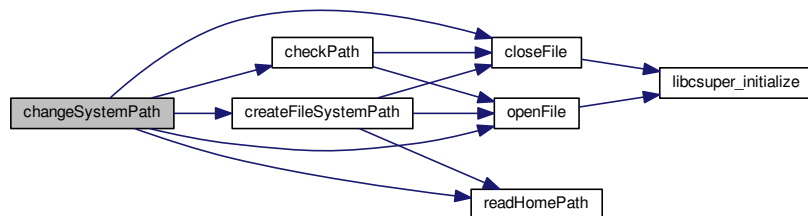
Parameters

<i>in, out</i>	<i>*new_path</i>	the new path
----------------	------------------	--------------

Returns

true if everything is OK, false otherwise

Here is the call graph for this function:



#### 4.15.3.2 bool createFileMainWidowSize ( char \* *home\_path*, main\_window\_size *size* )

Create the file which contain the main window size

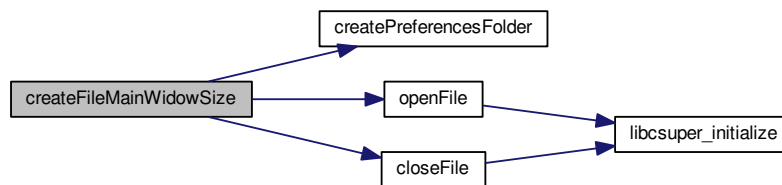
Parameters

<i>in</i>	<i>home_path</i>	the path to the home directory
<i>in</i>	<i>size</i>	the size of the main window

**Returns**

true if everything is OK, false otherwise

Here is the call graph for this function:

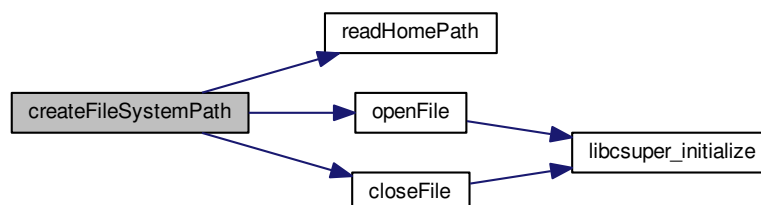
**4.15.3.3 bool createFileSystemPath ( )**

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:

**4.15.3.4 bool createFileToolBarButtonPreferences ( char \* home\_path, toolbar\_button\_preferences\_struct toolbar )**

Create the file which contain the preferences for the toolbar button

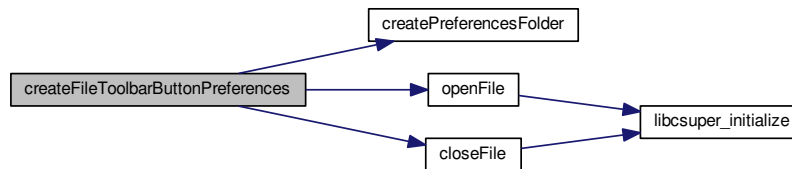
**Parameters**

in	<i>home_path</i>	the path to the home directory
in	<i>toolbar</i>	the toolbar button preferences

**Returns**

true if everything is OK, false otherwise

Here is the call graph for this function:



#### 4.15.3.5 void createPreferencesFolder ( char \* *home\_path* )

Create the folder which contain all preferences

**Parameters**

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

#### 4.15.3.6 bool differsToolbarButtonPreferencesStruct ( toolbar\_button\_preferences\_struct *toolbar1*, toolbar\_button\_preferences\_struct *toolbar2* )

Test if the two toolbar button preferences are different

**Parameters**

in	<i>toolbar1</i>	the first toolbar button preferences
in	<i>toolbar2</i>	the second toolbar button preferences

**Returns**

true if everything is OK, false otherwise

#### 4.15.3.7 bool readFileMainWidowSize ( char \* *home\_path*, main\_window\_size \* *size* )

Read the file which contain the main window size

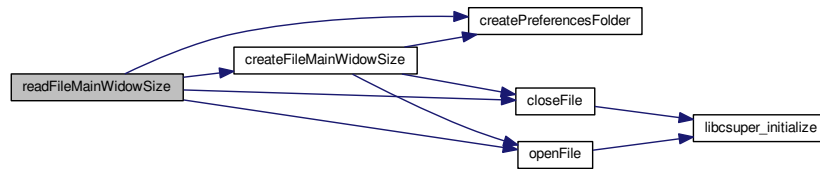
**Parameters**

in	<i>home_path</i>	the path to the home directory
in	<i>size</i>	the size of the main window

**Returns**

true if everything is OK, false otherwise

Here is the call graph for this function:



#### 4.15.3.8 bool readFileSystemPath ( char \* *file\_name* )

Read the system path and the path read to the filename

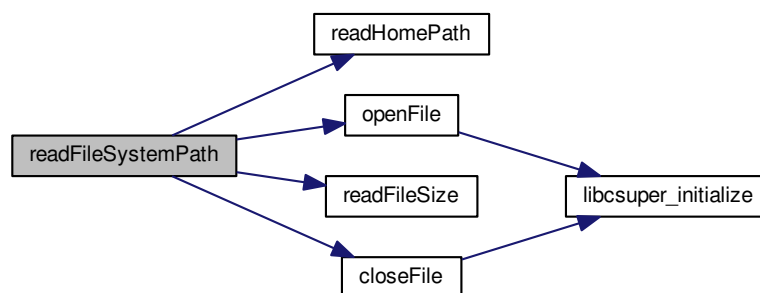
**Parameters**

<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:



#### 4.15.3.9 bool readFileToolBarButtonPreferences ( char \* *home\_path*, toolbar\_button\_preferences\_struct \* *toolbar* )

Read the file which contain the preferences for the toolbar button

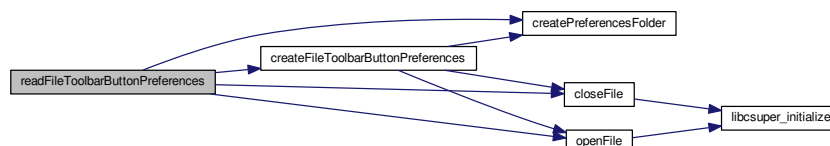
**Parameters**

in	<i>home_path</i>	the path to the home directory
in	<i>toolbar</i>	the toolbar button preferences

**Returns**

true if everything is OK, false otherwise

Here is the call graph for this function:



#### 4.15.3.10 bool readSystemPath ( char \* *file\_name* )

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

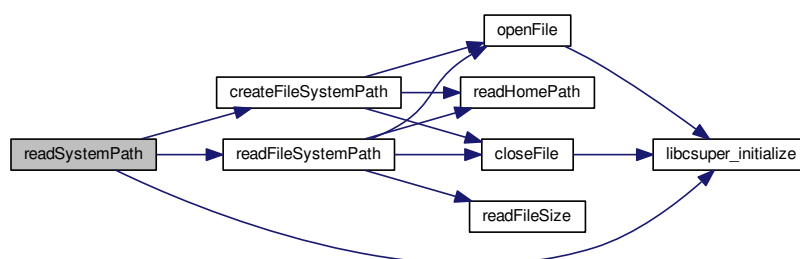
**Parameters**

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

**Returns**

true if everything is OK, false otherwise

Here is the call graph for this function:



## 4.16 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)
- void \* [myAlloc](#) (int size\_alloue)
- void [myRealloc](#) (void \*\*ptr, int size\_alloue)
- char \* [integerToYesNo](#) (int i, char \*yes, char \*no)

### 4.16.1 Detailed Description

Essential function of libcsuper.

#### Author

Remi BERTHO

#### Date

05/07/14

#### Version

4.0.1

### 4.16.2 Function Documentation

#### 4.16.2.1 void clearScreen ( )

Clear the terminal.

Here is the call graph for this function:



#### 4.16.2.2 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  $a < b$

**4.16.2.3 int 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  $a > b$

**4.16.2.4 char \* integerToYesNo ( int i, char \* yes, char \* no )**

Transform an integer to yes or no

**Parameters**

in	<i>i</i>	the integer
in	<i>yes</i>	the yes string
in	<i>no</i>	the no string

**Returns**

yes if  $i > 0$ , no otherwise

**4.16.2.5 void libcsuper\_initialize ( )**

Initialize libcsuper with gettext.

**4.16.2.6 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:

**4.16.2.7 void myRealloc ( void \*\* ptr, int size\_alloue )**

Here is the call graph for this function:

**4.16.2.8 void wrongChoice ( )**

Display an error message.

Here is the call graph for this function:



## 4.17 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>
#include <stdbool.h>
```

## Macros

- `#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)
- void \* [myAlloc](#) (int size\_alloue)
- void [myRealloc](#) (void \*\*ptr, int size\_alloue)
- char \* [integerToYesNo](#) (int i, char \*yes, char \*no)

### 4.17.1 Detailed Description

Header for the essential function of libcsuper.

#### Author

Remi BERTHO

#### Date

25/08/14

#### Version

4.0.2

### 4.17.2 Macro Definition Documentation

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

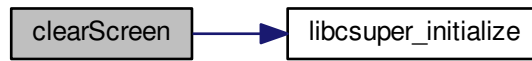
Define the `_` for gettext.

### 4.17.3 Function Documentation

#### 4.17.3.1 `void clearScreen ( )`

Clear the terminal.

Here is the call graph for this function:



#### 4.17.3.2 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  $a < b$

#### 4.17.3.3 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  $a > b$

#### 4.17.3.4 char\* integerToYesNo ( int *i*, char \* *yes*, char \* *no* )

Transform an integer to yes or no

##### Parameters

in	<i>i</i>	the integer
in	<i>yes</i>	the yes string
in	<i>no</i>	the no string

##### Returns

yes if  $i > 0$ , no otherwise

#### 4.17.3.5 void libcsuper\_initialize ( )

Initialize libcsuper with gettext.

#### 4.17.3.6 void\* myAlloc ( int *size\_alloue* )

Allocate a memory block and check if everything is OK.

## Parameters

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

## Returns

a pointer on the allocate memory block

Here is the call graph for this function:



#### 4.17.3.7 void myRealloc ( void \*\* *ptr*, int *size\_alloue* )

Here is the call graph for this function:



#### 4.17.3.8 void wrongChoice ( )

Display an error message.

Here is the call graph for this function:



# Index

- - share.h, [74](#)
- about
  - toolbar\_button\_preferences\_struct, [9](#)
- addConfigListFile
  - game\_config.c, [41](#)
  - game\_config.h, [49](#)
- addDistributorCsuStruct
  - csu\_struct.c, [19](#)
  - csu\_struct.h, [25](#)
- addFileCsuExtension
  - filename.c, [36](#)
  - filename.h, [38](#)
- begin\_score
  - game\_config, [7](#)
- changeSystemPath
  - preferences\_files.c, [60](#)
  - preferences\_files.h, [66](#)
- checkFilename
  - filename.c, [36](#)
  - filename.h, [39](#)
- checkPath
  - filename.c, [36](#)
  - filename.h, [39](#)
- clearScreen
  - share.c, [71](#)
  - share.h, [74](#)
- closeCsuStruct
  - csu\_struct.c, [19](#)
  - csu\_struct.h, [26](#)
- closeFile
  - file.c, [30](#)
  - file.h, [33](#)
- closeListGroupConfig
  - game\_config.c, [41](#)
  - game\_config.h, [50](#)
- compareFloatAscending
  - share.c, [71](#)
  - share.h, [75](#)
- compareFloatDescending
  - share.c, [72](#)
  - share.h, [75](#)
- config
  - csuStruct, [5](#)
- copy
  - toolbar\_button\_preferences\_struct, [9](#)
- copyCsuStruct
  - csu\_struct.c, [20](#)
  - csu\_struct.h, [26](#)
- createFileMainWidowSize
  - preferences\_files.c, [60](#)
  - preferences\_files.h, [66](#)
- createFileSystemPath
  - preferences\_files.c, [61](#)
  - preferences\_files.h, [67](#)
- createFileToolBarButtonPreferences
  - preferences\_files.c, [61](#)
  - preferences\_files.h, [67](#)
- createPreferencesFolder
  - preferences\_files.c, [62](#)
  - preferences\_files.h, [68](#)
- csu\_files.c, [13](#)
  - openFileCsuExtension, [13](#)
  - readCsuFile, [14](#)
  - writeCsuFile, [14](#)
  - writeFileNewTurn, [15](#)
- csu\_files.h, [15](#)
  - FILE\_EXTENSION, [16](#)
  - openFileCsuExtension, [16](#)
  - readCsuFile, [17](#)
  - writeCsuFile, [17](#)
  - writeFileNewTurn, [18](#)
- csu\_struct.c, [18](#)
  - addDistributorCsuStruct, [19](#)
  - closeCsuStruct, [19](#)
  - copyCsuStruct, [20](#)
  - differeentsPlayerName, [20](#)
  - endNewTurn, [20](#)
  - exceedMaxNumber, [21](#)
  - maxNbTurn, [21](#)
  - newCsuStruct, [21](#)
  - rankCalculation, [21](#)
  - searchIndexFromPosition, [23](#)
  - searchPlayerIndex, [23](#)
  - startNewTurn, [24](#)
- csu\_struct.h, [24](#)
  - addDistributorCsuStruct, [25](#)
  - closeCsuStruct, [26](#)
  - copyCsuStruct, [26](#)
  - differeentsPlayerName, [26](#)
  - endNewTurn, [27](#)
  - exceedMaxNumber, [27](#)
  - maxNbTurn, [27](#)
  - newCsuStruct, [27](#)
  - rankCalculation, [28](#)
  - SIZE\_MAX\_NAME, [25](#)



- searchIndexFromPosition, 28
- searchPlayerIndex, 29
- startNewTurn, 29
- VERSION, 25
- csuStruct, 5
  - config, 5
  - day, 5
  - distributor, 5
  - month, 5
  - nb\_player, 6
  - nb\_turn, 6
  - player\_names, 6
  - point, 6
  - rank, 6
  - size\_max\_name, 6
  - total\_points, 6
  - version, 6
  - year, 6
- cut
  - toolbar\_button\_preferences\_struct, 9
- day
  - csuStruct, 5
- decimal\_place
  - game\_config, 7
- delete
  - toolbar\_button\_preferences\_struct, 9
- deleteFile
  - file.c, 31
  - file.h, 33
- differeentsPlayerName
  - csu\_struct.c, 20
  - csu\_struct.h, 26
- differeentsToolbarButtonPreferencesStruct
  - preferences\_files.c, 62
  - preferences\_files.h, 68
- displayHelp
  - main\_argument.c, 56
  - main\_argument.h, 59
- distributor
  - csuStruct, 5
- endNewTurn
  - csu\_struct.c, 20
  - csu\_struct.h, 27
- exceedMaxNumber
  - csu\_struct.c, 21
  - csu\_struct.h, 27
- exportConfigFile
  - game\_config.c, 43
  - game\_config.h, 50
- FILE\_EXTENSION
  - csu\_files.h, 16
- file.c, 30
  - closeFile, 30
  - deleteFile, 31
  - openFile, 31
  - readFileSize, 32
  - renameFile, 32
- file.h, 32
  - closeFile, 33
  - deleteFile, 33
  - openFile, 34
  - readFileSize, 34
  - renameFile, 35
- filename.c, 35
  - addFileCsuExtension, 36
  - checkFilename, 36
  - checkPath, 36
  - getFolderFromFilename, 37
  - getSimpleFilenameFromFullFilename, 37
  - readHomePath, 37
  - readHomePathSlash, 38
- filename.h, 38
  - addFileCsuExtension, 38
  - checkFilename, 39
  - checkPath, 39
  - getFolderFromFilename, 39
  - getSimpleFilenameFromFullFilename, 40
  - readHomePath, 40
  - readHomePathSlash, 40
- first\_way
  - game\_config, 7
- game\_config, 6
  - begin\_score, 7
  - decimal\_place, 7
  - first\_way, 7
  - max, 7
  - name, 7
  - nb\_max, 7
  - turn\_based, 7
  - use\_distributor, 7
- game\_config.c, 40
  - addConfigListFile, 41
  - closeListGameConfig, 41
  - exportConfigFile, 43
  - importConfigFile, 43
  - makeConfigListFile, 44
  - newConfigFile, 44
  - newListGameConfig, 45
  - readConfigFile, 45
  - readConfigListFile, 46
  - removeConfigFile, 46
  - removeConfigListFile, 47
- game\_config.h, 47
  - addConfigListFile, 49
  - closeListGameConfig, 50
  - exportConfigFile, 50
  - importConfigFile, 51
  - makeConfigListFile, 51
  - newConfigFile, 52
  - newListGameConfig, 52
  - readConfigFile, 53
  - readConfigListFile, 53
  - removeConfigFile, 54
  - removeConfigListFile, 54

- game\_configuration\_preferences
  - toolbar\_button\_preferences\_struct, 10
- getFolderFromFilename
  - filename.c, 37
  - filename.h, 39
- getSimpleFilenameFromFullFilename
  - filename.c, 37
  - filename.h, 40
- HELP
  - main\_argument.h, 58
- height
  - main\_window\_size, 8
- importConfigFile
  - game\_config.c, 43
  - game\_config.h, 51
- integerToYesNo
  - share.c, 72
  - share.h, 75
- is\_maximize
  - main\_window\_size, 8
- libcsuper.h, 55
  - NOT\_LIBCSUPER, 56
- libcsuper\_initialize
  - share.c, 72
  - share.h, 75
- list\_game\_config, 8
  - name\_game\_config, 8
  - nb\_config, 8
- main\_argument.c, 56
  - displayHelp, 56
  - searchArgument, 56
- main\_argument.h, 57
  - displayHelp, 59
  - HELP, 58
  - OPEN\_FILE, 58
  - READ\_FILE, 58
  - STRING\_HELP, 58
  - STRING\_HELP\_RED, 58
  - STRING\_OPEN\_FILE, 58
  - STRING\_READ\_FILE, 58
  - searchArgument, 59
- main\_window\_size, 8
  - height, 8
  - is\_maximize, 8
  - width, 8
- makeConfigListFile
  - game\_config.c, 44
  - game\_config.h, 51
- max
  - game\_config, 7
- maxNbTurn
  - csu\_struct.c, 21
  - csu\_struct.h, 27
- month
  - csuStruct, 5
- myAlloc
  - share.c, 72
  - share.h, 75
- myRealloc
  - share.c, 73
  - share.h, 77
- NOT\_LIBCSUPER
  - libcsuper.h, 56
- name
  - game\_config, 7
- name\_game\_config
  - list\_game\_config, 8
- nb\_config
  - list\_game\_config, 8
- nb\_max
  - game\_config, 7
- nb\_player
  - csuStruct, 6
- nb\_turn
  - csuStruct, 6
- new
  - toolbar\_button\_preferences\_struct, 10
- newConfigFile
  - game\_config.c, 44
  - game\_config.h, 52
- newCsuStruct
  - csu\_struct.c, 21
  - csu\_struct.h, 27
- newListGameConfig
  - game\_config.c, 45
  - game\_config.h, 52
- OPEN\_FILE
  - main\_argument.h, 58
- open
  - toolbar\_button\_preferences\_struct, 10
- openFile
  - file.c, 31
  - file.h, 34
- openFileCsuExtension
  - csu\_files.c, 13
  - csu\_files.h, 16
- paste
  - toolbar\_button\_preferences\_struct, 10
- player\_names
  - csuStruct, 6
- point
  - csuStruct, 6
- preferences
  - toolbar\_button\_preferences\_struct, 10
- preferences\_files.c, 59
  - changeSystemPath, 60
  - createFileMainWidowSize, 60
  - createFileSystemPath, 61
  - createFileToolBarButtonPreferences, 61
  - createPreferencesFolder, 62
  - diffrentsToolBarButtonPreferencesStruct, 62

- readFileMainWidowSize, 62
- readFileSystemPath, 63
- readFileToolBarButtonPreferences, 63
- readSystemPath, 64
- preferences\_files.h, 64
  - changeSystemPath, 66
  - createFileMainWidowSize, 66
  - createFileSystemPath, 67
  - createFileToolBarButtonPreferences, 67
  - createPreferencesFolder, 68
  - differeentsToolBarButtonPreferencesStruct, 68
  - readFileMainWidowSize, 68
  - readFileSystemPath, 69
  - readFileToolBarButtonPreferences, 69
  - readSystemPath, 70
- properties
  - toolbar\_button\_preferences\_struct, 10
- READ\_FILE
  - main\_argument.h, 58
- rank
  - csuStruct, 6
- rankCalculation
  - csu\_struct.c, 21
  - csu\_struct.h, 28
- readConfigFile
  - game\_config.c, 45
  - game\_config.h, 53
- readConfigListFile
  - game\_config.c, 46
  - game\_config.h, 53
- readCsuFile
  - csu\_files.c, 14
  - csu\_files.h, 17
- readFileMainWidowSize
  - preferences\_files.c, 62
  - preferences\_files.h, 68
- readFileSize
  - file.c, 32
  - file.h, 34
- readFileSystemPath
  - preferences\_files.c, 63
  - preferences\_files.h, 69
- readFileToolBarButtonPreferences
  - preferences\_files.c, 63
  - preferences\_files.h, 69
- readHomePath
  - filename.c, 37
  - filename.h, 40
- readHomePathSlash
  - filename.c, 38
  - filename.h, 40
- readSystemPath
  - preferences\_files.c, 64
  - preferences\_files.h, 70
- redo
  - toolbar\_button\_preferences\_struct, 10
- removeConfigFile
  - game\_config.c, 46
- game\_config.h, 54
- removeConfigListFile
  - game\_config.c, 47
  - game\_config.h, 54
- renameFile
  - file.c, 32
  - file.h, 35
- SIZE\_MAX\_NAME
  - csu\_struct.h, 25
- STRING\_HELP
  - main\_argument.h, 58
- STRING\_HELP\_RED
  - main\_argument.h, 58
- STRING\_OPEN\_FILE
  - main\_argument.h, 58
- STRING\_READ\_FILE
  - main\_argument.h, 58
- save\_as
  - toolbar\_button\_preferences\_struct, 10
- searchArgument
  - main\_argument.c, 56
  - main\_argument.h, 59
- searchIndexFromPosition
  - csu\_struct.c, 23
  - csu\_struct.h, 28
- searchPlayerIndex
  - csu\_struct.c, 23
  - csu\_struct.h, 29
- separator\_1
  - toolbar\_button\_preferences\_struct, 10
- separator\_2
  - toolbar\_button\_preferences\_struct, 10
- separator\_3
  - toolbar\_button\_preferences\_struct, 10
- separator\_4
  - toolbar\_button\_preferences\_struct, 10
- separator\_5
  - toolbar\_button\_preferences\_struct, 11
- share.c, 70
  - clearScreen, 71
  - compareFloatAscending, 71
  - compareFloatDescending, 72
  - integerToYesNo, 72
  - libcsuper\_initialize, 72
  - myAlloc, 72
  - myRealloc, 73
  - wrongChoice, 73
- share.h, 73
  - \_, 74
  - clearScreen, 74
  - compareFloatAscending, 75
  - compareFloatDescending, 75
  - integerToYesNo, 75
  - libcsuper\_initialize, 75
  - myAlloc, 75
  - myRealloc, 77
  - wrongChoice, 77
- size\_max\_name

- csuStruct, [6](#)
- startNewTurn
  - csu\_struct.c, [24](#)
  - csu\_struct.h, [29](#)
- toolbar\_button\_preferences
  - toolbar\_button\_preferences\_struct, [11](#)
- toolbar\_button\_preferences\_struct, [9](#)
  - about, [9](#)
  - copy, [9](#)
  - cut, [9](#)
  - delete, [9](#)
  - game\_configuration\_preferences, [10](#)
  - new, [10](#)
  - open, [10](#)
  - paste, [10](#)
  - preferences, [10](#)
  - properties, [10](#)
  - redo, [10](#)
  - save\_as, [10](#)
  - separator\_1, [10](#)
  - separator\_2, [10](#)
  - separator\_3, [10](#)
  - separator\_4, [10](#)
  - separator\_5, [11](#)
  - toolbar\_button\_preferences, [11](#)
  - undo, [11](#)
- total\_points
  - csuStruct, [6](#)
- turn\_based
  - game\_config, [7](#)
- undo
  - toolbar\_button\_preferences\_struct, [11](#)
- use\_distributor
  - game\_config, [7](#)
- VERSION
  - csu\_struct.h, [25](#)
- version
  - csuStruct, [6](#)
- width
  - main\_window\_size, [8](#)
- writeCsuFile
  - csu\_files.c, [14](#)
  - csu\_files.h, [17](#)
- writeFileNewTurn
  - csu\_files.c, [15](#)
  - csu\_files.h, [18](#)
- wrongChoice
  - share.c, [73](#)
  - share.h, [77](#)
- year
  - csuStruct, [6](#)